Further studies of extreme learning machine and compressed signal detection

Cao, Jiuwen

2013

Cao, J. (2013). Further studies of extreme learning machine and compressed signal detection. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/53458>

<https://doi.org/10.32657/10356/53458>

Further Studies of Extreme Learning Machine and Compressed Signal Detection

Cao Jiuwen

**School of Electrical and Electronic Engineering
Nanyang Technological University**

A Thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

2013

Acknowledgements

This thesis would not have been possible without the guides, support and encouragement of numerous individuals, mentors, friends, and family.

First, I would like to express my deepest gratitude to my supervisor, Associate Professor Lin Zhiping and co-supervisor Associate Professor Huang Guang-Bin for their constant encouragement and guidance during the period of my PhD study. I also would like to thank technicians from the Software Engineering lab and the Center for Signal Processing, in School of Electrical and Electronic Engineering, Nanyang Technological University, for their technical support during my study. I would like to thank Nanyang Technological University for the four years financial support of my PhD scholarship.

Second, I would like to thank many friends and fellow classmates in Singapore for their support and help.

Finally, and most importantly, I would like to thank my beloved parents and my wife for the endless love and selfless support they have given me throughout the long journey of my study.

Abstract

In this thesis, we present further studies of extreme learning machine and signal detection in compressed sensing.

In Chapter 1, we give literature reviews of extreme learning machine (ELM) and compressed sensing (CS).

In part I of the thesis (Chapters 2, 3, and 4), we consider the recent ELM for training neural networks and present several improved algorithms. We first propose a composite function wavelet neural network (WNN) learning with the recent ELM algorithm in Chapter 2. The main contributions of the proposed WNN comparing with traditional ones are using composite functions at the hidden nodes and applying ELM algorithm to WNN as a learning algorithm. To reduce the network size and optimize the hidden node parameters, we then introduce an improvement method for training the proposed WNN by incorporating the global optimization algorithm Differential Evolution into searching for the optimal network input weights and the dilation and translation values in Chapter 3. To further enhance the classification rate of the ELM, we propose an improved algorithm named voting based ELM (V-ELM) for signal classification in Chapter 4. In V-ELM, the voting method is incorporated into the ELM in classification applications. Several individual ELMs with the same network structure are trained

with the same dataset and the final class label of a test sample is determined by majority voting method on all the results obtained by these independent ELMs. Numerical simulations are provided to illustrate the efficiency of our proposed methods.

In part II of the thesis (Chapters 5 and 6), we study the signal detection in CS. We first consider the theoretical bound of the probability of error by detecting the signal reconstructed in CS with the Bayesian approach in Chapter 5. Utilizing the oracle estimator in CS, we provide a theoretical bound of the probability of error when the noise in CS is white Gaussian noise. We then consider the Bayesian approach to signal detection in CS using compressed measurements directly in Chapter 6. We start by revisiting the classical signal detection problem and show that with an additive Gaussian noise, the probability of error for unequal prior probabilities of the hypotheses is always smaller than the one with equal prior probability. We then consider signal detection with compressed measurements directly, assuming that the additive noise is Gaussian but with unequal variances. A general expression is obtained for the probability of error where the prior probabilities could be equal or unequal. We have also derived performance bounds for the probability of error using the *restricted isometry property* constant and then the computationally more feasible mutual coherence of a given sampling matrix in CS. An approximate but simpler expression of the probability of error and its approximate upper bound are also obtained. Numerical simulations are given to verify the new theoretical results.

In Chapter 7, conclusions and future work are provided.

Contents

Contents	iv
List of Tables	viii
List of Figures	x
List of Symbols	xiii
1 Introduction	1
1.1 Extreme Learning Machine	1
1.2 Compressed Sensing	8
1.3 Motivations and Main Contributions	13
Part I Improvements on Extreme Learning Machine	16
2 Composite Function Wavelet Neural Networks with Extreme Learning Machine	18
2.1 Introduction	18
2.2 The Network Structure of Composite Function Wavelet Neural Network	21
2.3 Learning Algorithm: CFWNN-ELM	24

2.3.1	Parameter Initialization	25
2.3.2	CFWNN-ELM	27
2.4	Performance Evaluations	28
2.4.1	Evaluation on Regression Problems	29
2.4.2	Evaluation on Classification using Real-world Datasets	37
2.5	Discussions	41
2.6	Conclusions	45
3	Composite Function Wavelet Neural Networks with Differential Evolution and Extreme Learning Machine	46
3.1	Introduction	46
3.2	Differential Evolution	48
3.3	CWN-E-ELM	50
3.4	Performance Evaluations	54
3.4.1	Evaluation on Regression Problems	55
3.4.2	Evaluation on Classification using Real-world Datasets	65
3.5	Conclusions	68
4	Voting Based Extreme Learning Machine for Signal Classification	70
4.1	Introduction	70
4.2	Voting Based Extreme Learning Machine	72
4.3	Performance Evaluations	76
4.3.1	Comparisons with ELM	77
4.3.2	Comparisons with SVM, OP-ELM, BP, and KNN	79
4.3.3	Comparisons with several other recent algorithms	82
4.4	Discussions	83

4.5	Conclusions	88
4.6	Appendix: Proof of Proposition 4.1	89
4.7	Appendix: Numerical simulations to illustrate Propositions 4.1 to 4.3	91
 Part II Compressed Signal Detection		95
5	The Detection Bound of the Probability of Error in Compressed Sensing	97
5.1	Introduction	97
5.2	Bayesian Signal Detection	99
5.3	Oracle Estimator	100
5.4	The Bound of Probability of Error in Compressed Sensing	102
5.5	Simulations	104
5.5.1	Simulation set-ups	104
5.5.2	Comparison of the results	104
5.6	Conclusions	106
5.7	Appendix: Proof of Theorem 5.1	107
5.8	Appendix: Proof of Lemma 5.1	110
6	Bayesian Signal Detection with Compressed Measurements	111
6.1	Introduction	111
6.2	General Detection Performance Analysis with Prior Probabilities . .	114
6.3	Detection with Compressed Measurements	116
6.3.1	Signal detection with compressed measurements	116
6.3.2	Approximation of the probability of error and its upper bound with compressed measurements	124

6.4	Simulations	127
6.4.1	General Bayesian signal detection	127
6.4.2	Probability of error $\tilde{P}_e$ in Bayesian detection with compressed measurements	129
6.5	Conclusions	136
6.6	Appendix: Proof of Theorem 6.1	136
6.7	Appendix: Proof of Theorem 6.2	140
6.8	Appendix: Proof of Theorem 6.3	141
6.9	Appendix: Proof of Theorem 6.4	142
6.10	Appendix: Proof of Theorem 6.5	144
7	Conclusions and Future Work	146
7.1	Conclusions	146
7.2	Future Work	149
	Author's Publication	154
	References	156

List of Tables

2.1	Results comparisons for function approximations.	31
2.2	Performance index comparisons for learning 2-D nonlinear function f_3	34
2.3	Specification of datasets.	35
2.4	Results comparisons for regressions of real-world datasets.	36
2.5	Specifications of classification datasets	38
2.6	Results comparisons for classification on datasets.	39
2.7	Comparisons of network complexities.	40
2.8	Performance comparison with some traditional algorithms in real medical diagnosis application: Diabetes.	41
3.1	Comparisons for function approximations	57
3.2	Comparisons for regressions of real-world datasets	64
3.3	Results comparisons for classification on datasets.	66
3.4	Comparisons of network complexities.	67
4.1	Performance comparisons with ELM	78
4.2	Comparisons of network complexities with ELM	79
4.3	Performance comparisons with SVM, OP-ELM, BP, and KNN . . .	80

LIST OF TABLES

4.4	Comparisons of network complexities with SVM, OP-ELM, BP, and KNN	81
4.5	Performance comparisons with several recent methods	83
4.6	Comparisons of misclassification numbers between V-ELM and ELM	93

List of Figures

1.1	The structure of a single hidden layer feedforward neural network. .	4
2.1	The network structure of composite function wavelet neural networks.	22
2.2	The approximation curves of the ‘SinC’ function using 500 samples by CFWNN-ELM, SLFN-ELM, CFWNN-BP and SVR.	32
2.3	The generalization performance of CFWNN-ELM is stable on a wide range of number of hidden nodes.	33
2.4	Distribution of the input samples and the hidden nodes from two separate regions.	43
2.5	Distribution of the input samples and the hidden nodes from one region.	44
3.1	Flow chart of the proposed CWN-E-ELM.	51
3.2	Testing RMSEs of the two-dimensional function f_3 w.r.t different number of nodes.	58
3.3	Testing RMSEs of the two-dimensional function f_3 w.r.t different sizes of samples.	59
3.4	Approximation error surfaces of the two-dimensional function by CWN-E-ELM.	60

LIST OF FIGURES

3.5	Approximation error surfaces of the two-dimensional function by SLFN-E-ELM.	61
3.6	Approximation error surfaces of the two-dimensional function by CFWNN-ELM.	62
3.7	Approximation error surfaces of the two-dimensional function by SVR.	63
4.1	Classifiers using ELM with different random parameters.	71
4.2	Algorithmic description of the proposed V-ELM.	75
4.3	Success rates of the datasets Heart and Sonar using V-ELM (Sigmoid) with respect to the independent training number K	85
4.4	Standard deviations of success rates of the datasets Heart and Sonar using V-ELM (Sigmoid) with respect to the independent training number K	86
4.5	Misclassification numbers of the three samples with respect to the independent training number K	93
5.1	Comparisons of the detection performance by detecting the signal reconstructed with four recovery algorithms (BPDN, Dantzig, OMP, CoSaMP) and theoretical bound with respect to different variance σ^2 values.	105
6.1	The simulation results of P_e with respect to different prior probabilities and two different values of $\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}$	128
6.2	Upper, lower bounds, and the true value of the probability of error on different prior probabilities.	129

LIST OF FIGURES

6.3	Upper, lower bounds, and the true value of the probability of error with respect to signal power.	131
6.4	The probability of errors comparisons with the approximation result $\tilde{\bar{P}}_e$ and the true value $\tilde{P}_e$ on different prior probabilities.	133
6.5	Comparisons of performance bounds for the probability of error on different SNR levels.	135

List of Symbols

$\mathbf{x}$	Vector
A	Matrix
A^T	The transpose of the matrix A
A^{-1}	The inverse of the matrix A
I	Identity Matrix
$\ \mathbf{x}\ _0$	The support set of the vector $\mathbf{x}$
$\ \cdot\ _1$	ℓ_1 norm
$\ \cdot\ _2$	ℓ_2 norm
$\langle \mathbf{x}, \mathbf{y} \rangle$	The inner product of the vector $\mathbf{x}$ and the vector $\mathbf{y}$
$H^\dagger$	The Moore-Penrose generalized inverse of the matrix H
$\lambda_{\max,A}$	The maximal eigenvalue of the matrix A
$\lambda_{\min,A}$	The minimal eigenvalue of the matrix A
$A = \text{diag}\{a_1, \dots, a_n\}$	A is a diagonal matrix with $a_1, \dots, a_n$ as its diagonal entries

Chapter 1

Introduction

This thesis includes the Introduction Chapter and two parts. In the first part, we investigate the recent extreme learning machine (ELM) [1–3] for neural networks and propose several improved algorithms based on ELM. In the second part, we study the signal detection in compressed sensing (CS) [4–11] and present several new results on signal detection with the Bayesian approach [12].

In this chapter, we first give a brief review of the extreme learning machine algorithm in Section 1.1. The recent compressed sensing and compressed signal detection are reviewed in Section 1.2. Section 1.3 describes the motivations of this thesis and summarizes our contributions.

1.1 Extreme Learning Machine

Computational intelligence techniques have been widely used in many real world applications in recent years. Among these techniques, neural networks [13] have played an important role, such as for function regressions [14, 15], signal classifi-

cations [16–18], pattern recognitions [19–21], system controls [22, 23], and so on. Among different architectures of neural networks, feedforward neural networks were one of the most popular neural networks. The mainly used training algorithm for feedforward neural networks is the gradient-descent based approach, e.g., the Backpropagation (BP) algorithm [24]. Although the BP algorithm has exhibited many promising features in generalization capabilities, it still has some limitations, such as sticking at local minima, slow convergence performance, and poor computational scalability. Recently, an algorithm called extreme learning machine was proposed for training single hidden layer feedforward neural networks (SLFNs). ELM claims that random computational hidden nodes which are possibly independent of the training data can be used and the hidden layer of SLFNs may not need to be tuned during the training phase. As an emergent algorithm for SLFNs, ELM has attracted much attention in recent years [25–40] and can efficiently overcome some limitations faced by the conventional BP approach. Moreover, ELM can generally obtain good generalization performance with faster learning speed than some state-of-art methods, such as the Backpropagation (BP) algorithm and the support vector machines (SVM) [41].

Extreme learning machine is based on the concept that SLFNs with N random hidden nodes can exactly learn N distinct observations. It was built on the assumptions that hidden parameters are generated according to any continuous probability distribution with probability one and the activation functions are infinitely differentiable. The SLFN becomes a linear system after the hidden node parameters are randomly assigned *a priori* and the network output weights can then be derived analytically by a simple generalized inverse of the hidden layer output matrix. The merit of ELM compared with traditional gradient-based algorithms is

that it does not need to tune all the network parameters step by step and hence avoids many difficulties faced by gradient-based algorithms such as learning rate, learning epochs and local minima. It also has shown that ELM works well both on SLFNs with additive hidden nodes [2] and SLFNs with radial basis function (RBF) hidden nodes [42]. This method directly leads to its fast learning speed and also shows good generalization performance. This is the main reason that ELM has been widely used in many real world applications [25, 26, 28] and many improvement methods have been developed based on ELM [29, 30, 34]. The universal approximation capability of ELM has been proved in [27, 31–33].

For N arbitrary distinct samples $(\mathbf{x}_k, \mathbf{t}_k) \in \mathbf{R}^d \times \mathbf{R}^m$, the standard SLFNs with L hidden nodes and an activation function $G(\cdot)$ are mathematically described as

$$\mathbf{o}_i = \sum_{j=1}^L \beta_j G(\mathbf{a}_j, b_j, \mathbf{x}_i), \quad i = 1, 2, \dots, N \quad (1.1)$$

where $\mathbf{a}_j \in \mathbf{R}^d$ and $b_j \in \mathbf{R}$ ($j = 1, 2, \dots, L$) are learning parameters of the j th hidden node, respectively. $\beta_j \in \mathbf{R}^m$ is the link connecting the j th hidden node to the output node. $G(\mathbf{a}_j, b_j, \mathbf{x}_i)$ is the output of the j th hidden node with respect to the input sample $\mathbf{x}_i$. $\mathbf{o}_i$ is the network output with respect to the input sample $\mathbf{x}_i$. Fig. 1.1 depicts a general structure of a SLFN [3].

To minimize the cost function $E = \sum_{i=1}^N \|\mathbf{o}_i - \mathbf{t}_i\|$, the BP algorithm is to iteratively update the network parameter W according to the following equation

$$W_{k+1} = W_k - \eta \frac{\partial E(W_k)}{\partial W_k} \quad (1.2)$$

where W is a set of $(\mathbf{a}_j, b_j, \beta_j)$ and η is the learning rate. The learning stops when the cost function E arrives at a preset threshold or the iteration reaches the

Figure 1.1: The structure of a single hidden layer feedforward neural network.

maximal steps. Although this algorithm works well in many applications, it still has several issues, such as local minima, overfitting, slow convergence rate, etc.

For the mathematical model in (1.1), ELM states that the hidden layer of SLFNs need not be tuned. By using random computational nodes in the hidden layer, (1.1) can be written as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \quad (1.3)$$

where

$$\mathbf{H} = \begin{bmatrix} G(\mathbf{x}_1; b_1, \mathbf{a}_1) & \cdots & G(\mathbf{x}_1; b_L, \mathbf{a}_L) \\ \vdots & \ddots & \vdots \\ G(\mathbf{x}_N; b_1, \mathbf{a}_1) & \cdots & G(\mathbf{x}_N; b_L, \mathbf{a}_L) \end{bmatrix}_{N \times L} \quad (1.4)$$

$$\boldsymbol{\beta} = \left(\boldsymbol{\beta}_1^T \quad \boldsymbol{\beta}_2^T \quad \cdots \quad \boldsymbol{\beta}_L^T \right)_{m \times L}^T \quad (1.5)$$

and the target output

$$\mathbf{T} = \left(\mathbf{t}_1^T \quad \mathbf{t}_2^T \quad \cdots \quad \mathbf{t}_L^T \right)_{m \times N}^T. \quad (1.6)$$

$\mathbf{H}$ is called the hidden layer output matrix.

The ELM algorithm is based on the following two principles [2].

- When the number of training samples equals the number of hidden nodes, i.e., $N = L$, one can randomly assign the parameters of hidden nodes (the input weights and biases for additive hidden nodes or the centers and impact factors for RBF) and based on this analytically calculate the output weights by simply inverting $\mathbf{H}$ and realize zero training error. Calculation of the output weights is done in a single step here. There is no need for

any lengthy training procedure where the network parameters are adjusted interactively with appropriately chosen control parameters (learning rate and learning epochs, etc.) [2].

- When the number of training samples is greater than the number of hidden nodes, i.e., $N > L$, one can still randomly assign the parameters of hidden nodes and calculate the output weights by using a pseudoinverse of $\mathbf{H}$ to give a small nonzero training error $\epsilon > 0$. Here also the output weights calculation is done in a single step and does not need lengthy training procedure. These have been formally stated in the following two theorems [2].

Theorem 1.1. [2] *Given a standard SLFN with N hidden nodes and activation function $G(\cdot)$ which is infinitely differentiable in any interval, for N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i \in \mathbf{R}^d$ and $\mathbf{t}_i \in \mathbf{R}^m$, for any $\mathbf{a}_i$ and b_i randomly chosen from any intervals of $\mathbf{R}^d$ and $\mathbf{R}$, respectively, according to any continuous probability distribution, then with probability one, the hidden layer output matrix $\mathbf{H}$ in equation (1.4) of the SLFN is invertible and $\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_2 = 0$.*

Theorem 1.2. [2] *Given any small positive value $\epsilon > 0$ and activation function $G(\cdot)$ which is infinitely differentiable in any interval, there exists $L \leq N$ such that for N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i \in \mathbf{R}^d$ and $\mathbf{t}_i \in \mathbf{R}^m$, for any $\mathbf{a}_i$ and b_i randomly chosen from any intervals of $\mathbf{R}^n$ and $\mathbf{R}$, respectively, according to any continuous probability distribution, then with probability one, $\|\mathbf{H}_{N \times L} \boldsymbol{\beta}_{L \times m} - \mathbf{T}_{N \times m}\|_2 < \epsilon$.*

The detailed proofs of these two theorems can be found in [2].

According to Theorem 1.1 and 1.2, the output weights can be derived by finding

the least-square solutions to the above linear system which is given as

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^\dagger \mathbf{T} \quad (1.7)$$

where $\mathbf{H}^\dagger$ is the Moore-Penrose [43] generalized inverse of the hidden layer output matrix $\mathbf{H}$.

Remark 1.1. *The ELM algorithm has the following important properties:*

1. **Minimum training error.** *The special solution $\hat{\boldsymbol{\beta}} = \mathbf{H}^\dagger \mathbf{T}$ is one of the least-squares solutions of a general linear system $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$, meaning that the smallest training error can be reached by this special solution:*

$$\left\| \mathbf{H}\hat{\boldsymbol{\beta}} - \mathbf{T} \right\|_2 = \left\| \mathbf{H}\mathbf{H}^\dagger \mathbf{T} - \mathbf{T} \right\|_2 = \min_{\boldsymbol{\beta}} \left\| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \right\|_2. \quad (1.8)$$

Although almost all learning algorithms wish to reach the minimum training error, however, most of them cannot reach it because of local minimum or infinite training iteration is usually not allowed in applications.

2. **Smallest norm of weights.** *Further, the special solution $\hat{\boldsymbol{\beta}} = \mathbf{H}^\dagger \mathbf{T}$ has the smallest norm among all the least-squares solutions of $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$:*

$$\left\| \hat{\boldsymbol{\beta}} \right\|_2 \leq \left\| \mathbf{H}^\dagger \mathbf{T} \right\|_2 \leq \left\| \boldsymbol{\beta} \right\|_2, \quad \forall \boldsymbol{\beta} \in \{ \boldsymbol{\beta} : \left\| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \right\|_2 \leq \left\| \mathbf{H}\mathbf{z} - \mathbf{T} \right\|_2, \quad \forall \mathbf{z} \in R^{L \times N} \}. \quad (1.9)$$

3. **Unique solution.** *The minimum norm least-squares solution of $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$ is unique, which is $\hat{\boldsymbol{\beta}} = \mathbf{H}^\dagger \mathbf{T}$.*

The ELM algorithm can be summarized as follows:

Algorithm ELM: Given a training set $\aleph = \{(\mathbf{x}_i, \mathbf{t}_i) \mid \mathbf{x}_i \in \mathbf{R}^d, \mathbf{t}_i \in \mathbf{R}^m, i = 1, 2, \dots, N\}$, the hidden node output function $G(\mathbf{x}; b, \mathbf{a})$ and hidden node number L :

1. Randomly assign hidden node parameters $(b_i, \mathbf{a}_i), i = 1, 2, \dots, L$.
2. Calculate the hidden layer output matrix $\mathbf{H}$ in (1.4).
3. Calculate the output weight β in (1.7).

1.2 Compressed Sensing

The traditional sampling theory in acquiring signals follows the basic principle of the Nyquist/Shannon [44, 45] theory and states that to avoid losing information and to ensure reconstruction without error, the sampling rate should be at least twice of the maximum frequency of the signal. However, this makes it difficult to implement in many real world applications whose maximum frequency is very high. In addition, it is also expensive to store a large amount of sampled data. In many applications, it is wasteful to use the conventional sampling principle to capture signals since for convenient storage and transmission, a large amount of data will be thrown away after acquiring and representing the captured signal in a certain basis. Recently, an alternative sampling theory named “compressive sensing” or “compressed sensing” (CS) [4–11, 46–56] is proposed for sparse signal processing and reconstruction. CS has emerged as a protocol of signal acquisition by directly capturing a compressed signal representation instead of using the traditional sampled-then-compressed framework. It has been shown in CS that instead of using measurements required by the traditional Nyquist-rate, fewer measurements

may be enough for reconstructing the original signals. To make this possible, CS relies on two principles: *sparsity*, which pertains to the signals of interest, and *incoherence*, which pertains to the sensing modality.

- *Sparsity*. A signal is sparse means it has relatively few nonzero terms or its coefficients in some fixed basis have relatively few nonzero entries.
- *Incoherence*. Coherence measures the largest correlation between the sensing basis Φ and the representation basis Ψ and the definition is given below:

$$\mu(\Phi, \Psi) = \sqrt{n} \cdot \max_{1 \leq k, j \leq n} |\langle \varphi_k, \psi_j \rangle|, \quad (1.10)$$

where for a signal $\mathbf{x} \in \mathbf{R}^n$, $\Psi \in \mathbf{R}^{n \times n}$ and $\Phi \in \mathbf{R}^{m \times n}$ with $m < n$. φ_k and ψ_j are the k -th and j -th columns of the matrices Φ and Ψ , respectively.

Generally, compressive sensing is mainly concerned with low coherence pairs.

Instead of adopting the sinc interpolation approach to reconstruct the signal in the Nyquist/Shannon sampling framework, CS achieves the signal recovery by solving a convex optimization problem or by implementing iterative algorithms.

In the following, we assume that signal $\mathbf{s}$ is sparse after representing in some basis. Generally, in compressed sensing, a signal is acquired with

$$\mathbf{y} = \Phi \mathbf{s} + \mathbf{w} \quad (1.11)$$

where Φ is a $m \times n$ sampling matrix, $\mathbf{y}$ is the measured vector and $\mathbf{w}$ represents the noise vector. In the CS framework, the signal $\mathbf{s}$ can be recovered from the observation acquired by an underdetermined linear sampling system with $m \ll n$ provided that $\mathbf{s}$ is sparse in some basis [4, 5]. To get a satisfactory reconstruction,

the sampling matrix Φ plays an important role and is required to satisfy the *restricted isometry property* (RIP) introduced in [5]. For the set $\mathcal{S}_k$ of all k -sparse signals $\mathcal{S}_k = \{\mathbf{s} \in \mathbf{R}^n \mid \|\mathbf{s}\|_0 \leq k\}$, where $\|\mathbf{s}\|_0$ denotes the support of signal $\mathbf{s}$, the matrix Φ is said to satisfy the RIP if there exists $\eta \in (0, 1)$, such that

$$(1 - \eta) \|\mathbf{s}\|_2^2 \leq \|\Phi \mathbf{s}\|_2^2 \leq (1 + \eta) \|\mathbf{s}\|_2^2 \quad (1.12)$$

holds for all $\mathbf{s} \in \mathcal{S}_k$.

Several methods can be used to generate the measurement matrix Φ . Among all these generalization methods, random matrix construction by choosing the entries as independent and identically distributed (*i.i.d.*) Gaussian random variables is considered to be one of the most efficient approaches.

The most frequently used methods are the Dantzig algorithm, the BPDN algorithm, the orthogonal matching pursuit (OMP) algorithm and the compressive sampling matching pursuit (CoSaMP) algorithm. Here, we give a brief description of these four algorithms.

The Dantzig algorithm [47] is to find the solution to the following optimization problem

$$\min_{\mathbf{s} \in \mathbf{R}^n} \|\mathbf{s}\|_1, \text{ subject to } \|\Phi^T(\mathbf{y} - \Phi \mathbf{s})\|_\infty \leq \tau \quad (1.13)$$

where τ is a user-specified parameter. The constraint is to guarantee that the residual vector $(\mathbf{y} - \Phi \mathbf{s})$ is within the noise level.

The BPDN method [48, 57, 58] estimates the sparse signal by finding the solution to the following quadratic problem with an ℓ_1 norm constraint

$$\min_{\mathbf{s} \in \mathbf{R}^n} \frac{1}{2} \|\mathbf{y} - \Phi \mathbf{s}\|_2^2 + \gamma \|\mathbf{s}\|_1 \quad (1.14)$$

where γ is a positive parameter that enforces the level of sparsity.

Initializing the supporting set T_0 of the signal with an empty set and setting the residual vector $\mathbf{r}^0$ to $\mathbf{y}$, the OMP algorithm [46, 59] iteratively updates the estimated signal $\mathbf{s}_{\text{OMP}}^i$ at the i -th step using the least square solution with the updated supporting set T_i . At each iteration i , one atom in matrix Φ , noted as a_i which is most highly correlated with $\mathbf{r}^{i-1}$ is calculated and its index is added to the supporting set with $T_i = T_{i-1} \cup \{a_i\}$. The residual $\mathbf{r}$ is then updated with the following equation

$$\mathbf{r} = \mathbf{y} - \Phi \mathbf{s}_{\text{OMP}}^i \quad (1.15)$$

and the algorithm stops after k iterations and guarantees $\|\mathbf{s}_{\text{OMP}}\|_0 = k$.

The CoSaMP algorithm [50] is an improved algorithm based on OMP. At each iteration, a number of vectors having the largest correlation with $\mathbf{y}$ is selected to built the approximation. Then, the CoSaMP algorithm restricts the approximation to the required level of sparsity by removing all but the required amount of entries.

Besides investigations on recovery algorithms and sampling systems in compressed sensing, several recent researches have pointed out that signal recovery may not be necessary and processing with compressed measurements directly could provide promising results. The pioneering work on processing with compressed measurements directly covers several fundamental signal problems, such as detection, classification, estimation, and filtering [60–66]. Specifically, for detection with compressed measurements in CS, it was first pointed out in [60] and then comprehensively studied in [65, 66]. In [66], a general detection problem has been

considered where the two hypotheses are expressed as follows

$$\begin{aligned}\tilde{\mathcal{H}}_0 : \mathbf{y} &= \Phi \mathbf{w} \\ \tilde{\mathcal{H}}_1 : \mathbf{y} &= \Phi(\mathbf{s} + \mathbf{w})\end{aligned}\tag{1.16}$$

It is assumed that $\mathbf{s} \in \mathcal{S}_k$ is a known signal, $\mathbf{w} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$ is white Gaussian noise (WGN), and $\Phi \in \mathbf{R}^{m \times n}$ is the sampling matrix, where $\mathbf{I}$ indicates the identity matrix. By detecting with the measurement vector $\mathbf{y}$ directly using the *Neyman-Pearson* (NP) detector [67], the likelihood ratio test that makes decision on $\tilde{\mathcal{H}}_1$ and maximizes the detection rate P_D subject to the constraint of the false alarm rate $P_F \leq \alpha$ is

$$T := \frac{f_1(\mathbf{y})}{f_0(\mathbf{y})} > \zeta\tag{1.17}$$

Here, the threshold ζ is chosen such that $P_F = \int_{T > \zeta} f_0(\mathbf{y}) d\mathbf{y} = \alpha$,

$$f_0(\mathbf{y}) = \frac{e^{-\frac{1}{2}\mathbf{y}^T(\sigma^2\Phi\Phi^T)^{-1}\mathbf{y}}}{|\sigma^2\Phi\Phi^T|^{\frac{1}{2}}(2\pi)^{\frac{m}{2}}}\tag{1.18}$$

and

$$f_1(\mathbf{y}) = \frac{e^{-\frac{1}{2}(\mathbf{y}-\Phi\mathbf{s})^T(\sigma^2\Phi\Phi^T)^{-1}(\mathbf{y}-\Phi\mathbf{s})}}{|\sigma^2\Phi\Phi^T|^{\frac{1}{2}}(2\pi)^{\frac{m}{2}}}.\tag{1.19}$$

Therefore, detecting with compressed measurements directly results in the following detection rate [66]

$$P_D(\alpha) = Q(Q^{-1}(\alpha) - \|P_{\Phi^T}\mathbf{s}\|_2)\tag{1.20}$$

where $Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du$ [68] and $P_{\Phi^T} = \Phi^T(\Phi\Phi^T)^{-1}\Phi$. With the assumption that the matrix $\sqrt{n/m}P_{\Phi^T}$ satisfies the *restricted isometry property* with the constant $\eta \in (0, 1)$, it was shown in [66] that the detection rate P_D is

within the range

$$\left[Q^{-1}(\alpha) - \sqrt{\frac{m}{n}(1+\eta)\text{SNR}}, Q^{-1}(\alpha) + \sqrt{\frac{m}{n}(1-\eta)\text{SNR}} \right] \quad (1.21)$$

where $\text{SNR} = \frac{\|s\|_2^2}{\sigma^2}$. With the result shown in (1.21), it was stated in [66] that how much information lost in detection with compressed measurements rather than the signal itself is dependent on the ability to solve a detection problem but not the ability to recover the signal in CS.

1.3 Motivations and Main Contributions

Although there has been significant progress in ELM and compressed sensing, there still exist rooms for further improvements and new development, particularly in performance enhancement for ELM and signal detection in CS.

From the review of ELM, we see that almost all the past research works on ELM are based on using the sigmoid function or radial basis function as the activation function in hidden nodes. Although ELM claims that any infinitely differentiable function could be adopted as the activation function, few of the past researches have considered other functions. As we all know, activation functions play an important role in neural networks and different functions may result in different generalization performance. Moreover, using random hidden node parameters indeed increases the learning speed of the networks. But it also brings some problems, such as redundant hidden nodes, suboptimal hidden parameters, and high misclassification rate for samples nearing the classification boundary.

From the review of compressed sensing and compressed signal detection, we find

that recent works are mainly using the Neyman-Pearson (NP) method for detection with compressed measurements. What is more, the noise involved in consideration is white Gaussian. As we all know, the NP detector is more suitable for signal detection in sonar/radar applications while the Bayesian approach is more appropriate for signal detection in communications and the noise could not be always expected to be white Gaussian in general applications.

With the above motivations, we have carried out studies on improvement algorithms of extreme learning machine and on signal detection with Bayesian approach in compressed sensing which are documented in this thesis. The main contributions of this thesis are briefly summarized below.

1. In Chapter 2, we combine the extreme learning machine algorithm to train a novel structure of wavelet neural network. Instead of using random hidden node parameters, a parameter initialization method that takes into account the domain of input space has been used to initialize the translation and dilation parameters of hidden node wavelet functions.
2. In Chapter 3, we adopt the differential evolutionary algorithm to optimize network hidden node parameters. The network is then redesigned with a smaller network size than traditional ELM based tuning free neural networks as some redundant hidden nodes may exist in the neural networks proposed in Chapter 2.
3. In Chapter 4, we propose an improved algorithm named voting based extreme learning machine (V-ELM) for signal classification to further enhance the classification rate of ELM. In V-ELM, several individual ELMs with the same network structure have been trained with the same dataset and the final class

label of a test sample was determined by the majority voting method.

4. In Chapter 5, we consider the detection bound of the probability of error in compressed sensing where the signal is sparse and is reconstructed from a compressed measurement vector. Utilizing the oracle estimator in CS and the maximum likelihood detector, we provide the theoretical bound of the detection error when the noise in the estimation of CS is Gaussian noise.
5. In Chapter 6, we analyze signal detection with compressed measurements directly using the Bayesian approach. We obtain a general expression for the probability of error where the influence of prior probabilities is investigated and the additive noise is uncorrelated Gaussian noise with possibly unequal variances. The performance bounds for the probability of error using the *restricted isometry property* constant and then the mutual coherence of the sampling matrix are also presented in this chapter.

Part I

Improvements on Extreme Learning Machine

Although extreme learning machine (ELM) has shown several promising features for training single hidden layer feedforward neural networks and has presented good generalization performance, there is still a lot of room for improvements, such as the selection of hidden node activation functions and the optimization of the random assigned hidden node parameters. While ELM claims that any function which is infinitely differentiable in any interval can be adopted as the activation function in the hidden node, most of the work presented in the past several years focused on using sigmoid function or radial basis function. Moreover, though using the random computational nodes increases the learning speed dramatically in ELM, it still suffers from some issues, such as redundant hidden nodes and non-optimal hidden node parameters.

In part I of the thesis (Chapters 2, 3, and 4), we propose several improved methods based on ELM. To begin with, we adopt ELM to train wavelet neural networks (WNN), where wavelet functions are utilized as the hidden node activation function. Benefited from the excellent properties of time-frequency localization and multi-resolution signal processing of the wavelet transform and the fast learning speed of ELM, the proposed neural network with ELM has shown several salient features, such as a good generalization capability and learning faster than traditional wavelet neural networks with the gradient-based algorithm. Next, to reduce redundant hidden nodes and to optimize hidden node parameters which may exist in ELM, we introduce the differential evolutionary method into our proposed wavelet neural networks with ELM. In the third place, an ensemble method which performs multiple independent ELM training instead of a single ELM training and then makes the final decision based on the majority voting method is presented to improve the classification performance of ELM.

Chapter 2

Composite Function Wavelet Neural Networks with Extreme Learning Machine

2.1 Introduction

In this chapter, we focus on training wavelet neural network with extreme learning machine. The combination of wavelet theory and neural networks (NNs) has resulted in wavelet neural networks (WNN), which were first applied in image classification by Daugmann [69] and then popularized by the work of Pati [70] [71], Zhang [72], Szu [73] and many others. Due to the excellent properties of time-frequency localization and multi-resolution signal processing of the wavelet transform and the powerful capability of neural networks in handling large dimensional problems, wavelet neural networks have found many applications in signal processing areas during the past years. Szu et al. have applied it to speech seg-

mentation, speaker recognition and face tracking [73], [74], [75]. Kruger et al. have applied it to real environments characterization for haptic displays, forecasting and chaotic signal prediction in [76] and Subasi et al. have implemented it for electromyography (EMG) signal classification [77], etc. In general, the combination of wavelet analysis and neural networks can be classified into two categories. In the first category, the wavelet part is not included in the network learning. The wavelet coefficients of a signal decomposed on some wavelets are furnished to a neural network, and are mainly applied in signal classification problems as input features. In the second one, the wavelet function and neural networks are combined into a single method, where the wavelets are used at the hidden nodes instead of the traditional activation functions. This structure is inspired by the similarity between the wavelet decomposing theory and single hidden layer feedforward neural networks (SLFNs) as both of them show the capabilities in nonlinear system approximation and identification. In this chapter, we focus on wavelet neural networks in the second category.

Recent researches on wavelet neural networks have been mainly focused on the initialization of the wavelet dilation and translation parameters and the selection of learning algorithms. A good initialization is extremely important for a fast convergence of wavelet neural networks. Zhang and Benveniste [72] used an orthogonal least-squares procedure to initialize the coefficients and then utilized the stochastic gradient method to train the network. A clustering method to position the wavelets was applied by Echauts [78], where the distribution of points around a cluster was used to approximate the necessary dilation wavelet. Rao and Kumthekar [79] trained the network using a correlation architecture and added new wavelets at each step until convergence was reached. Yu et al. [80] first used

a large number of functions and then compacted the wavelet neural networks by using a shrinkage technique to keep only the important nodes. Oussar and Dreyfus [81] proposed a selection method to determine a set of best wavelets whose centers and dilation parameters are used as initial values and then trained the network by gradient-based techniques. Lin [82] presented a wavelet neural network with hybrid learning approach (WNN-HLA), which combined the on-line partition method and the gradient descent method to identify an internal structure and update the parameters of the proposed model. However, most of the existing wavelet neural networks are trained by the gradient-based method, which is very time-costing for large training samples and faced with many difficulties such as stopping criteria, learning rate, learning epochs and local minima, and the weight matrices are tuned iteratively to minimize the error function.

In this chapter, we propose a **composite function wavelet neural network** with **extreme learning machine** (in short, as CFWNN-ELM) [35]. In the proposed CFWNN-ELM, the input information is first processed by wavelet functions and then passed through a type of bounded nonconstant piecewise continuous activation functions $g(\cdot) : \mathbf{R} \rightarrow \mathbf{R}$. The formed network is then trained by the ELM algorithm recently proposed by Huang et al. [1, 2, 27]. In ELM, the output weights of SLFNs can be analytically determined by using Moore-Penrose (MP) generalized inverse of the hidden layer output matrix after the input weights and the hidden layer biases are chosen arbitrarily. ELM shows good generalization capability and learns faster than traditional gradient-based algorithms. To ensure that the proposed CFWNN-ELM has a good generalization property, the wavelet parameters need to be first initialized by equalizing the input information range to the wavelet function support generated by these parameters.

The rest of the chapter is organized as follows. The new structure of the proposed CFWNN is presented in Section 2.2. The new algorithm CFWNN-ELM is given in Section 2.3. The experimental evaluations are reported in Section 2.4, where the regression of some nonlinear functions and real-world data, the prediction of a chaotic signal and classifications on some real-world data sets are presented. The discussions are given in section 2.5 and the conclusions are drawn in Section 2.6.

2.2 The Network Structure of Composite Function Wavelet Neural Network

In this section, we propose a novel structure of wavelet neural networks, **composite function wavelet neural network** (in short, as CFWNN), which uses composite functions at the hidden nodes instead of traditional single wavelet function dilated and translated from the same mother wavelet.

As shown in Fig. 2.1, the structure modifications of this wavelet neural network comparing with traditional ones are on two aspects. One is the addition of a nonzero bias term x_0 in the input layer, which is seldom adopted in wavelet neural networks but is commonly used in Backpropagation (BP) networks to ensure that the network would never stop learning in case all the input values are zero. The major modification is in the hidden layer, where the composite functions are used as the activation functions, which are constituted by a type of bounded piecewise continuous activation functions $g(\cdot) : \mathbf{R} \rightarrow \mathbf{R}$ and different wavelets function generated from the same mother wavelet with different dilation values a_h

Figure 2.1: The network structure of composite function wavelet neural networks.

($h = 1, 2, \dots, H$) and translation values b_h ($h = 1, 2, \dots, H$). The resultant activation functions are more compact than those using single wavelet functions alone. This network would show good approximation ability provided that the wavelet parameters are properly assigned to cover the input space evenly. For convenience of presentation, we directly add the bias term x_0 into the input vector and the new formed k th input vector becomes $\hat{\mathbf{x}}_k = (x_0, x_{k1}, x_{k2}, \dots, x_{kI})^T$. Then, for N arbitrary training samples $(\hat{\mathbf{x}}_k, \mathbf{d}_k) \in \mathbf{R}^{(I+1)} \times \mathbf{R}^O$ ($k = 1, 2, \dots, N$) and H hidden nodes, the j th network output of $\mathbf{y}_k = (y_{k1}, y_{k2}, \dots, y_{kO})^T$ after feeding forward the training sample $\hat{\mathbf{x}}_k$ can be summed as

$$y_{kj} = \sum_{i=1}^H w_{ji} g(\psi_{a_i, b_i}(\mathbf{v}_i^T \hat{\mathbf{x}}_k)), \quad j = 1, 2, \dots, O. \quad (2.1)$$

The input weight matrix $V \in \mathbf{R}^{H \times (I+1)}$ can be described in vector form as

$$V = \begin{pmatrix} \mathbf{v}_1^T & \mathbf{v}_2^T & \dots & \mathbf{v}_H^T \end{pmatrix}^T$$

where $\mathbf{v}_i = (v_{i0}, v_{i1}, \dots, v_{iI})^T$. The output weight matrix $W \in \mathbf{R}^{H \times O}$ describe the relationship between the hidden layer and the output layer, can be written in vector form

$$W = \begin{pmatrix} \mathbf{w}_1^T & \mathbf{w}_2^T & \dots & \mathbf{w}_O^T \end{pmatrix}^T$$

where $\mathbf{w}_i = (w_{j1}, w_{j2}, \dots, w_{jH})^T$, ($j = 1, 2, \dots, O$) and $\mathbf{d}_k = (d_{k1}, d_{k2}, \dots, d_{kO})^T$ ($k = 1, 2, \dots, N$) is the O dimensional corresponding desired output vector. The functions ψ_{a_i, b_i} , called wavelets, are dilated and translated versions of a single

mother wavelet function ψ

$$\psi_{a_i, b_i}(x) = |a_i|^{-1/2} \psi\left(\frac{x - b_i}{a_i}\right), \quad i = 1, 2, \dots, H, \quad (2.2)$$

where a_i and b_i are the corresponding dilation and translation parameters respectively.

2.3 Learning Algorithm: CFWNN-ELM

In this section, the parameter initialization and the composite function wavelet neural network learning with ELM (in short, as CFWNN-ELM) will be presented.

From (2.1) and the structure depicted in Fig. 2.1, the proposed wavelet neural network formed a single-hidden layer feedforward network and can be expressed as a system model similar to (1.3). Therefore, it is shown next that the ELM algorithm can be adopted to train the proposed wavelet neural network. For the desired output

$$\mathbf{D} = \begin{pmatrix} \mathbf{d}_1^T & \mathbf{d}_2^T & \cdots & \mathbf{d}_N^T \end{pmatrix}_{N \times O}^T, \quad (2.3)$$

there exist $\mathbf{v}_i$, $\mathbf{w}_i$, a_i and b_i such that for H hidden nodes, $\sum_{k=1}^N \|\mathbf{d}_k - \mathbf{y}_k\| = 0$. Therefore, the wavelet neural network described in (2.1) can be written as the following system model

$$\mathbf{H}_w \mathbf{W} = \mathbf{D} \quad (2.4)$$

where

$$\mathbf{H}_w = \begin{bmatrix} g(\psi_{a_1, b_1}(\mathbf{v}_1^T \hat{\mathbf{x}}_1)) & \cdots & g(\psi_{a_H, b_H}(\mathbf{v}_H^T \hat{\mathbf{x}}_1)) \\ \vdots & \ddots & \vdots \\ g(\psi_{a_1, b_1}(\mathbf{v}_1^T \hat{\mathbf{x}}_N)) & \cdots & g(\psi_{a_H, b_H}(\mathbf{v}_H^T \hat{\mathbf{x}}_N)) \end{bmatrix} \quad (2.5)$$

$$\mathbf{W} = \begin{pmatrix} \mathbf{w}_1^T & \mathbf{w}_2^T & \cdots & \mathbf{w}_O^T \end{pmatrix}^T. \quad (2.6)$$

Since both the function $g(\cdot)$ and the wavelet functions are infinitely differentiable, composite functions constituted by these two functions are also infinitely differentiable. Hence, after random initializations of the network parameters, the ELM algorithm can be applied here as the learning algorithm for the proposed wavelet neural network. Consequently, the network output weights can be analytically determined by solving a least-square solution as

$$\hat{\mathbf{W}} = \mathbf{H}_w^\dagger \mathbf{D}, \quad (2.7)$$

where $\mathbf{H}_w^\dagger$ is the Moore-Penrose generalized inverse of the hidden layer output matrix $\mathbf{H}_w$.

2.3.1 Parameter Initialization

As initialization of the hidden node wavelet dilation and translation parameters is an important issue in wavelet neural network learning algorithm, a parameter initialization method (see, e.g. [83]) that takes the input information into account for the dilation and translation values of the wavelet transform is adopted in the proposed CFWNN-ELM and described in the following.

First, randomly assign the input weight matrix $\mathbf{V} = (v_{ij})_{H \times (I+1)}$. Let $[x_{i,\min}, x_{i,\max}]$

represent the domain containing the input item x_i in all the observed samples (the bias term x_0 is not taken into account here), and t^* , ψ^* denote the time-domain center and radius of the mother wavelet ψ , respectively. Then, the domain of the generated h -th hidden node wavelet function ψ_{a_h, b_h} is given as

$$[b_h + a_h(t^* - \psi^*), b_h + a_h(t^* + \psi^*)].$$

The input information range of the h -th hidden layer can be calculated as

$$\left[v_{h0}x_0 + \sum_{i=1}^I v_{hi}x_{i,\min}, v_{h0}x_0 + \sum_{i=1}^I v_{hi}x_{i,\max} \right],$$

where $h = 1, 2, \dots, H$. To ensure the hidden node wavelet function ψ_{a_h, b_h} can cover the input information domain, we make the following equations to hold

$$b_h + a_h(t^* - \psi^*) = v_{h0}x_0 + \sum_{i=1}^I v_{hi}x_{i,\min} \quad (2.8)$$

$$b_h + a_h(t^* + \psi^*) = v_{h0}x_0 + \sum_{i=1}^I v_{hi}x_{i,\max} \quad (2.9)$$

Since the input weights are already assigned randomly, based on equations (2.8) and (2.9), the dilation parameter a_h and translation parameter b_h in the h -th hidden layer can be derived as

$$a_h = \frac{1}{2\psi^*} \left[\sum_{i=1}^I v_{hi}(x_{i,\max} - x_{i,\min}) \right] \quad (2.10)$$

$$b_h = v_{h0}x_0 + \frac{1}{2\psi^*} \left[\sum_{i=1}^I v_{hi}x_{i,\max}(\psi^* - t^*) + \sum_{i=1}^I v_{hi}x_{i,\min}(\psi^* + t^*) \right] \quad (2.11)$$

where $h = 1, 2, \dots, H$, the time-domain center and radius are calculated by the following formulae

$$t^* = \frac{1}{\|\psi\|_2^2} \int t |\psi(t)|^2 dt \quad (2.12)$$

$$\psi^* = \frac{1}{\|\psi\|_2} \left\{ \int (t - t^*)^2 |\psi(t)|^2 dt \right\}^{\frac{1}{2}} \quad (2.13)$$

where $\|\cdot\|$ denotes the norm in nonconstant space. This finishes the initialization of the network parameters.

Since the input information is taken into account, CFWNN-ELM with the above parameter initialization procedure will lead to a better generalization performance than ELM, whose parameters are randomly assigned within the region $[0, 1]$.

2.3.2 CFWNN-ELM

The training algorithm of CFWNN can be summarized as follows:

Algorithm CFWNN-ELM:

Given a training set $\aleph = \{(\mathbf{x}_i, \mathbf{d}_i) | \mathbf{x}_i \in \mathbf{R}^I, \mathbf{d}_i \in \mathbf{R}^O, i = 1, 2, \dots, N\}$, the hidden node composite function $g(\psi(\mathbf{x}; \mathbf{a}, \mathbf{b}))$ and hidden node number H :

1. Randomly assign input weight matrix V and initialize the hidden node parameters $(a_h, b_h), h = 1, 2, \dots, H$ according to (2.10) and (2.11);
2. Calculate the hidden layer output matrix H_w in (2.5);
3. Calculate the output weight $\hat{W}$ in (2.7).

Remark 2.1. *The input weight matrix V used in this algorithm is randomly assigned once and is never changed again contrasting to some conventional algorithms*

that tune the weight matrix iteratively during the network learning, such as the popular BP algorithm.

Remark 2.2. In the proposed CFWNN-ELM, the hidden node parameters a_h and b_h are calculated from (2.10) and (2.11) after randomly initializing the input weight matrix V . The information of the input range $[x_{i,\min}, x_{i,\max}]$ is also taken into account for parameters initialization. This initialization method shows good generalization property and is different from ELM algorithm, in which the hidden node parameters are randomly assigned without considering the information of the training data.

2.4 Performance Evaluations

In this section, the performance of the proposed CFWNN-ELM is compared with support vector machines (SVMs) and three relevant neural networks, namely, SLFN-ELM, WNN-ELM and CFWNN-BP, which are briefly described below, on several problems in model regression, chaotic signal prediction and classification areas.

- SLFN-ELM: **S**ingle **H**idden **L**ayer **F**eedforward **N**etworks training with the ELM algorithm;
- WNN-ELM: **W**avelet **N**eural **N**etwork using wavelets at the hidden nodes and training with the ELM algorithm;
- CFWNN-BP: **C**omposite **F**unction **W**avelet **N**eural **N**etwork training with the BackPropagation algorithm.

All these simulations for these four NNs are carried out in MATLAB 7.4 environment running on a 2.66 GHZ CPU. The simulations for SVM are carried out using compiled C-code SVM package: Libsvm [84] running in the same PC. The kernel function used in SVM is radial basis function whereas for CFWNN-ELM and CFWNN-BP, the popular Morlet wavelet function [85]

$$\psi(t) = \cos(1.75t)e^{(-0.5t^2)} \quad (2.14)$$

is used as the mother wavelet and the sigmoid function $g(x) = 1/(1 + \exp(-x))$ is adopted to form the composite function used at the hidden nodes. For WNN-ELM, the same Morlet function is used as the mother wavelet. For WNN-ELM and CFWNN-BP, the dilation and translation parameters are initialized using the same method as that in CFWNN-ELM. The cost parameter C and the kernel parameter γ are searched in a grid as proposed in [82] for SVM and the best parameters are then obtained in terms of the generalized performance. Some of the experiments are compared with existing results obtained by various popular algorithms, such as WNN-HLA [86], RBF [87, 88], WNN [89], SAOCIF [90], Cascade-Correlation algorithm [90], bagging and boosting methods [91] and decision tree algorithm C4.5 [91].

2.4.1 Evaluation on Regression Problems

Simulations on the approximations of three nonlinear functions and the regressions of some real-world datasets are given in following subsections.

A. Evaluation on function approximations

In this example, the four neural networks (CFWNN-ELM, SLFN-ELM, WNN-

ELM and CFWNN-BP) and support vector machine for regression (SVR) are evaluated on three nonlinear function approximation problems, whose mathematical representations are described as follows:

$$f_1 = \frac{\sin(x)}{x}, \quad (2.15)$$

$$f_2 = \begin{cases} 43.72x - 8.996, & x \in [0, 0.4] \\ -84.62x + 42.46, & x \in (0.4, 0.5] \\ 10e^{-x} \sin(12x^2 + 2x - 4), & x \in (0.5, 1] \end{cases} \quad (2.16)$$

$$f_3 = (x_1^2 - x_2^2) \sin(0.5x_1), \quad x_1, x_2 \in [-10, 10] \quad (2.17)$$

Function f_1 is the famous ‘SinC’ function and the last two are an one dimensional piecewise function and a two dimensional nonlinear function, respectively. For function f_1 , a training set $(x_i, f_1(x_i))$ and a testing set $(x_i, f_1(x_i))$ with 5000 points, respectively, are generated where x_i is uniformly distributed on the input region $[-10, 10]$ for all these five methods.

Similar to function f_1 , a training set $(x_i, f_2(x_i))$ and a testing set $(x_i, f_2(x_i))$ with 2000 points, respectively, are generated where x_i is uniformly distributed on the input region $[0, 1]$ in this simulation for function f_2 .

For function f_3 , a training dataset $((x_{1i}, x_{2i}), f_3(i))$ and a testing dataset $((x_{1i}, x_{2i}), f_3(i))$ are collected where (x_{1i}, x_{2i}) are uniformly randomly distributed over the input domain $D = [-10, 10] \times [-10, 10]$, which forms a 60×60 grid.

To make these approximation problems ‘real’, uniformly distributed noise in $[-0.2, 0.2]$ has been added to all the training samples of these three functions, while the validating and testing data remain noise-free. Table 2.1 shows the performance comparisons using these five methods on the three function approximations. All

these results are collected by averaging 50 trials except CFWNN-BP, whose results are collected by only 10 trials due to its long training time. Abbreviations adopted in some of the subsequent tables are explained below.

- **Testing RMSE** (Testing Root Mean Square Error) denotes the average testing root mean square error of multiple trials.
- **Testing Dev** denotes the standard derivation of the average testing root mean square error of multiple trials.
- **Training Time** denotes the average training time of multiple trials.
- **Ite** denotes the amount iterations taken by the neural networks.
- **SVs** denotes the average number of support vectors used for SVM.

Table 2.1: Results comparisons for function approximations.

Func.	NNs	Testing		Training time (s)	Nodes / SVs	Ite / (C, γ)
		RMSE	Dev			
f_1	CFWNN-ELM	0.0062	0.0012	0.0484	30	/
	SLFN-ELM	0.0086	0.0012	0.0293	30	/
	WNN-ELM	0.0651	0.0309	0.0309	30	/
	CFWNN-BP	0.0489	0.0144	2688.7	70	8000
	SVR	0.0099	0.0016	1534.3	2504.9	($2^{-4}, 2^{-4}$)
f_2	CFWNN-ELM	0.2193	0.0224	0.0280	30	/
	SLFN-ELM	0.9863	0.4156	0.0093	30	/
	WNN-ELM	0.5965	0.0697	0.0254	30	/
	CFWNN-BP	1.4433	0.1066	1626	90	8000
	SVR	0.3440	0.0161	236.96	1260.7	($2^5, 2^5$)
f_3	CFWNN-ELM	0.2417	0.0687	0.4673	200	/
	SLFN-ELM	0.3729	0.1271	0.3953	200	/
	WNN-ELM	0.7016	0.1830	0.4200	200	/
	CFWNN-BP	34.431	2.9702	9594.8	200	7000
	SVR	0.3063	0.1253	1501.3	1994.7	($2^4, 2^{-3}$)

From Table 2.1, one can easily find that the proposed CFWNN-ELM is able to achieve the lowest testing RMSEs 0.0062, 0.2193 and 0.2417 corresponding to these

three nonlinear functions among all five methods. Moreover, CFWNN-ELM keeps the fast learning speed feature of ELM and the training time costed by CFWNN-ELM is thousand times smaller than conventional BP algorithm (CFWNN-BP) and SVR. The comparisons between CFWNN-ELM and WNN-ELM manifests that composite functions used at the hidden nodes can perform better than using wavelet functions alone.

Fig. 2.2 depicts the simulation curves using these four methods (CFWNN-

Figure 2.2: The approximation curves of the ‘SinC’ function using 500 samples by CFWNN-ELM, SLFN-ELM, CFWNN-BP and SVR.

ELM, SLFN-ELM, CFWNN-BP and SVR) and Fig. 2.3 shows the relationship between the generalization performance of CFWNN-ELM and its network size for

Figure 2.3: The generalization performance of CFWNN-ELM is stable on a wide range of number of hidden nodes.

the ‘SinC’ function approximation using 500 training samples. From Fig. 2.3, the generalization performance of CFWNN-ELM is stable on a wide range of number of hidden neurons.

In [72], a figure named the performance index J is introduced to assess the approximation results, which is defined in the following

$$J = \sqrt{\frac{\sum_{i=1}^I (y_i - y_i^d)^2}{\sum_{i=1}^I (y_i^d - \bar{y})^2}}, \quad \bar{y} = \frac{1}{I} \sum_{i=1}^I y_i^d, \quad (2.18)$$

where y_i^d is the i th desired output and y_i is the network approximated output. From the definition, it is apparent that the performance index is the same as the RMSE as the testing data has a constant standard derivation. For convenience of comparisons, the performance of the proposed CFWNN-ELM is compared with several related methods in terms of the performance index in the following.

Table 2.2: Performance index comparisons for learning 2-D nonlinear function f_3 .

NNs	CFWNN-ELM	WNN-HLA [86]	RBF [87]	WNN [89]	WN [72]
J	0.0087	0.019	0.05	0.03	0.033

Table 2.2 shows the comparison of the performance index J with some traditional existing methods, such as wavelet neural networks learning by a hybrid gradient descent method [86], wavelet neural networks learning by a stochastic gradient algorithm [72], the RBF networks [87], etc. For comparison, a dataset of 400 (20×20) training points is collected from the random inputs that are uniformly distributed over the domain $D = [-10, 10] \times [-10, 10]$, where these training samples are noise-free. For this simulation, a four-layer structure of wavelet neural network including an input layer, a wavelet layer, a product layer and an output layer was designed in [86]. It can be seen that the performance index of CFWNN-

ELM is the lowest among all NNs and it only takes 0.0447s for CFWNN-ELM to reach the lowest performance index value 0.0087.

B. Evaluation on real-world regression data

The performance of CFWNN-ELM is compared with other four methods on five real-world function approximations from the UCI database [92]. The specifications of these five datasets are shown in Table 2.3. In the simulations, all the attributes (inputs) have been normalized to the range of $[-1, 1]$ while the outputs have been normalized into $[0, 1]$.

For each problem, the training data and testing data are randomly generated

Table 2.3: Specification of datasets.

Datasets	# Training Data	# Testing Data	#Attributes	
			Continuous	Nominal
Abalone	2000	2177	7	1
Autompg	320	78	8	0
BosHousing	250	256	13	0
CalHousing	8000	12640	8	0
Auto Price	80	79	14	1

from its whole dataset in each trial. All the results are collected by taking the average of multiple trails. For the four NNs (CFWNN-ELM, SLFN-ELM, WNN-ELM and CFWNN-BP), the number of hidden nodes is gradually increased by 5 and the optimal number of hidden nodes is chosen based on the cross-validation method to ensure that the cost function is minimized. For SVR, the generalized accuracy is estimated using different combinations of the cost parameter C and the kernel parameter γ searched in the grid formed by $C = [2^{12}, 2^{11}, \dots, 2^{-2}]$ and $\gamma = [2^4, 2^3, \dots, 2^{-10}]$. Therefore, 225 combinations of (C, γ) are tested for SVR and the best combination with the lowest testing RMSE is then selected. Table 2.4 shows the average results of 50 trials of simulations for all these five methods.

Table 2.4: Results comparisons for regressions of real-world datasets.

Datasets	NNs	Testing		Nodes / SVs	Training time (s)	Ite / (C, γ)
		RMSE	Dev			
Abalone	CFWNN-ELM	0.0775	0.0016	50	0.0450	/
	SLFN-ELM	0.0824	0.0028	25	0.0153	/
	WNN-ELM	0.2314	0.6496	45	0.0372	/
	CFWNN-BP	0.0934	0.0017	35	472.5219	2000
	SVR	0.0821	0.0011	332.22	290.21	($2^0, 2^{-3}$)
Autompg	CFWNN-ELM	0.0743	0.0115	50	0.0084	/
	SLFN-ELM	0.0758	0.0090	50	0.0063	/
	WNN-ELM	0.1987	0.8202	50	0.0091	/
	CFWNN-BP	0.1010	0.0070	50	65.7891	2000
	SVR	0.0987	0.0026	80.54	22.3369	($2^{-2}, 2^{-5}$)
BosHousing	CFWNN-ELM	0.1067	0.0114	60	0.0109	/
	SLFN-ELM	0.1016	0.0096	50	0.0063	/
	WNN-ELM	0.7147	4.0149	20	0.0034	/
	CFWNN-BP	0.1495	0.0044	50	124.6063	2500
	SVR	0.1102	0.0069	57.24	13.3994	($2^1, 2^{-6}$)
CalHousing	CFWNN-ELM	0.1290	0.0018	50	0.3318	/
	SLFN-ELM	0.1332	0.0030	50	0.1044	/
	WNN-ELM	2.3861	6.0218	50	0.1725	/
	CFWNN-BP	0.1441	0.0043	50	2549.5	2500
	SVR	0.1193	0.0022	2176.2	1836.7	($2^3, 2^1$)
Auto Price	CFWNN-ELM	0.0912	0.0209	30	0.0001	/
	SLFN-ELM	0.0990	0.0119	15	$< 10^{-4}$	/
	WNN-ELM	0.1295	0.0480	30	0.0003	/
	CFWNN-BP	0.1163	0.0378	15	9.9219	2000
	SVR	0.0945	0.0098	22.75	5.2988	($2^8, 2^{-5}$)

As shown in this table, CFWNN-ELM wins the lowest testing RMSEs in three cases among all these methods and the testing RMSEs of the other two datasets obtained by CFWNN-ELM are slightly larger than the best one. CFWNN-ELM is more compact than SVR in most cases as the number of hidden nodes needed for CFWNN-ELM is smaller than the support vectors required for SVR in general. The training time costed by CFWNN-ELM is slightly longer than the one of SLFN-ELM, however, its dramatically shorter than the training time used by CFWNN-BP and SVR for all datasets.

2.4.2 Evaluation on Classification using Real-world Datasets

In this section, the proposed CFWNN-ELM is evaluated with 21 real world datasets and the performance is compared with SLFN-ELM, WNN-ELM, CFWNN-BP, and SVM. The first 20 datasets are download from the UCI¹ database while the last protein sequence classification dataset is from the Protein Information Resource (PIR²) center. Table 2.5 shows the specifications of all these 21 datasets. In the experiments, the training and testing data of the first 15 datasets in Table 2.5 are reshuffled at each trial of simulation while the training and testing data of the last 6 datasets are fixed for all trials of simulations. The performance of CFWNN-ELM on the Diabetes dataset is also compared with some existing popular algorithms.

For SVM, the best cost parameter C and kernel parameter γ are searched in the grid form by $C = [2^{12}, 2^{11}, \dots, 2^{-2}]$ and $\gamma = [2^4, 2^3, \dots, 2^{-10}]$ in terms of the success testing rate. Therefore, 225 (15×15) combinations of parameters (C, γ) are tried for each problem for SVM. Average results using 50 trials with each combination of (C, γ) are obtained and the best success testing rates are collected. Fifty trials of simulations have been conducted for CFWNN-ELM, SLFN-ELM, WNN-ELM and SVM, and 10 trials have been conducted for CFWNN-BP due to its long training time. The classification results including the successful testing rate, the training time and the network complexities on these three datasets using all these five methods are shown in Tables 2.6 and 2.7, respectively. Abbreviations adopted in this table are explained below

- **Success Testing Rate** represents the average successful testing rate of multiple trails;

¹<http://www.ics.uci.edu/~mllearn/MLRepository.html>

²<http://pir.georgetown.edu/>

Table 2.5: Specifications of classification datasets

Datasets	# Attributes	# Classes	# Training data	#Testing data
Balance	4	3	400	225
Breast-can	30	2	300	269
Diabetes	8	2	576	192
Ecoli	7	8	168	168
Glass	10	2	100	114
Heart	13	2	100	170
Iris	4	3	100	50
Liver	6	2	200	145
Satimage	36	7	4400	2200
Segment	18	7	1500	810
Sonar	60	2	100	108
Soybean	35	4	22	21
Spambase	57	2	2300	2301
Waveform	21	3	3000	2000
Wine	13	3	100	78
Digit	16	10	7494	3498
Hayes	4	3	132	28
Monk1	6	2	124	432
Monk2	6	2	169	432
Monk3	6	2	122	432
Protein	56	10	949	533

- **Dev** represents the standard derivation of successful testing rate.

From table 2.6, one can easily find that CFWNN-ELM and SVM are the best two methods for classifications of these datasets. CFWNN-ELM wins highest success testing rate in 8 datasets and SVM reaches the highest success testing rates in 11 datasets, while the SLFN-ELM wins in the rest 2 datasets. For the Digit dataset, the gradient descent based algorithm CFWNN-BP has run out of memory in our ordinary PC. Therefore, it is a problem for the traditional gradient descent algorithm for large size dataset applications in an ordinary PC. From table 2.7, one can find that the training time costed by SVM and the gradient descent based

Table 2.6: Results comparisons for classification on datasets.

Datasets	CFWNN-ELM		SLFN-ELM		WNN-ELM		CFWNN-BP		SVM	
	Testing (%)		Testing (%)		Testing (%)		Testing (%)		Testing (%)	
	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev
Balance	93.24	1.63	90.53	1.79	90.11	2.45	89.57	2.67	95.88	1.31
Breast-can	96.21	0.80	95.52	1.38	94.65	1.23	93.86	1.14	95.55	0.82
Diabetes	78.02	2.56	77.57	2.76	76.60	2.99	73.85	2.69	77.31	2.73
Ecoli	86.20	2.05	85.56	2.31	84.68	2.87	85.22	2.56	85.83	2.79
Glass	92.34	2.24	91.91	2.54	91.56	2.86	90.42	2.98	91.84	2.78
Heart	76.85	3.43	75.87	3.17	75.24	3.47	71.75	6.67	76.10	3.46
Iris	95.92	3.05	94.87	2.97	94.42	3.57	92.40	3.19	94.36	2.76
Liver	69.45	3.44	70.01	3.85	68.55	3.76	67.22	4.56	68.24	4.58
Satimage	88.82	0.48	89.04	0.4	88.53	0.52	83.35	0.59	91.25	0.28
Segment	95.40	0.80	95.01	0.74	95.16	0.72	82.12	2.65	96.93	0.62
Sonar	76.45	2.38	73.31	4.29	74.56	3.78	73.01	4.76	83.48	3.88
Soybean	98.46	2.54	99.00	2.21	97.24	3.56	97.55	2.66	99.56	1.32
Spambase	92.35	0.56	90.50	0.68	90.24	0.78	90.16	0.65	93.50	0.45
Waveform	85.90	0.55	85.50	0.83	84.33	0.98	84.02	1.21	85.78	0.62
Wine	97.66	1.42	96.79	1.66	96.22	1.78	96.01	1.89	97.48	1.57
Digit	96.56	0.44	97.01	0.23	96.22	0.55	\	\	98.14	0.01
Hayes	74.95	3.56	75.86	4.76	74.23	4.88	74.86	4.75	75	0
Monk1	84.65	2.24	82.78	2.34	81.55	3.02	82.44	2.56	94.44	0.01
Monk2	82.44	1.68	79.27	2.11	78.22	2.48	78.66	2.56	84.72	0.01
Monk3	87.90	2.45	87.38	2.83	86.44	3.12	85.78	3.56	90.04	0.01
Protein	88.12	1.22	87.90	1.34	87.02	1.56	84.86	1.87	90.63	0.01

algorithm is thousand times longer than the one used by CFWNN-ELM. Moreover, the number of support vectors required by SVM is generally more than the number of hidden nodes needed by CFWNN-ELM.

Some relevant results comparing with many existing methods on the data classification of the real medical diagnosis problem (Diabetes) are listed in Table 2.8. It is easy to find that CFWNN-ELM can reach the average testing success rate 78.02%, higher than all the results using various popular algorithms listed in the table, such as SAOCIF [90], Cascade-Correlation algorithm [90], bagging and boosting methods [91], decision tree algorithm C4.5 [91] and RBF [88].

Table 2.7: Comparisons of network complexities.

Datasets	CFWNN-ELM		SLFN-ELM		WNN-ELM		CFWNN-BP		SVM		
	Training Time	Nodes	Training Time	Nodes	Training Time	Nodes	Training Time	Nodes	Training Time	SVs	(C, γ)
Balance	0.0098	35	0.0088	35	0.0090	30	20.1145	20	23.1122	47.18	$(2^{12}, 2^{-3})$
Breast-can	0.0086	40	0.0053	45	0.0076	40	25.2258	30	18.0719	56.22	$(2^1, 2^{-3})$
Diabetes	0.0102	35	0.0009	20	0.0153	50	145.6531	30	364.6	62.28	$(2^{-1}, 2^{-1})$
Ecoli	0.0026	20	0.0003	20	0.0020	20	3.2215	20	10.942	67.86	$(2^{10}, 2^{-7})$
Glass	0.0018	30	0.0006	26	0.0014	30	1.9865	20	4.1444	47.88	$(2^1, 2^2)$
Heart	0.0016	30	0.0006	16	0.0014	30	12.3066	30	6.7094	81.6	$(2^4, 2^{-1})$
Iris	0.0009	20	0.0001	20	0.0004	20	8.3641	20	4.5488	23.3	$(2^8, 2^0)$
Liver	0.0044	30	0.0022	22	0.0038	30	3.4478	20	5.0789	156.2	$(2^3, 2^1)$
Satimage	2.8228	500	2.6556	500	2.8391	500	2823.7	300	681.77	1126.8	$(2^0, 2^{-1})$
Segment	0.5156	300	0.1953	200	0.5159	100	1916.4	200	452.9094	379.4	$(2^4, 2^0)$
Sonar	0.0086	50	0.0044	56	0.0060	50	3.5587	20	8.9113	69.7	$(2^8, 2^{-1})$
Soybean	0.0012	90	0.0008	90	0.0010	90	1.2253	70	4.2516	21.20	$(2^7, 2^{-5})$
Spambase	0.7654	200	0.3322	240	0.6657	200	827.5235	100	913.76	452.6	$(2^{11}, 2^{-6})$
Waveform	0.5644	100	0.1266	100	0.2489	100	542.4467	60	1253.3	1863.6	$(2^4, 2^{-10})$
Wine	0.0076	20	0.0006	24	0.0054	20	1.2687	20	5.8941	47.3	$(2^2, 2^{-1})$
Digit	1.4627	200	0.8741	200	1.2577	200	\	\	5894.8	1133	$(2^3, 2^{-1})$
Hayes	0.0058	50	0.0022	46	0.0042	50	2.2265	40	8.1366	82	$(2^8, 2^{-6})$
Monk1	0.0088	50	0.0031	45	0.0065	45	2.1489	40	6.6609	51	$(2^{12}, 2^{-3})$
Monk2	0.0553	100	0.0219	115	0.0463	100	4.7765	80	11.4766	101	$(2^{12}, 2^{-1})$
Monk3	0.0205	60	0.0075	70	0.188	60	2.4875	40	6.4887	55	$(2^8, 2^{-2})$
Protein	1.7655	120	0.9968	125	1.4438	120	420.5578	40	274.9672	261	$(2^7, 2^4)$

Table 2.8: Performance comparison with some traditional algorithms in real medical diagnosis application: Diabetes.

Algorithms	Testing Success Rate (%)
CFWNN-ELM	78.02
SAOCIF [90]	77.32
Cascade-Correlation [90]	76.58
AdaBoost [91]	75.60
C4.5 [91]	71.60
RBF [88]	76.30
Heterogeneous RBF [88]	76.30

2.5 Discussions

As evident from many simulation results reported in the previous section, CFWNN-ELM has shown several interesting and unique features that are different from some relevant feedforward neural networks.

The learning speed of CFWNN-ELM is much faster than CFWNN-BP, whose structure is the same as CFWNN-ELM but is trained with the conventional BP algorithm. By adopting the ELM algorithm for training, the learning of CFWNN-ELM can be completed in seconds or less than a second in most of the evaluation experiments. On the other hand, the training time for CFWNN-BP can be as high as hundreds or thousands of seconds, even for rather simple function approximation problems, such as the approximation of the two-dimensional nonlinear function.

Since the performances of CFWNN-ELM are much better than those of WNN-ELM and CFWNN-BP in general, we focus on the comparison between CFWNN-ELM and SLFN-ELM in the following.

As it can be seen from the experimental results, although the training time for CFWNN-ELM is slightly longer than the one for SLFN-ELM, the performances of CFWNN-ELM are better than those of SLFN-ELM in terms of regression, prediction and classification accuracy in most cases. We attribute the outstanding

performances of CFWNN-ELM to the introduction of composite functions as activation functions and the initialization of hidden node parameters by taking into account the information of the input space domains. This can be demonstrated through the analysis of the approximation of the ‘SinC’ function using CFWNN-ELM and SLFN-ELM as discussed below.

For both neural networks, the number of hidden nodes is set to be 10, and 20 samples are generated as the training data according to two ways to show the initialization differences between the method used in our algorithm and the random initialization method used in ELM algorithm by single realization. One is uniformly generated in two separate regions where 10 samples are collected from the region $[-10, -6]$ and the other 10 samples are from the region $[6, 10]$ separately. The other way is uniformly generated these 20 samples from one region $[-5, 5]$.

Fig. 2.4 and Fig. 2.5 depicted the distributions of the hidden node functions of SLFN-ELM and CFWNN-ELM under these two training sample generating ways, respectively. For SLFN-ELM, the parameters of the hidden node activation functions are randomly generated from the region $[0, 1]$ according to uniform distribution, for CFWNN-ELM, the translation and dilation parameters of the wavelet functions are calculated using equations (2.10) and (2.11). It is easy to find from these two figures that the distribution of the hidden node activation functions of CFWNN-ELM is consistent with the distribution of the input samples. As shown in Fig. 2.4, when training samples are collected from two separate regions, the activation functions of CFWNN-ELM are also attracted to two regions correspondingly. When training samples are generated from one region, the activation functions are gathering around these samples as shown in Fig. 2.5.

Figure 2.4: Distribution of the input samples and the hidden nodes from two separate regions.

Figure 2.5: Distribution of the input samples and the hidden nodes from one region.

2.6 Conclusions

Composite function wavelet neural networks utilizing the composite functions as the activation functions and the ELM algorithm as the learning algorithm have been first presented in this chapter. A parameter initialization method that takes the input information into account has been adopted to initialize the dilation and translation values of the wavelet transform to ensure the hidden node activation functions can cover the input data properly. In most applications at hand, CFWNN-ELM has shown a good generalization capability and it learns much faster than traditional wavelet neural networks using gradient-based algorithms. The experimental results on several artificial and real function approximations, signal prediction and classification problems have validated that the proposed CFWNN-ELM is efficient and promising.

Chapter 3

Composite Function Wavelet Neural Networks with Differential Evolution and Extreme Learning Machine

3.1 Introduction

The composite function wavelet neural network with extreme learning machine (CFWNN-ELM) proposed in Chapter 2 has shown several salient features. It can not only keep the fast learning speed feature of ELM but also has better generalization performances in model regressions and signal classifications than many relevant conventional methods, such as WNN training with BP and ELM, SLFN with ELM (SLFN-ELM).

However, a lot of redundant hidden nodes may exist and the network parameters

may not be optimal in ELM because the number of hidden neurons are assigned a priori and the input weight matrix and bias vector are randomly generated once and never changed during the learning phase. During the past few years, evolutionary algorithms (EA) are widely used heuristic methods for function optimizations because they had many excellent features, such as easy implementation, and performing well for nearly all types of problems. Differential evolution (DE), a variant of evolutionary computing techniques proposed by Storn [93], has been demonstrated that it can achieve more accurate performance than most evolutionary algorithms for optimizations. Due to this reason, DE is always used as a tool for searching the optimal parameters in machine learning fields. Zhu *et al.* [94] proposed a hybrid approach that combines both DE and ELM for SLFN (SLFN-E-ELM) and Subudhi [95] considered another combination method that takes advantages of both DE and Levenberg-Marquardt algorithm (DE-LM). It has shown that adopting DE in training SLFN is able to achieve better generalization performance with much more compact networks than many relevant methods. Moreover, Zhu *et al.* has shown in [94] that SLFN-E-ELM not only learns much faster but also achieves much better performances than DE-LM in model regressions and signal classifications.

Although CFWNN-ELM performs better than SLFN-ELM in many applications, it is also limited by the same problems encountered with SLFN-ELM mentioned above, that is, some redundant nodes may exist in CFWNN-ELM and the hidden node learning parameters may not be optimal as the number of hidden nodes are assigned a priori and the hidden node learning parameters are randomly generated once and they never changed during the learning phase. With this motivation, a composite function wavelet neural network learning with differential

evolution and **extreme learning machine** (in short, as CWN-E-ELM) is presented in this chapter [37]. In CWN-E-ELM, a parameter initialization method that takes the input information range and the generated wavelet function support into account is first utilized to initialize the dilation and translation parameters of the wavelet functions. The DE algorithm is then adopted to select the nearly optimal input weights and wavelet dilation and translation values and the ELM algorithm is applied to train the network and determine the output weights. Consequently, the network is redesigned with a much more compact network size than conventional ELM-based tuning free neural networks.

The rest of this chapter is organized as follows. Section 3.2 gives a brief review of the differential evolution algorithm. In Section 3.3, we present the detailed algorithm CWN-E-ELM. Experimental evaluations on the simulations of several regression and classification problems and comparisons with results of some relevant methods are given in Section 3.4. Discussions and conclusions are drawn in Section 3.5.

3.2 Differential Evolution

A new heuristic approach for minimizing possibly nonlinear and non-differentiable continuous space functions named differential evolution (DE) is proposed in [93], which is known as one of the most efficient evolutionary algorithms (EAs). In general, the strategy of DE can be described in the following steps

1. Initialization

Randomly generate a set of D –dimensional parameter vectors

$$\{\Theta_{i,G} | i = 1, 2, \dots, NP\} \quad (3.1)$$

as a population at each generation G , where the initial vector populations should cover the entire parameter space and NP dose not change during the minimization process.

2. Mutation

For each target vector $\Theta_{i,G}, i = 1, 2, \dots, NP$, a mutant vector is created according to

$$\mathbf{v}_{i,G+1} = \Theta_{r_1,G} + F \cdot (\Theta_{r_2,G} - \Theta_{r_3,G}) \quad (3.2)$$

with random and mutually different indices $r_1, r_2, r_3 \in \{1, 2, \dots, NP\}$ and an amplification constant factor $F \in [0, 2]$, which controls the amplification of the differential variation $(\Theta_{r_2,G} - \Theta_{r_3,G})$.

3. Crossover

The crossover is introduced to increase the diversity of the perturbed parameter vectors. In this step, the D –dimensional trial vector

$$\boldsymbol{\mu}_{i,G+1} = (\boldsymbol{\mu}_{1i,G+1}, \boldsymbol{\mu}_{2i,G+1}, \dots, \boldsymbol{\mu}_{Di,G+1}) \quad (3.3)$$

is formed, where

$$\boldsymbol{\mu}_{ji,G+1} = \begin{cases} \boldsymbol{v}_{ji,G+1} & \text{if } (\text{randb}(j) \leq \text{CR}) \\ & \text{or } j = \text{rnbr}(i) \\ \boldsymbol{\Theta}_{ji,G} & \text{if } (\text{randb}(j) > \text{CR}) \\ & \text{and } j \neq \text{rnbr}(i) \end{cases} \quad (3.4)$$

$j = 1, 2, \dots, D$ and $i = 1, 2, \dots, NP$. In the above equation, $\text{randb}(j)$ is the j th evaluation of an uniform random number generator with outcome $\in [0, 1]$, CR is the crossover constant $\in [0, 1]$ which has to be determined by user. $\text{rnbr}(i)$ is a randomly chosen index $\in \{1, 2, \dots, D\}$ which ensures that $\boldsymbol{\mu}_{ji,G+1}$ gets at least one parameter from $\boldsymbol{v}_{ji,G+1}$.

4. Selection

If vector $\boldsymbol{\mu}_{ji,G+1}$ yields a smaller cost function value than $\boldsymbol{\Theta}_{i,G}$, then $\boldsymbol{\Theta}_{i,G+1}$ is set to $\boldsymbol{\mu}_{ji,G+1}$. Otherwise, the old value $\boldsymbol{\Theta}_{i,G}$ is retained as $\boldsymbol{\mu}_{ji,G+1}$.

3.3 CWN-E-ELM

To ensure a compact network with a good generalization performance and fast learning speed, the differential evolution and the extreme learning machine are combined to train the proposed composite function wavelet neural network. In CWN-E-ELM, the differential evolutionary algorithm is used to select the network parameters including the input weights, the dilation and translation values and the MP generalized inverse is adopted to derived the output weights. A parameter initialization method that takes the input information into account is used to

initialize the dilation and translation parameters in the proposed CWN-E-ELM. The flow chart of combining the DE algorithm and the ELM algorithm to train CFWNN is illustrated in Fig. 3.1. The detailed learning steps are given as follows

Figure 3.1: Flow chart of the proposed CWN-E-ELM.

Step 1) **Initialization**

A set of populations is created initially at generation G , where each population is composed of input weights and the dilation and translation parameters as

$$\Theta_{i,G} = \left[v_{10}^i, v_{11}^i, \dots, v_{1I}^i, v_{20}^i, v_{21}^i, \dots, v_{2I}^i, \dots, v_{H0}^i, v_{H1}^i, \dots, v_{HI}^i, \right. \\ \left. a_1^i, a_2^i, \dots, a_H^i, b_1^i, b_2^i, \dots, b_H^i \right] \quad (3.5)$$

where $i = 1, 2, \dots, NP$. The input weight v_{jn}^i ($j = 1, 2, \dots, H$) ($n = 0, 1, \dots, I$) is randomly initialized within the range $[-1, 1]$. To ensure that the hidden node activation functions cover the input domain evenly, the following dilation and translation parameters initialization method used in [83] and the algorithm CFWNN-ELM is extended to NP populations and adopted in our work. Let $[x_{n,\min}, x_{n,\max}]$ ($n = 1, 2, \dots, I$) represent the domain containing the input item x_n in all the observed samples (the bias term x_0

is not taken into account here) and t^* , ψ^* denote the time-domain center and radius of the mother wavelet ψ , respectively. Then, the domain of the generated h th hidden node wavelet function $\psi_{a_h^i, b_h^i}$ is given as

$$[b_h^i + a_h^i (t^* - \psi^*), b_h^i + a_h^i (t^* + \psi^*)].$$

The input information range of the h th hidden layer can be calculated as $\left[v_{h0}^i x_0 + \sum_{n=1}^I v_{hn}^i x_{n,\min}, v_{h0}^i x_0 + \sum_{i=n}^I v_{hn}^i x_{n,\max} \right]$, where $h = 1, 2, \dots, H$ and $n = 1, 2, \dots, I$. With the same initialization method used in CFWNN-ELM, the dilation parameter a_h^i and translation parameter b_h^i in the h th hidden layer can be derived with the following equations (3.6) and (3.7), respectively,

$$a_h^i = \frac{\sum_{n=1}^I v_{hn}^i (x_{n,\max} - x_{n,\min})}{2\psi^*} \quad (3.6)$$

$$b_h^i = v_{h0}^i x_0 + \frac{\sum_{n=1}^I v_{hn}^i x_{n,\max} (\psi^* - t^*) + \sum_{i=n}^I v_{hn}^i x_{n,\min} (\psi^* + t^*)}{2\psi^*} \quad (3.7)$$

where the time-domain center and radius are calculated with formulae in equations (2.12) and (2.13), respectively.

Step 2) **Output weights and the cost function for initializations**

For each population $\Theta_{i,G}$ ($i = 1, 2, \dots, NP$) initialized above, the corresponding output weight $W^{\Theta_{i,G}}$ and the root mean square error (RMSE $^{\Theta_{i,G}}$)

can be calculated according to (2.7) and the following (3.8), respectively,

$$\text{RMSE}^{\Theta_{i,G}} = \sqrt{\frac{\sum_{k=1}^N \left(\sum_{j=1}^M \left(\sum_{i=1}^H w_{ji} g(\psi_{a_i, b_i}(\mathbf{v}_i^T \cdot \hat{\mathbf{x}}^k)) - d_j^k \right)^2 \right)}{N \times M \times H}}. \quad (3.8)$$

Similar to BP and SLFN-E-ELM, the RMSE is calculated on a validating dataset, which normally has no overlap with the training data to avoid overfitting.

Step 3) Mutation and crossover

The mutation and crossover steps of the DE algorithm shown in Section 3.2 are applied here to generate new populations $\boldsymbol{\mu}_{i,G+1}$ ($i = 1, 2, \dots, NP$) for the candidates of next generation.

Step 4) Calculation of output weights and cost function

For each new candidate $\boldsymbol{\mu}_{i,G+1}$ ($i = 1, 2, \dots, NP$), the network output weight $W^{\boldsymbol{\mu}_{i,G+1}}$ and the root mean square error ($\text{RMSE}^{\boldsymbol{\mu}_{i,G+1}}$) can be derived according to (2.7) and (3.8), respectively.

Step 5) Selection

Generally, in the selection step of the DE algorithm, the RMSE of the new candidate $\boldsymbol{\mu}_{i,G+1}$ is compared with the RMSE of the original one $\Theta_{i,G}$, and the vector who has the smaller RMSE will be retained to the next generation. However, as analyzed in SLFN-E-ELM [94], a small validation error does not guarantee a small testing error. Therefore, the norm of the output weight $\|W\|$ is added as one more criteria for selection as mentioned in SLFN-E-ELM [94], which was on account of the analysis of in [96]

that the neural networks tend to have better generalization performance with smaller weights. Consequently, the criterions of selecting the new generation $\Theta_{i,G+1}$ can be described as follows

$$\Theta_{i,G+1} = \begin{cases} \mu_{i,G} & \text{if } \text{RMSE}^{\Theta_{i,G}} - \text{RMSE}^{\mu_{i,G+1}} > \varepsilon \cdot \text{RMSE}^{\Theta_{i,G}}, \\ \mu_{i,G} & \text{if } |\text{RMSE}^{\Theta_{i,G}} - \text{RMSE}^{\mu_{i,G+1}}| < \varepsilon \cdot \text{RMSE}^{\Theta_{i,G}} \\ & \text{and } \|W^{\mu_{i,G+1}}\| < \|W^{\Theta_{i,G}}\|, \\ \Theta_{i,G} & \text{else,} \end{cases} \quad (3.9)$$

where ε is the preset positive tolerance rate and $i = 1, 2, \dots, NP$.

Once the populations of the new generation are generated, repeat from step 3) to step 5) until the preset goal is reached or the maximum number of learning iterations is completed.

3.4 Performance Evaluations

In this section, the proposed CWN-E-ELM is compared with some relevant methods, namely, SLFN-E-ELM, CFWNN-ELM and support vector machine [97, 98] (SVM) on several problems in the function approximation, real-world data regressions and classifications as described in the performance evaluation section of Chapter 2. For all these methods, the experimental set-ups are similar to the one used in Section 2.4. For CWN-E-ELM and SLFN-E-ELM, the parameters of the DE algorithm, including the amplification factor F , the crossover constant CR and the population sizes NP are problem dependent, i.e., for different problem, the values of these parameters are assigned differently.

3.4.1 Evaluation on Regression Problems

Simulations on the approximations of the three nonlinear functions and the regressions of five real-world datasets mentioned in Section 2.4 using CWN-E-ELM and some related methods are given below.

A. Evaluation on function approximations

Function f_1 is the famous ‘SinC’ function and the last two are a one-dimensional piecewise function and a two-dimensional nonlinear function, respectively. For function approximation problems, the parameters of the DE are set as follows: the amplification factor F and the crossover constant CR are set to 1 and 0.8, respectively. The number of populations NP is determined manually according to the number of training samples, where the more the training samples is, the larger the number of populations NP will be used.

For function f_1 , a training set $(x_i, f_1(x_i))$ and a testing set $(x_i, f_1(x_i))$ with 400 points, respectively, are generated where x_i is uniformly distributed in the input region $[-10, 10]$ for all these four algorithms. For CWN-E-ELM and SLFN-E-ELM, the training dataset is separated into two non-overlapped subsets with a partition ratio 2 : 1, where one is for training and the other is for validation in the differentials evolution phase. The number of populations NP is set to be 30 in this simulation.

Similar to function f_1 , a training set $(x_i, f_2(x_i))$ and a testing set $(x_i, f_2(x_i))$ with 400 points, respectively, are generated where x_i is uniformly distributed in the input region $[0, 1]$. For CWN-E-ELM and SLFN-E-ELM, the training dataset is separated into the training subset and the validation subset with the same partition ratio used in function f_1 and the number of populations NP is set to be 40.

For function f_3 , a training dataset $((x_{1i}, x_{2i}), f_3(i))$ and a testing dataset $((x_{1i}, x_{2i}), f_3(i))$ are collected where (x_{1i}, x_{2i}) are uniformly randomly distributed over the input domain $D = [-10, 10] \times [-10, 10]$, which forms a 40×40 grid. Similarly, a validation dataset is separated from the training dataset for CWN-E-ELM and SLFN-E-ELM and the partition ratio is the same to the one used in function f_1 and f_2 . The numbers of populations NP for CWN-E-ELM and SLFN-E-ELM are set to be 80 in this experiment.

To make these approximation problems ‘real’, uniformly distributed noise in $[-0.2, 0.2]$ is added to all the training samples of these three functions, while the testing data remain noise-free. In the simulations, the number of hidden nodes is incremented by 5 for CWN-E-ELM, SLFN-E-ELM and CFWNN-ELM. The nearly optimal numbers of nodes for these three NNs are then decided based on cross-validation method. Table 3.1 shows the performance comparisons using these four methods on the three function approximations. All these results are collected by averaging the results of 50 trials, and for each trial, all the datasets are randomly generated and validation datasets are randomly collected from the training datasets with the partition ratio mentioned above.

From Table 3.1, one can easily find that CWN-E-ELM wins the lowest testing RMSEs among these four methods for all three function approximations while CFWNN-ELM has the shortest training time. Although the training time by CWN-E-ELM is longer than the one by CFWNN-ELM as the DE algorithm is adopted for searching the optimal network parameters, the testing RMSE obtained by CWN-E-ELM is much lower than the one by CFWNN-ELM, especially for the piecewise function f_2 and the two-dimensional function f_3 . In our simulations, the training time costed by support vector machine for regression (SVR) includes the

Table 3.1: Comparisons for function approximations

Func.	NNs	Testing		Training time (s)	Nodes / SVs	NP / (C, γ)
		RMSE	Dev			
f_1	CWN-E-ELM	0.0112	0.0029	5.4409	10	30
	SLFN-E-ELM	0.0147	0.0041	1.1247	10	30
	CFWNN-ELM	0.0187	0.0040	0.0036	20	/
	SVR	0.0258	0.0053	19.6114	199.85	$(2^{-3}, 2^{-3})$
f_2	CWN-E-ELM	0.0956	0.0200	5.3844	25	40
	SLFN-E-ELM	0.1138	0.0262	1.5781	25	40
	CFWNN-ELM	0.2205	0.0480	0.0062	30	/
	SVR	0.4545	0.0476	12.6745	264.06	$(2^5, 2^5)$
f_3	CWN-E-ELM	0.0534	0.0109	38.6711	70	80
	SLFN-E-ELM	0.5138	0.0667	65.7883	140	80
	CFWNN-ELM	0.3938	0.2005	0.1944	180	/
	SVR	0.4527	0.1880	397.6044	912.94	$(2^5, 2^{-3})$

searching time of the optimal combination of C and γ . The training time by SVR is the longest one among all these methods and the average support vectors used by SVR are larger than the number of hidden nodes adopted by other three methods. We can also find from Table 3.1 that the network sizes of CWN-E-ELM and SLFN-E-ELM are much more compact than other methods. For these three cases, only 10, 25 and 70 nodes are enough to ensure a reasonable generalization performance for CWN-E-ELM for f_1 , f_2 , and f_3 , respectively. Fig. 3.2 shows the testing RMSEs of the two-dimensional function f_3 obtained by CWN-E-ELM and CFWNN-ELM using 40×40 training samples with respect to different number of nodes. It is easy to find that to achieve the testing RMSE 0.2737, CFWNN-ELM requires 180 hidden nodes, while for CWN-E-ELM, only 50 hidden nodes are needed. Moreover, we can also see that the differences of the testing RMSEs between different numbers of hidden nodes obtained by CWN-E-ELM are smaller than the one derived by CFWNN-ELM. When the number of nodes used by CWN-E-ELM is 40, the testing RMSE is 0.6840, and when increasing the number of nodes to 200, the

Figure 3.2: Testing RMSEs of the two-dimensional function f_3 w.r.t different number of nodes.

corresponding testing RMSE derived by CWN-E-ELM is 0.0310. The difference between the RMSEs obtained by CWN-E-ELM with the hidden nodes 40 and 200 is only 0.6530. However, the difference between the RMSEs obtained by CFWNN-ELM with the hidden nodes 40 and 200 is higher to 7.9038.

Different sample sizes, from 20×20 pairs to 60×60 pairs for function f_3 are also used to train and test by all these four methods to validate the effectiveness of the proposed method. Fig. 3.3 depicts the curve of the averaging testing RMSEs collected by 50 trials. From this figure, one can easily find that CWN-E-ELM can

Figure 3.3: Testing RMSEs of the two-dimensional function f_3 w.r.t different sizes of samples.

achieve the best generalization performance among the four methods. For each data pair, the testing RMSE derived by CWN-E-ELM is many times lower than the results obtained by the other three methods. Fig. 3.4 to Fig. 3.7 depict the error surfaces between the target two-dimensional function and the approximation curves by all these four methods using 40×40 training sample pairs. It has shown that the error surface obtained by CWN-E-ELM is much more close to zero surface than other methods.

Figure 3.4: Approximation error surfaces of the two-dimensional function by CWN-E-ELM.

Figure 3.5: Approximation error surfaces of the two-dimensional function by SLFN-E-ELM.

Figure 3.6: Approximation error surfaces of the two-dimensional function by CFWNN-ELM.

Figure 3.7: Approximation error surfaces of the two-dimensional function by SVR.

B. Evaluation on real-world regression datasets

In this section, the performance of CWN-E-ELM is evaluated by regressions on several real-world datasets. The specifications of these real-world datasets are the same to the ones used in Table 2.3 of Section 2.4. All the attributes (inputs) have been normalized to the range of $[-1, 1]$ while the outputs have been normalized into $[0, 1]$ in the simulations. The number of populations NP for all these real-world datasets is set to be 80. For the first 4 datasets, the training data and the testing data are randomly generated from its whole dataset before each trial of simulation while the training and testing data of the last dataset are fixed for all trials of simulations. For CWN-E-ELM and SLFN-E-ELM, the validation set is obtained from the training set where the partition ratio between the training set and the validation set is 2 : 1.

Table 3.2: Comparisons for regressions of real-world datasets

Datasets	NNs	Testing		Training time (s)	Nodes / SVs	NP / (C, γ)
		RMSE	Dev			
Abalone	CWN-E-ELM	0.0767	0.0021	28.432	24	80
	SLFN-E-ELM	0.0774	0.0022	4.8721	22	80
	CFWNN-ELM	0.0775	0.0016	0.0450	50	/
	SVR	0.0821	0.0011	290.21	332.22	$(2^0, 2^{-3})$
Autompg	CWN-E-ELM	0.0712	0.0105	3.9783	30	80
	SLFN-E-ELM	0.0770	0.096	0.8967	20	80
	CFWNN-ELM	0.0743	0.0115	0.0084	50	/
	SVR	0.0987	0.0026	22.336	80.54	$(2^{-2}, 2^{-5})$
BosHousing	CWN-E-ELM	0.0922	0.0111	3.4312	26	80
	SLFN-E-ELM	0.1055	0.0124	0.8675	24	80
	CFWNN-ELM	0.1067	0.0114	0.0109	60	/
	SVR	0.1102	0.0069	13.399	57.24	$(2^1, 2^{-6})$
CalHousing	CWN-E-ELM	0.1311	0.0020	188.43	40	80
	SLFN-E-ELM	0.1313	0.0025	65.78	40	80
	CFWNN-ELM	0.1290	0.0018	0.3318	50	/
	SVR	0.1193	0.0022	1836.7	2176.2	$(2^3, 2^1)$
Auto Price	CWN-E-ELM	0.0512	0.0076	1.0785	12	80
	SLFN-E-ELM	0.0611	0.0096	0.3521	10	80
	CFWNN-ELM	0.0912	0.0209	0.0001	30	/
	SVR	0.0945	0.0098	5.2988	22.75	$(2^8, 2^{-5})$

In the simulations, the number of hidden nodes is incremented by 5 for CWN-E-ELM, SLFN-E-ELM and CFWNN-ELM. The nearly optimal numbers of nodes for these three NNs are then decided based on cross-validation method. For SVR, the cost parameter C and kernel parameter γ are searched in a grid and the best parameter pairs are collected in terms of the generalization performance. Average results of 50 trials for all these four methods are reported. Table 3.2 shows the testing RMSEs, the standard derivations of testing RMSEs of these datasets, the numbers of hidden nodes and the training times used for these methods. As observed from this table, CWN-E-ELM obtains the best generalization performances in most cases among all these methods. The numbers of hidden nodes needed for CWN-E-ELM are much smaller than the one of CFWNN-ELM and the average support vectors of SVR. The training time used by CFWNN-ELM is the smallest among all these methods. Although the training times by CWN-E-ELM and SLFN-E-ELM are longer than CFWNN-ELM as the DE algorithm is applied to optimize the parameters of these two NNs, they still learn much faster than SVR.

3.4.2 Evaluation on Classification using Real-world Datasets

The classification performance of the proposed CWN-E-ELM is tested in this section by using 21 real-world datasets, where the specifications of these real-world datasets are shown in Table 2.5 of Section 2.4. All the attributes (inputs) have been normalized to the range of $[-1, 1]$ in the simulations. The number of populations NP for all these real-world datasets is set to be 80 except for the Digit dataset. For the Digit dataset, the NP value is set to be 40 as the large number of NP takes a long training time. For CWN-E-ELM and SLFN-E-ELM, the validation set is

obtained from the training set where the partition ratio between the training set and the validation set is 2 : 1.

Table 3.3: Results comparisons for classification on datasets.

Datasets	CWN-E-ELM		SLFN-E-ELM		CFWNN-ELM		SVM	
	Testing (%)		Testing (%)		Testing (%)		Testing (%)	
	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev
Balance	94.05	1.45	92.77	1.84	93.24	1.63	95.88	1.31
Breast-can	96.75	0.75	96.10	0.80	96.21	0.80	95.55	0.82
Diabetes	79.85	1.76	78.45	1.85	78.02	2.56	77.31	2.73
Ecoli	86.70	1.86	86.48	1.98	86.20	2.05	85.83	2.79
Glass	92.55	2.12	92.20	2.35	92.34	2.24	91.84	2.78
Heart	81.62	2.29	80.96	3.71	76.85	3.43	76.10	3.46
Iris	96.85	2.30	95.80	3.78	95.92	3.05	94.36	2.76
Liver	70.22	2.89	71.56	2.14	69.45	3.44	68.24	4.58
Satimage	91.02	0.35	88.46	1.36	88.82	0.48	91.25	0.28
Segment	97.02	0.57	95.32	1.52	95.40	0.80	96.93	0.62
Sonar	78.86	2.56	75.64	2.54	76.45	2.38	83.48	3.88
Soybean	99.11	2.12	99.24	1.66	98.46	2.54	99.56	1.32
Spambase	93.01	0.48	91.44	0.50	92.35	0.56	93.50	0.45
Waveform	86.30	0.45	85.82	0.65	85.90	0.55	85.78	0.62
Wine	97.80	1.24	97.20	1.52	97.66	1.42	97.48	1.57
Digit	97.20	0.21	97.43	0.18	96.56	0.44	98.14	0.01
Hayes	75.32	3.45	76.25	3.26	74.95	3.56	75	0
Monk1	86.25	2.12	84.65	2.24	84.65	2.24	94.44	0.01
Monk2	83.12	1.52	81.41	1.87	82.44	1.68	84.72	0.01
Monk3	88.34	2.22	88.26	2.35	87.90	2.45	90.04	0.01
Protein	88.75	1.05	88.15	1.25	88.12	1.22	90.63	0.01

For CFWNN-ELM, CWN-E-ELM, and SLFN-E-ELM, the number of hidden nodes is incremented during the training phase and the nearly optimal numbers of nodes are then decided based on cross-validation method. Table 3.3 shows the average simulation results comparisons with 50 trials for these four methods. From this table, we can see that the proposed CWN-E-ELM wins the highest success testing rate in 9 datasets out of 21, the SVM wins the highest success testing rate in 10 datasets and the SLFN-E-ELM wins the rest 2 datasets. Moreover, we can also find that the proposed CWN-E-ELM algorithm performs better than the SLFN-E-ELM algorithm and the CFWNN-ELM method in 19 and all datasets, respectively. The network complexities including the training time, the number of hidden nodes, and the number of NP value are shown in Table 3.4. As can be

Table 3.4: Comparisons of network complexities.

Datasets	CWN-E-ELM			SLFN-E-ELM			CFWNN-ELM		SVM		
	Training Time	Nodes	NP	Training Time	Nodes	NP	Training Time	Nodes	Training Time	SVs	(C, γ)
Balance	3.2548	10	80	2.1547	10	80	0.0098	35	23.1122	47.18	$(2^{12}, 2^{-3})$
Breast-can	2.1458	10	80	1.5624	10	80	0.0086	40	18.0719	56.22	$(2^1, 2^{-3})$
Diabetes	6.6987	10	80	2.2978	10	80	0.0102	35	364.6	62.28	$(2^{-1}, 2^{-1})$
Ecoli	1.1256	10	80	0.8755	10	80	0.0026	20	10.942	67.86	$(2^{10}, 2^{-7})$
Glass	1.1056	10	80	0.7852	10	80	0.0018	30	4.1444	47.88	$(2^1, 2^2)$
Heart	1.2247	10	80	0.2062	10	80	0.0016	30	6.7094	81.6	$(2^4, 2^{-1})$
Iris	0.9125	10	80	0.1500	10	80	0.0009	20	4.5488	23.3	$(2^8, 2^0)$
Liver	1.4521	10	80	0.9854	10	80	0.0044	30	5.0789	156.2	$(2^3, 2^1)$
Satimage	325.67	100	80	115.44	100	80	2.8228	500	681.77	1126.8	$(2^0, 2^{-1})$
Segment	210.55	70	80	55.46	70	80	0.5156	300	452.9094	379.4	$(2^4, 2^0)$
Sonar	3.5684	20	80	2.5614	20	80	0.0086	50	8.9113	69.7	$(2^8, 2^{-1})$
Soybean	2.1541	40	80	1.2618	40	80	0.0012	90	4.2516	21.20	$(2^7, 2^{-5})$
Spambase	185.23	50	80	49.85	50	80	0.7654	200	913.76	452.6	$(2^{11}, 2^{-6})$
Waveform	125.65	50	80	42.38	50	80	0.5644	100	1253.3	1863.6	$(2^4, 2^{-10})$
Wine	1.2584	10	80	0.8954	10	80	0.0076	20	5.8941	47.3	$(2^2, 2^{-1})$
Digit	268.45	100	40	156.98	100	40	1.4627	200	5894.8	1133	$(2^3, 2^{-1})$
Hayes	1.2248	20	80	0.4526	20	80	0.0058	50	8.1366	82	$(2^8, 2^{-6})$
Monk1	1.2278	30	80	0.4678	30	80	0.0088	50	6.6609	51	$(2^{12}, 2^{-3})$
Monk2	1.2564	50	80	0.5874	50	80	0.0553	100	11.4766	101	$(2^{12}, 2^{-1})$
Monk3	0.9058	20	80	0.2015	20	80	0.0205	60	6.4887	55	$(2^8, 2^{-2})$
Protein	85.27	50	80	34.95	50	80	1.7655	120	274.9672	261	$(2^7, 2^4)$

seen from this table, the numbers of hidden nodes needed for CWN-E-ELM and SLFN-E-ELM are fewer than the one of CFWNN-ELM and the average support vectors of SVM. Moreover, we can also find that CWN-E-ELM and SLFN-E-ELM learn faster than SVM in all datasets.

3.5 Conclusions

From the simulation results presented in the previous section, we have the following observations.

1. Compared with the CFWNN-ELM, the adoption of the differential evolution algorithm enables the CWN-E-ELM to efficiently choose the optimal hidden node parameters with a smaller network size by redesigning the network. The generalization performance of CWN-E-ELM is also better because the DE algorithm is incorporated into searching for the optimal network parameters. The only disadvantage of CWN-E-ELM is that the training time is longer due to the introduction of DE.
2. Although the number of hidden nodes and the training time for both CWN-E-ELM and SLFN-E-ELM are about the same, CWN-E-ELM performs much better than SLFN-E-ELM in general.
3. Compared with SVR/SVM, the proposed CWN-E-ELM is able to achieve a much better generalization performance with a faster learning speed for most of the simulations.

In conclusion, the CWN-E-ELM which incorporates the differential evolution algorithm into the composite function wavelet neural network trained with extreme

learning machine, has been proposed in this chapter. In most of the simulations, CWN-E-ELM has shown better performances in function approximations, real-world dataset regressions and real-world dataset classifications than some relevant methods. Moreover, the network size of CWN-E-ELM is much more compact than CFWNN-ELM due to the introduction of the DE algorithm which helps select the optimal hidden nodes.

Chapter 4

Voting Based Extreme Learning Machine for Signal Classification

4.1 Introduction

In this chapter, we focus on improving the classification performance of the general ELM. Generally, ELM performs classification by mapping the signal label to a high dimensional vector and transforming the classification task to a multi-output function regression problem. A problem with ELM is that as the hidden node learning parameters in ELM are randomly assigned and they remain unchanged during the training procedure, the classification boundary may not be an optimal one. Some samples may be misclassified by ELM, especially for those which are near the classification boundary.

Generally, ELM adopts an One-Against-All (OAA) method to decompose multi-classification applications into multiple binary classifiers and transforms the classification application to a multi-output function regression problem. Then, the one

with the largest output value is used to represent the class label of the given sample. For a C -labels classification application, the output label $\mathbf{t}_i$ of a sample $\mathbf{x}_i$ is usually encoded to a C -dimensional vector $[t_{i1} \ t_{i2} \ \dots \ t_{iC}]^T$ with $t_{ij} \in \{1, -1\}$ ($j = 1, 2, \dots, C$). In the OAA approach, the class labels are first assigned to a sequences $\{1, 2, \dots, C\}$, respectively. Then, if the class label $\mathbf{t}_i$ of the sample $\mathbf{x}_i$ is c with $c \in \{1, 2, \dots, C\}$, then t_{ic} will be set to be 1 and the others are set to be -1 in the newly formed C -dimensional output vector. For example, suppose we are considering a 3-classes classification problem with class labels $\{1, 2, 3\}$. Then, if the class label of a test sample is 1, the corresponding output vector is $[1 \ -1 \ -1]^T$. Similarly, if the class label of the sample is 2, then, the corresponding output vector is $[-1 \ 1 \ -1]^T$. Therefore, the class label c^{test} of a testing sample $\mathbf{x}^{\text{test}}$ predicted by the ELM algorithm is the index of the largest entry in the corresponding output vector.

ELM constructs a nonlinear separation boundary in classification applications.

Figure 4.1: Classifiers using ELM with different random parameters.

Since the randomized hidden nodes are used and they remain unchanged during the training phase, some samples may be misclassified in certain realizations, especially for samples that are near the classification boundary. As illustrated in Fig. 4.1,

sample “⊗” which is in the class of “*” is misclassified to the class “o” using ELM in realization $n = 1$ for a 2-label classification problem. However, with different randomly initialized hidden node parameters in the ELM algorithm, this sample “⊗” is correctly classified in realizations $n = 2$ and $n = 3$. Therefore, decisions based on a single realization of ELM may not be reliable and the classification results in different realizations may vary due to different nonlinear separation boundaries constructed with different random hidden node learning parameters. To reduce the number of such misclassified samples, we introduce an improved algorithm called voting based extreme learning machine (in short, as V-ELM) for signal classification [39]. The main idea in V-ELM is to perform multiple independent ELM trainings instead of a single ELM training, and then to make the final decision based on the majority voting method [99].

The rest of this chapter is organized as follows. Section 4.2 presents the improved voting based extreme learning machine algorithm. Numerical simulation results to validate our proposed method comparisons with several related algorithms are shown in Section 4.3. Discussions and conclusions are given in Section 4.4 and Section 4.5, respectively. In Section 4.6 and Section 4.7, the proof of the proposition and numerical simulations to illustrate these propositions are presented.

4.2 Voting Based Extreme Learning Machine

To tackle the problem mentioned above and improve the classification performance of ELM, an algorithm referred to voting based extreme learning machine (V-ELM) [39] by incorporating multiple independent ELMs and making decision with a majority voting method is proposed in this section.

In V-ELM, several individual ELMs with the same number of hidden nodes and the same activation function in each hidden node are used. As in ELM, the optimal number of hidden nodes for a given application are usually located in a small region and the variation of the network performance with respect to different number of hidden nodes selected from this region is usually very small. Hence, for the easy implementation, we fix the number of hidden nodes for all the individual ELMs to the same value. All these individual ELMs are trained with the same dataset and the learning parameters of each ELM are randomly initialized independently. The final class label is then determined by majority voting on all the results obtained by these independent ELMs. Suppose that K independent networks trained with the ELM algorithm are used in V-ELM. Then, for each testing sample $\mathbf{x}^{\text{test}}$, K prediction results can be obtained based on these independent ELMs. A corresponding vector $S_{K,\mathbf{x}^{\text{test}}} \in \mathbf{R}^C$ with dimension equal to the number of class labels is used to store all these K results of $\mathbf{x}^{\text{test}}$, where if the class label predicted by the k th ($k \in [1, \dots, K]$) ELM is i , the value of the corresponding entry i in the vector $S_{K,\mathbf{x}^{\text{test}}}$ is increased by one, that is

$$S_{K,\mathbf{x}^{\text{test}}}(i) = S_{K,\mathbf{x}^{\text{test}}}(i) + 1. \quad (4.1)$$

After all these K results are assigned to $S_{K,\mathbf{x}^{\text{test}}}$, the final class label of $\mathbf{x}^{\text{test}}$ is then determined by conducting a majority voting with the following equation

$$c^{\text{test}} = \arg \max_{i \in [1, \dots, C]} \{S_{K,\mathbf{x}^{\text{test}}}(i)\}. \quad (4.2)$$

The algorithmic description of the proposed V-ELM is presented in Fig. 4.2.

For a C -label classification application, assuming that the true label of the testing sample $\mathbf{x}^{\text{test}}$ is c , we have the following observations.

Proposition 4.1. (Appendix 4.6) *Given a standard SLFN with L hidden nodes, an output function $G(\cdot)$ and a set of training samples $\{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^N$, assume that the probability of correctly predicting the testing sample $\mathbf{x}^{\text{test}}$ using ELM [2] under all different possible hidden node learning parameters $\mathbf{a}$ and b is $P_{ELM}(c|\mathbf{x}^{\text{test}})$. If the following inequality holds*

$$P_{ELM}(c|\mathbf{x}^{\text{test}}) > \max \{P_{ELM}(i|\mathbf{x}^{\text{test}})\}_{i \in [1, \dots, C] \text{ and } i \neq c} \quad (4.3)$$

where $P_{ELM}(i|\mathbf{x}^{\text{test}})$ is the probability that ELM classifies $\mathbf{x}^{\text{test}}$ to category i that is different from the class c , then, with a sufficiently large independent training number K , V-ELM is able to correctly classify $\mathbf{x}^{\text{test}}$ with probability one.

The above proposition states that when the assumption (4.3) holds, with a sufficiently large independent training number K , V-ELM could make a correct prediction with probability one.

Proposition 4.2. *Assume condition (4.3) holds. Let*

$$P_{ELM}(c_1|\mathbf{x}^{\text{test}}) = \max \{P_{ELM}(i|\mathbf{x}^{\text{test}})\}_{i \in [1, \dots, C] \text{ and } i \neq c}. \quad (4.4)$$

That is, $P_{ELM}(c_1|\mathbf{x}^{\text{test}})$ is the incorrect probability for the highest class c_1 for the testing sample $\mathbf{x}^{\text{test}}$. Then, the required independent training number K for V-ELM to correctly classify $\mathbf{x}^{\text{test}}$ is related to the difference between $P_{ELM}(c|\mathbf{x}^{\text{test}})$ and

Algorithm V-ELM

Given:

- a training set $\aleph = \{(\mathbf{x}_n, \mathbf{t}_n) \mid \mathbf{x}_n \in \mathbf{R}^d, \mathbf{t}_n \in \mathbf{R}^m\}_{n=1}^N$
- the hidden node output function $G(\mathbf{a}, b, \mathbf{x})$
- hidden node number L
- independent training number K
- zero valued vector $S_K \in \mathbf{R}^C$, C is the number of classes

Training phase:

- (1) Set $k = 1$
- (2) **while** ($k \leq K$) **do**
 - Randomly assign the learning parameters $(\mathbf{a}_j^k, b_j^k)$ ($j = 1, 2, \dots, L$) of the k th ELM
- (3) Calculate the hidden layer output matrix $\mathbf{H}^k$
- (4) Calculate the output weight β^k : $\beta^k = (\mathbf{H}^k)^\dagger \mathbf{T}$, where $\mathbf{T}$ is the target output matrix
- (5) $k = k + 1$
- (6) **end while**

Testing phase:

- (1) **for** any testing sample $\mathbf{x}^{\text{test}}$,
- (2) Set $k = 1$
- (3) **while** ($k \leq K$) **do**
 - Using the k th trained basic ELM with learning parameters $(\mathbf{a}_j^k, b_j^k, \beta_j^k)$ to predict the label of the testing sample $\mathbf{x}^{\text{test}}$, say, as i where $i \in [1, 2, \dots, C]$
- (4) Then $S_{K, \mathbf{x}^{\text{test}}}(i) = S_{K, \mathbf{x}^{\text{test}}}(i) + 1$
- (5) $k = k + 1$
- (6) **end while**
- (7) The final class label of the testing sample $\mathbf{x}^{\text{test}}$ is

$$c^{\text{test}} = \arg \max_{i \in [1, \dots, C]} \{S_{K, \mathbf{x}^{\text{test}}}(i)\}$$

- (8) **end for**

Figure 4.2: Algorithmic description of the proposed V-ELM.

$P_{ELM}(c_1 | \mathbf{x}^{test})$. The larger the difference is, the smaller the independent training number K is required. This means that for samples that have obviously discriminative features to the samples from other classes, correct predictions could always be obtained at each ELM. However, for samples that are close to the classification boundary but still satisfy condition (4.3), the proposed V-ELM can improve the classification performance if a sufficiently large independent training number K is used.

Proposition 4.3. *If $P_{ELM}(c | \mathbf{x}^{test}) < P_{ELM}(c_1 | \mathbf{x}^{test})$, with a sufficiently large independent training number K , the success classification rate obtained by V-ELM will go to 0%. That is, V-ELM will predict $\mathbf{x}^{test}$ to the category c_1 under this condition.*

To illustrate these three propositions numerically, simulation results on classifying real world dataset are shown in Appendix 4.7.

4.3 Performance Evaluations

In this section, the performance of the proposed V-ELM is compared with the original ELM [2], SVM [98] and several other recent classification methods, including Optimally pruned extreme learning machine (OP-ELM) [100], Backpropagation algorithm (BP) [13, 101, 102], K nearest neighbors algorithm (KNN) [103, 104], robust fuzzy relational classifier (RFRC) [105], radial basis function neural network (RBFNN) [106] and multiobjective simultaneous learning framework (MSCC) [107]. Simulations are conducted on many real world classification datasets. All experiments on ELM, V-ELM, SVM and OP-ELM are carried in the Matlab 7.4 environ-

ment running on an ordinary PC with 2.66 GHZ CPU and 2 GB RAM. Both the Gaussian radial basis function (RBF) activation function $G(\mathbf{a}, b, \mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{a}\|^2}{b}}$ and the sigmoid additive activation function $G(\mathbf{a}, b, \mathbf{x}) = \frac{1}{1+e^{-(\mathbf{a} \cdot \mathbf{x} + b)}}$ are used in ELM and V-ELM.

4.3.1 Comparisons with ELM

The performance of V-ELM is compared with the conventional ELM whose training is based on a single network structure on 21 real world datasets in this subsection. The specifications of these real-world datasets are shown in Table 2.5 of Section 2.4. Similar to the set-up given in Section 2.4, the training and testing data of the first 15 datasets in Table 2.5 are reshuffled at each trial of simulation while the training and testing data of the last 6 datasets are fixed for all trials of simulations.

For both V-ELM and ELM, the number of hidden nodes is gradually increased and the optimal number of nodes is then selected based on the cross-validation method. Average results of 50 trials of simulations for each fixed size of SLFN are obtained and the performance obtained by V-ELM and ELM is reported in this section. For V-ELM, 7 independent ELMs are used for voting. Table 4.1 lists the classification results of V-ELM and ELM on all these 21 datasets. As observed from this table, the proposed V-ELM outperforms ELM in all 21 datasets. For several datasets, the enhancement of the success rate is apparent. For example, compared with ELM, the enhancements of success rates of the datasets Sonar, Hayes, Monk2, Monk3 and Protein are more than 3%. The standard deviations of V-ELM are much lower than ELM for all these 21 datasets, as shown in bold

Table 4.1: Performance comparisons with ELM

Datasets	ELM [2]				V-ELM ($K = 7$)			
	Sigmoid		RBF		Sigmoid		RBF	
	Testing (%)				Testing (%)			
	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev
Balance	90.53	1.79	90.16	2.16	91.24	1.49	91.00	1.63
Breast-can	95.52	1.38	94.45	1.60	96.75	0.94	96.39	0.92
Diabetes	77.57	2.76	77.24	2.65	78.56	2.46	78.10	2.68
Ecoli	85.56	2.31	85.43	2.31	86.42	1.81	86.41	2.03
Glass	91.91	2.54	92.33	2.45	92.77	2.31	93.33	2.21
Heart	80.68	2.79	71.34	4.66	82.53	2.27	77.05	2.68
Iris	96.60	5.93	97.60	4.76	98.40	4.22	98.00	4.95
Liver	70.01	3.84	68.36	3.75	72.17	2.81	71.37	3.11
Satimage	89.04	0.4	89.90	0.48	89.53	0.35	90.32	0.49
Segment	95.01	0.74	94.85	0.99	95.36	0.68	95.13	0.69
Sonar	73.31	4.29	72.67	3.91	79.11	3.51	78.69	3.72
Soybean	99.00	2.21	88.61	7.77	99.83	0.86	95.54	5.54
Spambase	90.50	0.68	88.81	0.79	91.60	0.64	90.65	0.58
Waveform	85.50	0.83	84.59	0.79	86.48	0.56	86.26	0.63
Wine	96.79	1.66	92.62	3.10	98.31	1.61	96.79	2.17
Digit	97.01	0.23	96.40	0.39	97.22	0.11	97.13	0.15
Hayes	75.86	4.76	61.14	7.09	79.00	4.87	66.57	4.99
Monk1	82.78	2.34	78.45	3.26	85.75	1.41	82.98	2.10
Monk2	79.27	2.11	81.25	2.37	82.81	1.15	85.84	1.27
Monk3	87.38	2.83	79.02	2.81	90.44	1.00	85.12	1.64
Protein	87.90	1.34	87.89	1.43	90.06	0.66	90.35	0.66

face in Table 4.1. For V-ELM, using sigmoid function as the activation function performs better than the one using RBF activation function in general. Among these two activation functions, V-ELM with sigmoid function wins 17 out of 21 datasets while V-ELM with RBF wins the remaining 4 datasets.

Table 4.2 shows the comparisons of the network complexities of ELM and V-ELM, including the training time and the number of hidden nodes. It can be found from this table that the computational time of using V-ELM is about 7 times on average that of using ELM when $K = 7$ in the above simulations. However, the

Table 4.2: Comparisons of network complexities with ELM

Datasets	ELM				V-ELM ($K = 7$)			
	Sigmoid		RBF		Sigmoid		RBF	
	Time	Nodes	Time	Nodes	Time	Nodes	Time	Nodes
Balance	0.0088	58	0.0094	60	0.0278	38	0.0584	60
Breast-can	0.0053	45	0.0153	85	0.0506	60	0.0856	75
Diabetes	0.0009	20	0.0044	28	0.0134	22	0.0159	22
Ecoli	0.0003	20	0.0003	24	0.0103	24	0.0091	24
Glass	0.0006	26	0.0003	26	0.0056	24	0.0100	34
Heart	0.0006	16	0.0009	16	0.0019	12	0.0081	20
Iris	0.0003	28	0.0019	24	0.0063	24	0.0066	18
Liver	0.0022	22	0.0019	30	0.0075	20	0.0119	24
Satimage	2.6556	500	3.5445	550	30.6547	600	31.6563	600
Segment	0.1953	200	0.4519	300	1.4778	200	2.0888	240
Sonar	0.0044	56	0.0028	50	0.0334	64	0.0356	64
Soybean	0.0008	90	0.0014	100	0.0037	90	0.0078	100
Spambase	0.3322	240	0.4594	280	3.2725	260	2.3675	240
Waveform	0.1266	100	0.1128	100	0.8734	100	0.8944	100
Wine	0.0006	24	0.0016	40	0.0053	20	0.0153	38
Digit	0.8741	200	0.8484	200	6.1709	200	6.2997	200
Hayes	0.0022	46	0.0053	58	0.0191	48	0.0294	58
Monk1	0.0031	45	0.0063	70	0.0631	85	0.0394	70
Monk2	0.0219	115	0.0225	110	0.1325	110	0.1231	105
Monk3	0.0075	70	0.0072	75	0.0494	75	0.0450	70
Protein	0.9968	125	0.9969	130	0.6853	155	0.6878	155

training phase can still be completed in less than one second for most applications. The training time costed by V-ELM is linearly increased with a proportion to the number of independent training ELMs used for voting.

4.3.2 Comparisons with SVM, OP-ELM, BP, and KNN

In this subsection, simulations using V-ELM, SVM [98], OP-ELM [100], BP [13, 101, 102], and KNN [103, 104] are conducted on 21 datasets, where the specifications of these datasets are given in Table 2.5 of Section 2.4. Note that the results reported here are obtained by us using Matlab codes provided in [98], [100], [101, 102], and [103, 104]. For V-ELM, 7 independent ELMs are adopted for training and majority voting. For SVM, the Gaussian RBF is used as the kernel function, the cost parameter C and the kernel parameter γ are searched in a grid formed by $C = [2^{12}, 2^{11}, \dots, 2^{-2}]$ and $\gamma = [2^4, 2^3, \dots, 2^{-10}]$ as suggested in [98] and the best

combination of these two parameters is then obtained in terms of the generalization performance. For OP-ELM, the three possible kernels, linear, sigmoid and Gaussian are used as a combination suggested in [100] that using the combination of these three functions as the activation function in OP-ELM is more robust and efficient than utilizing only one of them. For BP, the most frequently adopted Levenberg-Marquardt algorithm [101, 102] is used to train the neural network. For KNN, 7 nearest neighbors are used and the Euclidean norm is adopted to calculate the distance. Average results with 50 trials of simulations are reported in this section.

Table 4.3: Performance comparisons with SVM, OP-ELM, BP, and KNN

Datasets	SVM (RBF)		OP-ELM		BP		KNN		V-ELM ($K=7$)	
	Testing (%)		Testing (%)		Testing (%)		Testing (%)		Testing (%)	
	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev
Balance	95.88	1.31	92.31	1.83	90.92	2.14	87.00	1.80	91.24	1.49
Breast-can	95.55	0.82	95.33	1.29	95.01	1.66	96.32	1.03	96.75	0.94
Diabetes	77.31	2.73	77.34	3.17	77.23	2.81	74.09	2.73	78.56	2.46
Ecoli	85.83	2.79	85.20	2.88	80.27	3.91	83.68	2.22	86.42	1.81
Glass	91.84	2.78	91.65	3.23	91.30	3.09	90.18	2.60	93.33	2.21
Heart	76.10	3.46	81.05	2.96	71.25	8.54	80.79	2.57	82.53	2.27
Iris	94.36	2.76	97.80	8.93	95.60	3.00	96.04	2.23	98.40	4.22
Liver	68.24	4.58	65.85	4.75	66.50	4.45	61.46	3.27	72.17	2.81
Satimage	91.25	0.28	89.01	0.56	88.72	0.51	89.54	0.65	90.32	0.49
Segment	96.93	0.62	95.12	0.57	95.46	0.45	94.36	0.78	95.36	0.68
Sonar	83.48	3.88	71.70	4.79	70.31	5.40	66.30	4.93	79.11	3.51
Soybean	99.56	1.32	99.12	1.51	88.17	9.38	79.74	11.47	99.83	0.86
Spambase	93.50	0.45	91.23	0.78	92.06	0.78	88.61	0.53	91.60	0.64
Waveform	85.78	0.62	85.46	0.64	85.94	0.76	82.65	0.72	86.48	0.56
Wine	97.48	1.57	98.18	1.72	94.10	3.12	96.23	2.01	98.31	1.61
Digit	98.14	0.01	98.34	0.25	\	\	97.54	0.01	97.22	0.11
Hayes	75	0	70.43	4.95	74.43	7.08	75.00	0	79.00	4.87
Monk1	94.44	0.01	74.79	3.91	69.99	13.82	80.56	0.01	85.75	1.41
Monk2	84.72	0.01	70.35	3.58	72.84	2.92	71.53	0.01	85.84	1.27
Monk3	90.04	0.01	88.77	2.31	80.41	6.07	80.79	0.01	90.44	1.00
Protein	90.63	0.01	91.26	1.15	87.45	2.13	89.33	0.01	90.35	0.66

Table 4.3 shows the performance comparisons of our method with SVM, OP-ELM, BP, and KNN. From this table, it is easy to see that V-ELM wins the highest testing success rates in 13 datasets, while SVM has the highest testing success rates in 6 datasets and OP-ELM reaches the highest testing success rates in 2 datasets.

Table 4.4: Comparisons of network complexities with SVM, OP-ELM, BP, and KNN

Datasets	SVM (RBF)			OP-ELM		BP		KNN	V-ELM ($K = 7$)	
	Training Time	SVs	(C, γ)	Training Time	Nodes	Training Time	Nodes	Training Time	Training Time	Nodes
Balance	23.1122	47.18	$(2^{12}, 2^{-3})$	0.4550	40	0.8478	25	0.0544	0.0278	38
Breast-can	18.0719	56.22	$(2^1, 2^{-3})$	0.6100	25	1.9875	20	0.0725	0.0506	60
Diabetes	364.6	62.28	$(2^{-1}, 2^{-1})$	0.6709	16	0.4516	10	0.1353	0.0134	22
Ecoli	10.942	67.86	$(2^{10}, 2^{-7})$	0.6021	20	0.8712	20	0.0437	0.0103	24
Glass	4.1444	47.88	$(2^1, 2^2)$	0.0459	8	0.3459	20	0.0384	0.0100	34
Heart	6.7094	81.6	$(2^4, 2^{-1})$	0.0447	8	0.3063	5	0.0384	0.0019	12
Iris	4.5488	23.3	$(2^8, 2^0)$	0.0063	4	0.3050	5	0.0103	0.0063	24
Liver	5.0789	156.2	$(2^3, 2^1)$	0.0409	10	0.3234	15	0.0297	0.0075	20
Satimage	681.77	1126.8	$(2^0, 2^{-1})$	459.2	100	597.32	100	2.4516	30.6547	600
Segment	452.9094	379.4	$(2^4, 2^0)$	210.31	100	119.7812	200	4.5613	1.4778	200
Sonar	8.9113	69.7	$(2^8, 2^{-1})$	0.1391	12	3.1541	20	0.0681	0.0334	64
Soybean	4.2516	21.20	$(2^7, 2^{-5})$	0.0524	80	0.4197	10	0.0037	0.0037	90
Spambase	913.76	452.6	$(2^{11}, 2^{-6})$	227.5824	240	9.5344	10	6.6422	3.2725	260
Waveform	1253.3	1863.6	$(2^4, 2^{-10})$	3.2894	110	3.9066	10	2.3575	0.8734	100
Wine	5.8941	47.3	$(2^2, 2^{-1})$	0.0628	28	0.5175	30	0.0181	0.0053	20
Digit	5894.8	1133	$(2^3, 2^{-1})$	628.5689	240	\	\	9.2844	6.1709	200
Hayes	8.1366	82	$(2^8, 2^{-6})$	0.0988	16	0.7588	45	0.0059	0.0191	48
Monk1	6.6609	51	$(2^{12}, 2^{-3})$	0.0500	28	0.3253	10	0.2597	0.0631	85
Monk2	11.4766	101	$(2^{12}, 2^{-1})$	0.0594	25	1.6578	150	0.2778	0.1231	105
Monk3	6.4887	55	$(2^8, 2^{-2})$	0.0516	36	0.3159	10	0.2719	0.0494	75
Protein	274.9672	261	$(2^7, 2^4)$	19.9381	160	1326	60	0.3225	0.6878	155

From these results, we can see that V-ELM outperforms the other four algorithms in general. For the dataset Digit, BP algorithm always ran out of memory in our ordinary PC. Table 4.4 shows the network complexities of these five algorithms. It is easy to find from this table that with $K = 7$, V-ELM learns much faster than SVM and BP for all the datasets, faster than OP-ELM in all the datasets except for Iris, Monk1 and Monk2, and faster than KNN in all the datasets except for Soybean, Hayes and Protein.

4.3.3 Comparisons with several other recent algorithms

The performance of V-ELM is next compared with three other classification algorithms recently reported in the literature, namely the robust fuzzy relational classifier (RFRC) [105], the radial basis function neural network (RBFNN) [106] and the multiobjective simultaneous learning framework (MSCC) [107]. Note that the results of RFRC, RBFNN and MSCC are taken from references [105–107] directly. To make the comparison fair, in the simulations the training and testing datasets are randomly generated with half-half cases as done in [107] for V-ELM, where the specifications of these datasets are given in Table 2.5 of Section 2.4. Two different independent training numbers, $K = 7$ and $K = 51$, respectively, are used in V-ELM. All the results of V-ELM are obtained by averaging 50 trials.

Table 4.5 shows the simulation results of V-ELM and those of the three recent methods mentioned above. When $K = 7$, V-ELM performs better than RFRC and RBFNN in almost all datasets except for Sonar. The performance of V-ELM is about the same as that of MSCC, with V-ELM winning the highest testing success rate in 5 out of 11 datasets (underlined in Table 4.5), while MSCC reaching the

Table 4.5: Performance comparisons with several recent methods

Datasets	RFRC [105]		RBFNN [106]		MSCC [107]		V-ELM ($K = 7$)		V-ELM ($K = 51$)	
	Testing (%)		Testing (%)		Testing (%)		Testing (%)		Testing (%)	
	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev	Rate	Dev
Balance	84.7	1.5	90.5	1.0	90.8	1.2	<u>90.89</u>	1.25	91.35	1.03
Breast-can	92.0	1.6	95.0	1.2	97.3	0.7	96.70	1.02	97.34	0.60
Diabetes	70.7	3.2	74.2	2.3	76.5	1.1	<u>77.61</u>	1.62	77.78	1.77
Ecoli	81.8	3.3	85.2	2.7	85.0	2.4	<u>86.42</u>	1.81	86.68	1.99
Heart	80.9	2.2	82.5	2.3	84.2	1.8	82.89	2.80	83.31	2.55
Iris	95.3	1.1	96.4	1.6	97.1	1.7	97.07	2.01	97.60	0.84
Liver	61.0	2.4	70.8	3.6	68.2	5.7	<u>71.38</u>	2.66	72.00	2.79
Sonar	77.5	3.9	80.2	3.0	85.6	4.1	78.96	3.67	81.69	3.75
Soybean	99.1	1.7	98.1	1.7	100	0	99.83	0.86	100	0
Spambase	85.1	1.1	80.7	1.0	89.9	0.8	<u>91.60</u>	0.64	91.98	0.50
Wine	96.0	1.7	97.3	1.1	98.3	1.1	98.16	1.32	98.29	1.40

highest success rate in the remaining 6 datasets. However, when the number of independent ELMs is increased to $K = 51$, V-ELM performs better, with V-ELM winning the highest success rate in 8 out of 11 datasets (in bold face in Table 4.5).

4.4 Discussions

In this section, we first discuss the performance of V-ELM with respect to different independent training numbers K . Then, we briefly discuss three recent methods [34, 108, 109] which also exploit multiple classifiers in ELM.

It was stated in *Proposition 4.1* that for the V-ELM to work properly, a sufficiently large independent training number K is required. However, it may not be practically feasible to use V-ELM if the required K is too large, say well above 100 as the computational time of V-ELM increases with K proportionally, roughly about K times of that for a single ELM. It is hence important to know the performance of V-ELM as a function of K .

From the simulations conducted in the previous section, we know that for $K = 7$, V-ELM outperforms the original ELM by about 2% in average for the

21 datasets and also outperforms most of other methods except for MSCC which has about the same performance. When K increases to 35, the performance of V-ELM increases and in such a case, it also outperforms MSCC. The purpose of this section is to illustrate through simulations that in most practical applications, it suffices to implement V-ELM when K is between 5 and 35. For this purpose, two datasets Heart and Sonar are selected. The independent training number K , which is also the number of ELMs, is gradually increased from 1 to 151 with the interval 2. For each K , 5000 repetitions are conducted and the average results are reported here.

Fig. 4.3 depicts the curves of success rates of the datasets Heart and Sonar using V-ELM (using Sigmoid function) as a function of the number of ELMs K . It is easy to find from Fig. 4.3 that the success rates for both Heart and Sonar datasets increase monotonically with K . It is obvious that the changes are more dramatic at the beginning parts than at the ending parts. For example, in the dataset Heart, when K changes from 1 (corresponding to the original ELM algorithm) to 23, the success rate of V-ELM increases from 80.15% to 83.04%, an increase of 2.9%. However, when K is further increased from 23 to 151, the success rate only increases to 83.23%, or slightly less than 0.2%. As the training time by V-ELM with $K = 151$ is near 10 times of that with $K = 23$, we conclude that choosing K to be around 23 would be the best for this dataset. The situation is more obvious for the dataset Sonar. when K changes from 1 (corresponding to the original ELM algorithm) to 35, the success rate of V-ELM increases from 70.03% to 80.05%, an increase of more than 10%. However, when K is further increased from 35 to 151, the success rate only increases to 80.74%, or less than 0.8%. As the training time by V-ELM with $K = 151$ is about 5 times of that with $K = 35$,

Figure 4.3: Success rates of the datasets Heart and Sonar using V-ELM (Sigmoid) with respect to the independent training number K .

we conclude that choosing K to be around 35 would be the best for the dataset Sonar.

Fig. 4.4 displays the standard deviations of success rates among 5000 real-

Figure 4.4: Standard deviations of success rates of the datasets Heart and Sonar using V-ELM (Sigmoid) with respect to the independent training number K .

izations for the datasets Heart and Sonar using V-ELM (Sigmoid) with respect to different independent training numbers. One can find that the standard deviations decrease dramatically when the number of independent training times K increases from 1 to 30 for both these two datasets. However, when one keeps increasing

the number of independent training times after 30, the variations of the standard deviations among using different number of independent training times K are very small for these two datasets.

From the simulations on the above two datasets and many more other datasets, we recommend to implement V-ELM by choosing K between 5 and 35. The actual value of K depends on the application at hand. As the original ELM runs much faster than most other classification algorithms, the computational time of V-ELM with K in this range should still be comparable with or faster than other classification algorithms.

For the other datasets we have used in our performance evaluations section, the same performance can be found as the two datasets Heart and Sonar we have discussed here. We do not display all these results to save space.

Before ending this section, it is worth pointing out the difference between our method and three recent methods [34, 108, 109] which also exploit multiple classifiers in ELM. In [109], a multiple classifier system based on ELM was proposed. The main idea of [109] is to generate a set of classifiers using ELM and discard the ones with the recognition rate lower than the threshold that is calculated by averaging all the recognition rates among all these ELM classifiers, and then to update the whole classification system by generating new ELM classifiers instead of the discarded ones till the performance goal is reached. The difference between our algorithm and the method in [109] is that our voting procedure is operated on the individual sample while the threshold in [109] is an average value derived based on the whole validation set among all ELM classifiers. In addition, we do not adopt the updating procedure in our approach. In [34], an ensemble of on-line sequential extreme learning machine (EOS-ELM) was developed to lower the

variance of online sequential extreme learning machine (OS-ELM). By implementing multiple OS-ELM networks for training and taking the average value as the final output, EOS-ELM is more stable than OS-ELM. However, EOS-ELM focused on the online sequential learning and the performance of EOS-ELM has not been improved except for lowering the variance, as shown by the simulation results in [34]. In [108], the training dataset is divided into several non-overlap subsets and the training and validation phases are completed by conducting on each subset independently. This is different from our algorithm on that we make use of all the information in the whole dataset to train each independent ELM network while only partial samples are used for training each ELM in [108].

4.5 Conclusions

In this chapter, we have proposed an improved classification algorithm named voting based extreme learning machine to train SLFNs. Based on the simulation results, comparisons and discussions, we have the following conclusions:

1. Compared with the original ELM algorithm, the incorporation of the voting method enables V-ELM to achieve a much higher success classification rate in general. However, the improvement of performance of V-ELM is at the price of increasing the training time about K folds of that of the single ELM. Hence, a small independent training number K should always be tried first and gradually increased.
2. Compared with several recent classification algorithms, V-ELM is able to obtain better success rates in most cases for the many datasets we have at

hand.

3. Generally, the success rate obtained by V-ELM increases with K . For practical applications, we recommend to implement V-ELM by choosing K between 5 and 35, starting from 5 and gradually increasing K until 35 or a satisfactory validation result is obtained before K reaches 35. For large scale datasets applications, the number of independent ELMs should remain small as the larger the number is used, the more complex the calculation tends to be.

4.6 Appendix: Proof of Proposition 4.1

We assume that $P_{\text{ELM}}(1|\mathbf{x}^{\text{test}}), \dots, P_{\text{ELM}}(c-1|\mathbf{x}^{\text{test}}), P_{\text{ELM}}(c+1|\mathbf{x}^{\text{test}}), \dots, P_{\text{ELM}}(C|\mathbf{x}^{\text{test}})$ are the probabilities that ELM misclassifies the sample $\mathbf{x}^{\text{test}}$ to classes $1, \dots, c-1, c+1, \dots, C$ under all possible hidden node parameters $(\mathbf{a}, b)$. Then, we rearrange these misclassification probabilities in the descending order and denote as: $P_{\text{ELM}}(c_1|\mathbf{x}^{\text{test}}) \geq \dots \geq P_{\text{ELM}}(c_{C-1}|\mathbf{x}^{\text{test}})$ where $c_i \in (1, \dots, c-1, c+1, \dots, C)$ is the class label that ranks the i th largest misclassification probability among $P_{\text{ELM}}(1|\mathbf{x}^{\text{test}}), \dots, P_{\text{ELM}}(c-1|\mathbf{x}^{\text{test}}), P_{\text{ELM}}(c+1|\mathbf{x}^{\text{test}}), \dots, P_{\text{ELM}}(C|\mathbf{x}^{\text{test}})$. With assumption (4.3), there exists a scalar $\epsilon > 0$ such that

$$P_{\text{ELM}}(c|\mathbf{x}^{\text{test}}) - P_{\text{ELM}}(c_1|\mathbf{x}^{\text{test}}) > \epsilon. \quad (4.5)$$

In V-ELM, K independent ELMs are used where each ELM network is trained with the same dataset and the hidden node learning parameters are initialized independently. Assuming that $P_{\text{ELM},K}(i|\mathbf{x}^{\text{test}})$ ($i = c, c_1, \dots, c_{C-1}$) is the probability that predicting the testing sample $\mathbf{x}^{\text{test}}$ to label i among all these K ELMs, therefore,

we have $P_{\text{ELM},K}(i|\mathbf{x}^{\text{test}}) = \frac{S_{K,\mathbf{x}^{\text{test}}(i)}}{K}$. For a sufficiently large number K , we can expect that

$$\lim_{K \rightarrow \infty} P_{\text{ELM},K}(i|\mathbf{x}^{\text{test}}) = P_{\text{ELM}}(i|\mathbf{x}^{\text{test}}), \quad i = c, c_1, \dots, c_{C-1}$$

and $\sum_{i=c, c_1, \dots, c_{C-1}} P_{\text{ELM},K}(i|\mathbf{x}^{\text{test}}) = 1$. Hence, from the definition of limit [110], we can get that for any given positive scalar $\frac{1}{C}\epsilon \geq \delta > 0$ and for each i , there exists a corresponding positive integer K_{0i} , when $K_i \geq K_{0i}$, the following inequalities hold

$$|P_{\text{ELM},K_i}(i|\mathbf{x}^{\text{test}}) - P_{\text{ELM}}(i|\mathbf{x}^{\text{test}})| < \delta, \quad i = c, c_1, \dots, c_{C-1}.$$

Therefore, among all these K_i , set $K = \max\{K_i\} (i = c, c_1, \dots, c_{C-1})$, then, we have the following inequalities hold

$$|P_{\text{ELM},K}(i|\mathbf{x}^{\text{test}}) - P_{\text{ELM}}(i|\mathbf{x}^{\text{test}})| < \delta, \quad i = c, c_1, \dots, c_{C-1}.$$

When $i = c$, we get the following inequality from the above equation

$$P_{\text{ELM}}(c|\mathbf{x}^{\text{test}}) - \delta < P_{\text{ELM},K}(c|\mathbf{x}^{\text{test}}) < P_{\text{ELM}}(c|\mathbf{x}^{\text{test}}) + \delta. \quad (4.6)$$

For any $c_j \in (c_1, \dots, c_{C-1})$, we have

$$P_{\text{ELM},K}(c_j|\mathbf{x}^{\text{test}}) = 1 - P_{\text{ELM},K}(c|\mathbf{x}^{\text{test}}) - \sum_{i \neq c, c_j} P_{\text{ELM},K}(i|\mathbf{x}^{\text{test}}). \quad (4.7)$$

From (4.6) and (4.7), we have

$$P_{\text{ELM},K}(c_j | \mathbf{x}^{\text{test}}) < 1 - P_{\text{ELM}}(c | \mathbf{x}^{\text{test}}) + \delta - \sum_{i \neq c, c_j} P_{\text{ELM},K}(i | \mathbf{x}^{\text{test}}) \quad (4.8)$$

which leads to the following corresponding inequality

$$P_{\text{ELM},K}(c_j | \mathbf{x}^{\text{test}}) < P_{\text{ELM}}(c_j | \mathbf{x}^{\text{test}}) + (C - 1)\delta. \quad (4.9)$$

Using equations (4.5), (4.6) and (4.9), we have

$$\begin{aligned} P_{\text{ELM},K}(c | \mathbf{x}^{\text{test}}) &> P_{\text{ELM}}(c_1 | \mathbf{x}^{\text{test}}) + \epsilon - \delta \\ &\geq P_{\text{ELM}}(c_j | \mathbf{x}^{\text{test}}) + (C - 1)\delta \\ &> P_{\text{ELM},K}(c_j | \mathbf{x}^{\text{test}}) \quad \text{for any } c_j \in (c_1, \dots, c_{C-1}) \end{aligned} \quad (4.10)$$

The above inequality (4.10) means that with a sufficiently large independent training number K , the number of ELMs that predicts the sample $\mathbf{x}^{\text{test}}$ to category c is the largest one among all these K ELMs. Therefore, with a majority voting method, V-ELM could make a correct prediction with probability one.

4.7 Appendix: Numerical simulations to illustrate Propositions 4.1 to 4.3

In this appendix, we will use a real world application to illustrate *Propositions 4.1 to 4.3*. Through this application, we will show that the required number of ELMs for V-ELM to correctly classify $\mathbf{x}^{\text{test}}$ is related to the difference between

$P_{\text{ELM}}(c|\mathbf{x}^{\text{test}})$ and $P_{\text{ELM}}(c_1|\mathbf{x}^{\text{test}})$, where $P_{\text{ELM}}(c|\mathbf{x}^{\text{test}})$ is the probability of correctly predicting the testing sample using ELM under all different possible hidden node learning parameters and $P_{\text{ELM}}(c_1|\mathbf{x}^{\text{test}}) = \max\{P_{\text{ELM}}(i|\mathbf{x}^{\text{test}})\}_{i \in [1, \dots, C] \text{ and } i \neq c}$ is the incorrect probability for the highest class c_1 for the testing sample $\mathbf{x}^{\text{test}}$. The larger the difference is, the smaller the number of ELMs is required.

The 10-label dataset Protein is adopted in this appendix. In the simulation, 949 samples are used as the training set. Three samples, denoted as “ s_1 ”, “ s_2 ” and “ s_3 ” are selected as testing samples, where s_1 , s_2 and s_3 belong to the classes 2, 8 and 5, respectively. 2000 repetitions of ELMs are conducted and the rate of a testing sample $\mathbf{x}^{\text{test}}$ being classified into the category i by these 2000 ELMs (2000 possible input weight matrices $\mathbf{a}$ and hidden bias vectors $\mathbf{b}$) is used to approximate the probability that ELM classifies $\mathbf{x}^{\text{test}}$ to the category i (under all different possible $\mathbf{a}$ and $\mathbf{b}$), i.e., $P_{\text{ELM}}(i|\mathbf{x}^{\text{test}})$. In 2000 repeated trials, s_1 , s_2 and s_3 are correctly classified to the classes 2, 8 and 5 for 639, 945 and 388 times, respectively. Hence, the approximate correct probability for s_1 , s_2 and s_3 is $P_{\text{ELM}}(2|s_1) \approx 31.95\%$, $P_{\text{ELM}}(8|s_2) \approx 47.25\%$ and $P_{\text{ELM}}(5|s_3) \approx 19.4\%$, respectively. It follows that in these 2000 repeated trials, s_1 , s_2 and s_3 are misclassified to other classes for 1361, 1055 and 1612 times, or they have the approximate incorrect probability of 68.05%, 52.75% and 80.6%, respectively. Note that for all the three testing samples s_1 , s_2 and s_3 , the correct probability is lower than the corresponding incorrect probability. Moreover, s_1 , s_2 and s_3 are misclassified to classes 3, 2 and 4, for 586, 322 and 1540 times, respectively, which are the corresponding highest misclassification classes. Hence, the approximate incorrect probability for the highest class for s_1 , s_2 and s_3 is $P_{\text{ELM}}(3|s_1) \approx 29.3\%$, $P_{\text{ELM}}(2|s_2) \approx 16.6\%$ and $P_{\text{ELM}}(4|s_3) \approx 77\%$, respectively.

Table 4.6: Comparisons of misclassification numbers between V-ELM and ELM

Samples	s_1	s_2	s_3
ELM	1361	1055	1612
V-ELM ($K = 19$)	800	196	1993
V-ELM ($K = 77$)	491	0	2000

Figure 4.5: Misclassification numbers of the three samples with respect to the independent training number K .

For samples s_1 and s_2 , since the approximate correct probability is larger than the corresponding approximate incorrect probability for the highest class, i.e., $P_{\text{ELM}}(2|s_1) > P_{\text{ELM}}(3|s_1)$ and $P_{\text{ELM}}(8|s_2) > P_{\text{ELM}}(2|s_2)$, condition (4.3) is satisfied. By *Proposition 4.1*, the correct classification rates for these two samples using V-ELM could go to 100% if the independent training number K is sufficiently large. In the simulations with a total of 2000 trials, each trial is now run by V-ELM instead of ELM. For V-ELM with $K = 19$ at each trial, the misclassification numbers of s_1 and s_2 decrease from 1361 to 800, and from 1055 to 196, respectively. When K is further increased to 77, the corresponding misclassification numbers decrease dramatically to 491 and 0, respectively. On the other hand, for sample s_3 , it goes in the opposite direction stated in *Proposition 4.3* as the approximate correct probability is smaller than the approximate incorrect probability for the highest class, i.e., $P_{\text{ELM}}(5|s_3) < P_{\text{ELM}}(4|s_3)$. As a result, the misclassification number of s_3 increases from 1612 to 1993 for $K = 19$ and further to 2000 for $K = 77$. The comparisons of misclassification numbers between V-ELM and ELM are shown in Table 4.6 for the two cases of $K = 19$ and $K = 77$.

Fig. 4.5 depicts the misclassification numbers of the three samples s_1 , s_2 and s_3 with respect to the independent training number K , ranging from 1 to 120. It illustrates all the three propositions, *Proposition 4.1* to *Proposition 4.3*. In particular, since the difference between $P_{\text{ELM}}(8|s_2)$ and $P_{\text{ELM}}(2|s_2)$ ($\approx 30\%$) is far greater than that between $P_{\text{ELM}}(2|s_1)$ and $P_{\text{ELM}}(3|s_1)$ ($\approx 2.65\%$), it only requires K to be 77 for the classification rate for s_2 to reach 100%, while for s_1 , the required K is much bigger, far beyond 120 as evident from Fig. 4.5. This has verified *Proposition 4.2* clearly.

Part II

Compressed Signal Detection

Due to the powerful capability in handling sparse signal reconstruction, compressed sensing (CS) has attracted much attention in the past several years. Besides developing signal acquiring systems and signal recovery algorithms, recent research on CS also focuses on some fundamental signal processing problems by using the measurement vector directly, such as detection, classification, estimation, and filtering. Instead of the traditional reconstructing-then-processing method, extracting information directly with compressed measurements of the sparse signal in CS could avoid the complex reconstructing process and can guarantee a result which is comparable to the one by processing the original signal.

In part II of the thesis (Chapters 5 and 6), we focus on investigating the detection performance in CS. We first exploit the detection bound of the probability of error for the recovered signal in CS by using the Bayesian approach in Chapter 5. With the prior knowledge of the support set of the sparse signal, we obtain a bound of the probability of error by detecting the signal reconstructed with the oracle estimator. Then, we conduct a detailed investigation on the signal detection problem by processing compressed measurements directly using the Bayesian approach in Chapter 6. We derive a general expression for the probability of error by detecting with compressed measurements directly where the prior probabilities could be equal or unequal and the additive noise is uncorrelated Gaussian noise with possibly unequal variances. Performance bounds for the probability of error using the *restricted isometry property* constant and then the mutual coherence of the sampling matrix are also provided.

Chapter 5

The Detection Bound of the Probability of Error in Compressed Sensing

5.1 Introduction

Detection theory is fundamental to designing decision making system in signal processing problems. Applications of signal detection include radar signal detection [111], sonar signal processing [112], communication system [113], image processing [114], speech signal detection [115] and many others. Two most frequently used methods in signal detection are the *Neyman-Pearson* theorem (NP) [67] and the Bayesian approach [12], respectively, where NP detector is more suitable for signal detection in sonar/radar applications while the Bayesian approach is more appropriate for signal detection in communication systems. These detection systems are mainly founded on the traditional Nyquist/Shannon sampling theory [44, 45]. Re-

cently, a popular topic named Compressed Sensing (CS) was developed for signal sampling and reconstruction [4, 5, 9]. It has been stated in CS that signal $\mathbf{s}$ could be reconstructed within the noise level by linear measurement vector $\mathbf{y}$ sampled with the system $\mathbf{y} = \Phi \mathbf{s} + \mathbf{w}$ where $\Phi \in \mathbf{R}^{m \times n}$, $\mathbf{s} \in \mathbf{R}^n$ and $\mathbf{y} \in \mathbf{R}^m$. The required number of measurements can be less than the one in traditional Nyquist/Shannon sampling system which states the sampling rate should be at least twice of the maximum frequency of the signal. The achievement of the CS sampling framework is based on two aspects: the signal $\mathbf{s}$ is sparse and the sampling system Φ satisfies certain conditions. It has been first stated in CS [5] that to guarantee a satisfactory reconstruction, the sampling matrix Φ should satisfy the *restricted isometry property* (RIP) property. This is similar to requiring that the subspaces spanned by any subdictionaries extracted from Φ differ substantially from one another, where the dimension of the subspace is equivalent to the number of the sparseness of the signal $\mathbf{s}$. In such case, different values of the signal $\mathbf{s}$ could produce different measurement vectors. Different to using the sinc interpolation approach for reconstructing the signal in the Nyquist/Shannon sampling framework, CS achieves the signal recovery by solving a convex optimization problem.

In this chapter, we consider the theoretical detection bound of the probability of error by using the Bayesian approach in CS where the signal is sparse and is reconstructed from a compressed measurement vector. The bound of the probability of error is obtained by utilizing the oracle estimator, which knows the localizations of the nonzero elements of the sparse signal $\mathbf{s}$ [116]. It has been pointed out in [9, 117] that once the nonzero localizations of signal $\mathbf{s}$ is not provided and the estimation is under Gaussian noise, there always exists a $\log m$ gap between the performance of the oracle estimator and the performance of any practical estimator,

and no recovery algorithm would perform better than the result obtained by the oracle estimator. Based on this property and utilizing the Bayesian approach for detection, we propose the theoretical bound of the probability of error in CS where the detection is based on the signal reconstructed from linear measurements and the estimation is under the white Gaussian noise. We compare our result with the detection error obtained by constructing the signal using four recent algorithms: the basis pursuit denoising (BPDN) [48, 57], the Dantzig selector [47], the orthogonal matching pursuit (OMP) method [59] and the compressive sampling matching pursuit (CoSaMP) algorithm [50]. Simulation has been conducted by using these four algorithms and the theoretical bound of the probability of error with Bayesian approach is derived. The result comparisons demonstrate the effectiveness of our proposed bound.

The rest of this chapter is organized as follows. In Sections 5.2 and 5.3, we introduce the Bayesian signal detection method and the oracle estimator in compressed sensing. The detection bound of the probability of error is given in Section 5.4. Section 5.5 shows the simulation results and Section 5.6 draws the conclusion. For convenience of presentation, proof is given in the Appendix.

5.2 Bayesian Signal Detection

A general detection problem is to make decision between two hypotheses as follows

$$\begin{aligned}\mathcal{H}_0 : \mathbf{x} &= \mathbf{w} \\ \mathcal{H}_1 : \mathbf{x} &= \mathbf{s} + \mathbf{w}\end{aligned}\tag{5.1}$$

where $\mathbf{s} \in \mathbf{R}^n$ is a known signal, $\mathbf{w}$ is the Gaussian noise. In the Bayesian approach, the probability of error P_e is defined as

$$P_e = P(\mathcal{H}_0|\mathcal{H}_1)P(\mathcal{H}_1) + P(\mathcal{H}_1|\mathcal{H}_0)P(\mathcal{H}_0). \quad (5.2)$$

where $P(\mathcal{H}_i) = p_i$, $i = 0, 1$, and $P(\mathcal{H}_i|\mathcal{H}_j)$, $i, j = 0, 1 (i \neq j)$, are the prior probability of hypothesis $\mathcal{H}_i$ and the conditional probability, respectively. The detector that minimizes the probability of error P_e is to decide $\mathcal{H}_1$ if

$$\frac{p(\mathbf{x}|\mathcal{H}_1)}{p(\mathbf{x}|\mathcal{H}_0)} > \frac{P(\mathcal{H}_0)}{P(\mathcal{H}_1)} =: \delta, \quad (5.3)$$

where $p(\mathbf{x}|\mathcal{H}_i)$, $i = 0, 1$, is the conditional probability density function. The above test is also called the conditional likelihood ratio test. Then, for equal prior probabilities, the above inequality is equivalent to

$$p(\mathbf{x}|\mathcal{H}_1) > p(\mathbf{x}|\mathcal{H}_0) \quad (5.4)$$

which is also called the *maximum likelihood detector* (ML), where $p(\mathbf{x}|\mathcal{H}_1)$ is the conditional probability density function.

5.3 Oracle Estimator

To obtain a good reconstruction performance of the original sparse signal, several algorithms focus on approximating the localization of the support set for the sparse signal, such as the OMP algorithm and its improvement method CoSaMP algorithm. However, it has been pointed out in [9, 118, 119] that without any

additional information, no recovery algorithm can perform fundamentally better than the oracle estimator in compressed sensing when the estimation is under the Gaussian noise. The oracle estimator is based on the prior knowledge of support T_0 of the signal $\mathbf{s}$. The estimation problem becomes well-posed by calculating the signal $\mathbf{s}$ using the following least squares method

$$\mathbf{s}^\sharp = \begin{cases} (\Phi_{T_0}^T \Phi_{T_0})^{-1} \Phi_{T_0}^T \mathbf{y} & \text{on } T_0 \\ 0 & \text{elsewhere} \end{cases} \quad (5.5)$$

where $\Phi_{T_0} \in \mathbf{R}^{m \times T_0}$ is a matrix whose columns are extracted from Φ according to the index T_0 . Therefore, the theoretic bound of the recovery is

$$\|\mathbf{s}^\sharp - \mathbf{s}\|_2 = \left\| (\Phi_{T_0}^T \Phi_{T_0})^{-1} \Phi_{T_0}^T \mathbf{w} \right\|_2. \quad (5.6)$$

Hence, for the signal $\tilde{\mathbf{s}}$ reconstructed from any recovery algorithm, the error will not be less than the above bound. Specifically, when the noise is Gaussian, the performance obtained by ℓ_1 relaxation algorithms is close to the oracle estimator. It has been shown in [47, 48] that with high probability, the Euclidean distance between the signal $\mathbf{s}$ and the signal $\tilde{\mathbf{s}}$ reconstructed by both the Dantzig algorithm and the BPDN algorithm is within a constant times $\log n$ of the performance of the oracle estimator.

5.4 The Bound of Probability of Error in Compressed Sensing

In this section, we consider the bound of the probability of error P_e in compressed sensing using the maximum likelihood detector. We consider the detection problem between two hypotheses $\mathcal{H}_0 : \mathbf{y} = \mathbf{w}$ and $\mathcal{H}_1 : \mathbf{y} = \Phi \mathbf{s} + \mathbf{w}$ where the signal $\mathbf{s}$ is sparse and is reconstructed from $\mathbf{y}$. For simplicity, we assume that the noise in the detection problem (5.1) in Section 5.2 is WGN with $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ and the prior probabilities of the two hypotheses are equal. Then, according to (5.4), making decision on $\mathcal{H}_1$ is equivalent to the following inequality

$$\frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} e^{-\frac{\|\mathbf{x}-\mathbf{s}\|_2^2}{2\sigma^2}} > \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} e^{-\frac{\|\mathbf{x}\|_2^2}{2\sigma^2}} \quad (5.7)$$

or

$$\|\mathbf{x} - \mathbf{s}\|_2^2 < \|\mathbf{x}\|_2^2. \quad (5.8)$$

Suppose the signal $\mathbf{x}$ is a k -sparse signal and is reconstructed from the measurement vector $\mathbf{y} = \Phi \mathbf{s} + \mathbf{w}$. We have the following theorem.

Theorem 5.1. (*Appendix 5.7*) Assume the signal $\tilde{\mathbf{s}}$ is reconstructed by any recovery algorithm using the compressed measurements $\mathbf{y} = \Phi \mathbf{s} + \mathbf{w}$, where $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ is a white Gaussian noise, $\mathbf{s}$ is a k -sparse signal and the sampling matrix Φ satisfies the RIP property. The probability of error P_e using the Bayesian approach to detect the signal which is reconstructed with any recovery algorithm in CS is always larger than the following bound

$$\frac{2\sigma \sqrt{\lambda_{\min, A}} e^{-\frac{\|\mathbf{s}\|_2^2}{8\sigma^2 \lambda_{\min, A}}}}{\sqrt{2\pi} \|\mathbf{s}\|_2} \quad (5.9)$$

where $A = (\Phi_{T_0}^T \Phi_{T_0})^{-1}$.

This theorem states that in the detection problem, when the signal is sparse and is reconstructed by recovery algorithms using compressed measurements, the probability of error obtained by Bayesian approach is always larger than the bound given in (5.9) when the noise is WGN.

Remark 5.1. *From the result given in (5.9), we can find that the bound of the probability of error obtained by Bayesian approach is related to the signal-to-noise ratio $SNR = \frac{\|\mathbf{s}\|_2^2}{\sigma^2}$ and the minimal eigenvalue of matrix A . The larger the SNR is, the smaller the bound is. The smaller the minimal eigenvalue of matrix A is, the smaller the bound of the probability of error tends to have. This result is also the same to the one in real application because the larger the SNR in real application is, the smaller the probability of error will be.*

For eigenvalues of the matrix $A = (\Phi_{T_0}^T \Phi_{T_0})^{-1}$, we have the following properties in terms of the RIP constant of the sampling matrix Φ .

Lemma 5.1. *(Appendix 5.8) Assumed that the matrix Φ satisfies the restricted isometry property for sparse signal $\mathbf{s}$ with parameter $\delta \in (0, 1)$, then, the eigenvalues of matrix $\Phi_{T_0}^T \Phi_{T_0}$ are within the range $[1 - \eta, 1 + \eta]$, where $\Phi_{T_0} \in \mathbf{R}^{m \times T_0}$ is a matrix whose columns are extracted from Φ according to the index T_0 and T_0 is the index vector of the nonzero elements in signal $\mathbf{s}$.*

With Lemma 5.1, we can easily find that the eigenvalues of the matrix A are within the region $\left[\frac{1}{1+\eta}, \frac{1}{1-\eta}\right]$. Therefore, we can further extend the bound of the probability of error in (5.9) as follows

$$P_e \geq \frac{2\sigma e^{-\frac{\|\mathbf{s}\|_2^2(1+\eta)}{8\sigma^2}}}{\sqrt{2\pi(1+\eta)} \|\mathbf{s}\|_2} \quad (5.10)$$

5.5 Simulations

In the simulation, we compare our theoretical bound of the probability of error derived in Section 5.4 with the detection error obtained by detecting the sparse reconstructed from linear measurements with four recent algorithms: BPDN, Dantzig, OMP, and CoSaMP.

5.5.1 Simulation set-ups

A sparse signal $\mathbf{x}$ with dimension of 1024 is tested in the simulation. For this signal, 102 random nonzero spikes are generated and the remainders are set to be 0. The sampling matrix Φ with 512 measurements is generated where the entries of the sampling matrix are identically and independently sampled from a standard normal distribution and then orthogonalized. Different noise variance values σ^2 varying from 0.25 to 0.35 with interval 0.01 are used to generate the white Gaussian noise. For the Dantzig selector, τ is set to be 3×10^{-3} while in the BPDN method, γ is set to be 1×10^{-4} in the simulation. The sparsity of the approximation in OMP and CoSaMP is set to the cardinality of the support set of the sparse signal. For each variance value of the noise, 1000 trials of simulations are conducted. The average bound of the probability of error using (5.9) with different signal powers and minimal eigenvalues of the measurement matrix at different trials is given and the average results for the four algorithms are recorded.

5.5.2 Comparison of the results

Fig. 5.1 depicts the curves of the theoretical bound of the probability of error derived by (5.9) given in Theorem 5.1, and detection errors obtained by detecting

Figure 5.1: Comparisons of the detection performance by detecting the signal reconstructed with four recovery algorithms (BPDN, Dantzig, OMP, CoSaMP) and theoretical bound with respect to different variance σ^2 values.

the signal reconstructed with four algorithms. From this figure, one can easily find that the probability of errors obtained by detecting the signal reconstructed with the four used algorithms are always higher than the corresponding theoretical bound. As shown in this figure, the probability of error obtained by the CoSaMP algorithm is the smallest among all these four algorithms. This comparison demonstrate our result given in Theorem 5.1, which states that the probability of error P_e using the Bayesian approach to detect the signal reconstructed by any recovery algorithm is always larger than the theoretical bound. Furthermore, we can also find that the theoretical bound of the probability of error increases when the signal-to-noise ratio decreases. This result also validates our Remark 5.1.

5.6 Conclusions

In this chapter, we have obtained the theoretical detection bound of the probability of error by using the Bayesian approach in compressed sensing where the signal is sparse and is reconstructed from a compressed measurement vector. Based on the oracle estimator in CS, the theoretical bound of the detection error has been provided when the noise in the estimation of CS is white Gaussian noise. Without any additional information in CS, we have shown in this chapter that the probability of error obtained using the signal reconstructed by any recovery algorithm is always larger than the theoretical bound. We have compared our result with the detection error obtained by using the signal reconstructed with four recovery algorithms: BPDN algorithm, the Dantzig selector, the OMP method and the CoSaMP algorithm. Simulations have been conducted to demonstrate the effectiveness of our result.

5.7 Appendix: Proof of Theorem 5.1

As we can see from the *maximum likelihood detector*, deciding that the test belongs to $\mathcal{H}_1$ is equivalent to ensuring inequality (5.8). This is equivalent to that the reconstructed signal $\tilde{\mathbf{s}}$ has to satisfy the following inequality

$$\|\tilde{\mathbf{s}} - \mathbf{s}\|_2^2 < \|\tilde{\mathbf{s}}\|_2^2, \quad (5.11)$$

or

$$\|\mathbf{s}\|_2^2 + 2\mathbf{s}^T(\tilde{\mathbf{s}} - \mathbf{s}) > 0 \quad (5.12)$$

As shown in Section 5.3, no recovery algorithm can have a better performance than the one derived by the oracle estimator. Therefore, we will start the analysis from this fundamental point. As we have shown in the beginning of this section, the correct decision is to ensure inequality (5.8). The error in detection will be made if we get $\|\mathbf{x} - \mathbf{s}\|_2^2 > \|\mathbf{x}\|_2^2$ during the detection process. Based on the oracle estimator given in Section 5.3, the decision that the error detection is made is equivalent to having the following inequality

$$(\mathbf{s})_{T_0}^T (\Phi_{T_0}^T \Phi_{T_0})^{-1} \Phi_{T_0}^T \mathbf{w} < -\frac{\|\mathbf{s}\|_2^2}{2} \quad (5.13)$$

where $(\mathbf{s})_{T_0} \in \mathbf{R}^{T_0}$ is extracted from the original signal $\mathbf{s}$ according to the index T_0 and T_0 is the index vector of the nonzero elements in signal $\mathbf{s}$. Since $(\tilde{\mathbf{s}} - \mathbf{s}) = \mathbf{0}$ on the complement of T_0 , the above inequality (5.13) is derived by inequality (5.12) and the following equation

$$\mathbf{s}^T(\tilde{\mathbf{s}} - \mathbf{s}) = (\mathbf{s})_{T_0}^T(\tilde{\mathbf{s}} - \mathbf{s})_{T_0}$$

The left hand side of inequality (5.13) obeys the Gaussian distribution with

$$\mathcal{N}\left(\mathbf{0}, \sigma^2 \left\| \Phi_{T_0} (\Phi_{T_0}^T \Phi_{T_0})^{-1} (\mathbf{s})_{T_0} \right\|_2^2\right). \quad (5.14)$$

Rewrite the mean center as

$$\left\| \Phi_{T_0} (\Phi_{T_0}^T \Phi_{T_0})^{-1} (\mathbf{s})_{T_0} \right\|_2^2 = (\mathbf{s})_{T_0}^T \mathbf{A} (\mathbf{s})_{T_0} \quad (5.15)$$

where $\mathbf{A} = (\Phi_{T_0}^T \Phi_{T_0})^{-1} \in \mathbf{R}^{T_0 \times T_0}$ is a real symmetric matrix. Therefore, the probability of error P_e can be calculated as

$$\begin{aligned} P_e &= \mathbf{P}(\|\mathbf{x} - \mathbf{s}\|_2^2 > \|\mathbf{x}\|_2^2) \\ &= \mathbf{P}\left((\mathbf{s})_{T_0}^T (\Phi_{T_0}^T \Phi_{T_0})^{-1} \Phi_{T_0}^T \mathbf{w} < -\frac{\|\mathbf{s}\|_2^2}{2}\right). \end{aligned} \quad (5.16)$$

Since the left part in (5.16) satisfies the Gaussian distribution as shown in (5.14), we have

$$P_e = Q\left(\frac{\|\mathbf{s}\|_2^2}{2\sigma\sqrt{(\mathbf{s})_{T_0}^T \mathbf{A} (\mathbf{s})_{T_0}}}\right). \quad (5.17)$$

An approximation that is sometimes used for the complementary cumulative distribution function $Q(x)$ is [68].

$$Q(x) \approx \frac{1}{\sqrt{2\pi}x} e^{-\frac{1}{2}x^2}. \quad (5.18)$$

From (5.16) and the above approximation of $Q(x)$, we get the following approximation of the probability of error P_e

$$P_e \approx \frac{2\sigma \sqrt{(\mathbf{s})_{T_0}^T \mathbf{A} (\mathbf{s})_{T_0}} e^{-\frac{\|\mathbf{s}\|_2^4}{8\sigma^2 (\mathbf{s})_{T_0}^T \mathbf{A} (\mathbf{s})_{T_0}}}}{\sqrt{2\pi} \|\mathbf{s}\|_2^2} \quad (5.19)$$

As $\mathbf{A}$ is a real symmetric matrix, it can be decomposed using the eigenvalue decomposition as $\mathbf{A} = \mathbf{V}_A \mathbf{D}_A \mathbf{V}_A^T$ where $\mathbf{V}_A$ is an orthogonal matrix (the columns of which are eigenvectors of $\mathbf{A}$), and $\mathbf{D}_A = \text{diag} \{\lambda_1, \lambda_2, \dots, \lambda_n\}$ is real and diagonal [120]. Therefore, we have

$$\lambda_{\min, A} \|\mathbf{V}_A^T (\mathbf{s})_{T_0}\|_2^2 \leq (\mathbf{s})_{T_0}^T \mathbf{A} (\mathbf{s})_{T_0} \leq \lambda_{\max, A} \|\mathbf{V}_A^T (\mathbf{s})_{T_0}\|_2^2$$

As $\mathbf{V}_A$ is an orthogonal matrix and for the original signal $\mathbf{s}$: $\|(\mathbf{s})_{T_0}\|_2^2 = \|\mathbf{s}\|_2^2$, we get the following equivalent inequality

$$\lambda_{\min, A} \|\mathbf{s}\|_2^2 \leq (\mathbf{s})_{T_0}^T \mathbf{A} (\mathbf{s})_{T_0} \leq \lambda_{\max, A} \|\mathbf{s}\|_2^2 \quad (5.20)$$

Using the above inequality (5.20), we can derive the following approximation inequality from inequality (5.19) that

$$P_e \geq \frac{2\sigma \sqrt{\lambda_{\min, A}} e^{-\frac{\|\mathbf{s}\|_2^2}{8\sigma^2 \lambda_{\min, A}}}}{\sqrt{2\pi} \|\mathbf{s}\|_2} \quad (5.21)$$

This finishes the proof.

5.8 Appendix: Proof of Lemma 5.1

With the prior information of the support T_0 of the nonzero elements in signal $\mathbf{s}$, we have

$$\Phi \mathbf{s} = \Phi_{T_0} \mathbf{s}_{T_0} \quad (5.22)$$

where $\mathbf{s}_{T_0} \in \mathbf{R}^{T_0}$ is extracted from the original signal $\mathbf{s}$ according to the index T_0 . Therefore,

$$\|\Phi_{T_0} \mathbf{s}_{T_0}\|_2^2 = \|\Phi \mathbf{s}\|_2^2 \quad (5.23)$$

and

$$\lambda_{\min, \Phi_{T_0}^T \Phi_{T_0}} \|\mathbf{s}\|_2^2 \leq \|\Phi_{T_0} \mathbf{s}_{T_0}\|_2^2 \leq \lambda_{\max, \Phi_{T_0}^T \Phi_{T_0}} \|\mathbf{s}\|_2^2 \quad (5.24)$$

as $\|\mathbf{s}_{T_0}\|_2^2 = \|\mathbf{s}\|_2^2$. Since the matrix Φ satisfies the restrict isometry property with parameter $\eta \in (0, 1)$, from (1.12), we have

$$\begin{cases} \lambda_{\max, \Phi_{T_0}^T \Phi_{T_0}} \leq 1 + \eta \end{cases} \quad (5.25a)$$

$$\begin{cases} \lambda_{\min, \Phi_{T_0}^T \Phi_{T_0}} \geq 1 - \eta \end{cases} \quad (5.25b)$$

This finishes our proof.

Chapter 6

Bayesian Signal Detection with Compressed Measurements

6.1 Introduction

In this chapter, we focus on investigating signal detection with the Bayesian approach by using compressed measurements directly.

Compressed sensing (CS) for sparse signal sampling and reconstruction has attracted much attention in the past decade, see, e.g., [4–6, 9, 46–51]. CS has emerged as a protocol of signal acquisition by directly capturing a compressed signal representation instead of using the traditional sampled-then-compressed framework. While most of the research effort in the CS community has been focused on signal recovery, it was shown recently that signal recovery from compressed measurements may not be necessary in some signal processing problems, such as detection, classification and filtering [60–66]. In particular, in signal detection, it was first pointed out in [60] that it was possible to carry out signal detection based on partial sig-

nal reconstruction using CS projections. In [63], the idea of detection using CS measurements was extended to more general classes of signals. A comprehensive discussion on compressive signal detection and other compressive signal processing problems was recently given in [65, 66]. Specifically, instead of first reconstructing the original signal from the compressive measurements and then performing signal processing such as detection, an alternative approach was proposed to directly processing the signals in the compressive domain [65, 66]. This novel idea has opened the door for new research activities, including our work on signal detection with compressed measurements using the Bayesian approach to be presented in this chapter.

While pioneering work on signal detection with compressed measurements was given in [60, 63, 65, 66], the approach taken was based on the *Neyman-Pearson* theorem (NP) [67, 68] and the noise in consideration is limited to white Gaussian noise (WGN). Although NP detector is an optimal and powerful tool in signal detection, it is more suitable for signal detection in sonar/radar applications. In some other applications such as communications, Bayesian approach is often adopted for tackling the signal detection problem [12, 68]. The ability of incorporating prior knowledge into detection is one of the main advantages of the Bayesian approach. With this background, we adopt the Bayesian approach to signal detection with compressed measurements in this chapter. The emphasis of our work is on the influence of prior probabilities of the hypotheses on the probability of error in signal detection and on the issue of computing the probability of error and its bounds. We also consider the more general noise which is uncorrelated Gaussian noise with possibly unequal variances. Note that the work presented here is new since the Bayesian approach to signal detection in [116] considered the detection problem

using reconstructed signals while the Bayesian theory for CS was concerned with the signal estimation problem in [121–123].

The organization of this chapter is as follows. In Section 6.2, we start with a brief review of the Bayesian approach to signal detection. We also present a new result on the influence of prior probabilities of the hypotheses on the probability of error in signal detection. Specifically, we show that for a given signal with a fixed signal to noise ratio (SNR), the probability of error with unequal prior probabilities in hypotheses is always smaller than the one with equal prior probabilities in hypotheses. In Section 6.3, we present several new results on signal detection with compressed measurements using the Bayesian approach. First, we give an expression of the probability of error in detecting a signal using compressed measurements. Similar to the result presented in Section 6.2, we show that with compressed measurements, the probability of error with unequal prior probabilities in hypotheses is always smaller than the one with equal prior probabilities in hypotheses. Moreover, bounds of the probability of error are given based on the *restricted isometry property* (RIP) and the mutual coherence of the sampling matrix. Secondly, for a sufficiently small difference between the two prior probabilities of hypotheses, a simple expression to calculate the probability of error is derived. Thirdly, the detection bound of the probability of error by adopting a piecewise function is also presented in this section. Simulations are conducted in Section 6.4 to illustrate the theoretical results presented in this chapter and conclusions are shown in Section 6.5. For convenience of presentation, proofs are given in the Appendix.

6.2 General Detection Performance Analysis with Prior Probabilities

In this section, we investigate the influence of prior probabilities of the hypotheses on the probability of error in general signal detection with the Bayesian approach.

By taking a natural logarithm, an equivalent test T of (5.3) in Section 5.2 of Chapter 5 which decides $\mathcal{H}_1$ is given by

$$T := \mathbf{x}^T C_{\mathbf{w}}^{-1} \mathbf{s} > \ln(\delta) + \frac{1}{2} \mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s} =: \gamma, \quad (6.1)$$

where $C_{\mathbf{w}}$ is the covariance matrix of the noise. We note that the probability density function (PDF) of the test T can be written as

$$T \sim \begin{cases} \mathcal{N}(\mathbf{0}, \mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}), & \mathcal{H}_0 \\ \mathcal{N}(\mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}, \mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}), & \mathcal{H}_1 \end{cases} \quad (6.2)$$

Thus, the probability of error P_e becomes

$$P_e = p_0 Q\left(\frac{\gamma}{\sqrt{\mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}}}\right) + p_1 Q\left(\frac{\mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s} - \gamma}{\sqrt{\mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}}}\right) \quad (6.3)$$

where $Q(x) := \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du$, p_0 and p_1 are the prior probabilities of hypotheses $\mathcal{H}_0$ and $\mathcal{H}_1$ in (5.1), respectively. The above discussion is well known and can be found in the literature, e.g., [68]. When $p_0 = p_1 = \frac{1}{2}$, properties of P_e have been well explored. However, when $p_0 \neq p_1$, less attention has been paid to P_e .

Since $p_0 + p_1 = 1$, P_e in expression (6.3) can be equivalently rewritten as

$$P_e = Q\left(\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2}\right) + \kappa \quad (6.4)$$

where $\kappa := \frac{p_1}{\sqrt{2\pi}} \int_{\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2} - \xi}^{\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2}} e^{-\frac{u^2}{2}} du - \frac{p_0}{\sqrt{2\pi}} \int_{\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2}}^{\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2} + \xi} e^{-\frac{u^2}{2}} du$, $\xi := \frac{\ln p_0 - \ln p_1}{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}$.

Note that $\kappa = 0$ when $p_0 = p_1 = \frac{1}{2}$ and hence κ arises only when $p_0 \neq p_1$. For convenience of expression, we denote $\varepsilon := p_0 - p_1$ in the remainder of the chapter. Note that $-1 \leq \varepsilon \leq 1$.

Analyzing the probability of error P_e carefully, we discover the following new properties of P_e .

Theorem 6.1. (*Appendix 6.6*) *Consider the detection problem in (5.1), we have*

1. *For a fixed $\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2}$ and $\varepsilon \neq 0$, then, $\kappa < 0$ and P_e decreases when $|\varepsilon|$ increases.*
2. *For a fixed ε , P_e decreases when $\frac{\sqrt{\mathbf{s}^T \mathbf{C}_{\mathbf{w}}^{-1} \mathbf{s}}}{2}$ increases.*

To our best knowledge, other than the case when $\varepsilon = 0$ in result 2), Theorem 6.1 is new and the results obtained will be further extended to signal detection with compressed measurements.

From result 1) of Theorem 6.1, $\kappa < 0$ means that when the signal and the covariance matrix of the noise are fixed, the probability of error with unequal prior probabilities of the two hypotheses is always smaller than that with equal prior probabilities. Moreover, result 1) of Theorem 6.1 also shows that the probability of error P_e decreases when the difference between the prior probabilities of the two hypotheses increases. This is intuitively clear since unequal prior probabilities

means that more information is available than that with equal prior probabilities. In particular, when the prior probability of one hypothesis reaches its limit of 1, the probability of error P_e tends to 0 since there is no uncertainty in signal detection in such a special case. As $\frac{\sqrt{\mathbf{s}^T \mathbf{C}_w^{-1} \mathbf{s}}}{2}$ may be interpreted as the signal to noise ratio (SNR), result 2) of Theorem 6.1 means that for a fixed difference between the prior probabilities of the two hypotheses, P_e will decrease when $\frac{\sqrt{\mathbf{s}^T \mathbf{C}_w^{-1} \mathbf{s}}}{2}$ increases. This result may be viewed as a generalization of the known result on the probability of error P_e for the special case with equal prior probability to the case when the two prior probabilities are unequal.

6.3 Detection with Compressed Measurements

In this section, the probability of error by detecting the signal directly using compressed measurements with the Bayesian theory is first presented. We then discuss the bounds for the probability of error with compressed measurements in terms of the *restricted isometry property* (RIP) as well as the computationally more tractable mutual coherence property of a sampling matrix. An approximate expression of the probability of error which is simple and easy to calculate is derived when the difference between the prior probabilities is small. Moreover, an approximate but simple upper bound for the probability of error in terms of a piecewise function is also obtained.

6.3.1 Signal detection with compressed measurements

Instead of reconstructing the original signal $\mathbf{s}$ and then carrying out signal detection [116], we consider the signal detection problem by directly processing the

compressed measurements as follows [66].

$$\begin{aligned}\tilde{\mathcal{H}}_0 : \mathbf{y} &= \Phi \mathbf{w} \\ \tilde{\mathcal{H}}_1 : \mathbf{y} &= \Phi(\mathbf{s} + \mathbf{w})\end{aligned}\tag{6.5}$$

where $\mathbf{s} \in \mathbb{S}_k$ is a k -sparse signal, $\mathbf{w}$ is an uncorrelated Gaussian noise with unequal variances, i.e., $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, C_{\mathbf{w}})$ with $C_{\mathbf{w}} = \text{diag}\{\sigma_{\max}^2, \sigma_2^2, \dots, \sigma_{\min}^2\}$, $\Lambda = C_{\mathbf{w}}^{\frac{1}{2}}$ and $\sigma_{\max} \geq \sigma_2 \geq \dots \geq \sigma_{\min} > 0$.

Suppose the prior probabilities of these two hypotheses are $P(\tilde{\mathcal{H}}_0) := p_0 \geq 0$ and $P(\tilde{\mathcal{H}}_1) := p_1 \geq 0$, respectively, with $p_0 + p_1 = 1$ and p_0 and p_1 not necessarily being equal. Similar to the discussion in Section 6.2, the probability of error $\tilde{P}_e$ by directly using compressed measurements is defined as

$$\tilde{P}_e = P\left(\tilde{\mathcal{H}}_0 \mid \tilde{\mathcal{H}}_1\right) P(\tilde{\mathcal{H}}_1) + P\left(\tilde{\mathcal{H}}_1 \mid \tilde{\mathcal{H}}_0\right) P(\tilde{\mathcal{H}}_0)\tag{6.6}$$

where $P\left(\tilde{\mathcal{H}}_i \mid \tilde{\mathcal{H}}_j\right)$, $i, j = 0, 1 (i \neq j)$, is the conditional probability. Then, the detector that minimizes the probability of error $\tilde{P}_e$ is to decide $\tilde{\mathcal{H}}_1$ if

$$\frac{p(\mathbf{y} \mid \tilde{\mathcal{H}}_1)}{p(\mathbf{y} \mid \tilde{\mathcal{H}}_0)} > \frac{P(\tilde{\mathcal{H}}_0)}{P(\tilde{\mathcal{H}}_1)} := \tilde{\delta},\tag{6.7}$$

where $p(\mathbf{y} \mid \tilde{\mathcal{H}}_i)$, $i = 0, 1$, is the conditional probability density function. Assuming that $\Phi C_{\mathbf{w}} \Phi^T$ is invertible and taking the natural logarithm, an equivalent test $\tilde{T}$ is given by as

$$\tilde{T} := \mathbf{y}^T (\Phi C_{\mathbf{w}} \Phi^T)^{-1} \Phi \mathbf{s} > \ln(\tilde{\delta}) + \frac{1}{2} \mathbf{s}^T \Phi^T (\Phi C_{\mathbf{w}} \Phi^T)^{-1} \Phi \mathbf{s} := \gamma\tag{6.8}$$

The following theorem contains general results on signal detection with compressed measurements.

Theorem 6.2. (Appendix 6.7) Consider the detection problem given in (6.5). Assume that the matrix $\sqrt{n/m}\mathbb{H}_{C_w}$ satisfies the RIP with parameter $\eta \in (0, 1)$ where $\mathbb{H}_{C_w} = (\Phi\Lambda)^T ((\Phi\Lambda)(\Phi\Lambda)^T)^{-1} (\Phi\Lambda)$. We have

1. The probability of error $\tilde{P}_e$ in detecting the signal $\mathbf{s}$ using compressed measurements is

$$\tilde{P}_e = Q\left(\frac{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}{2}\right) + \tilde{\kappa}. \quad (6.9)$$

where

$$\tilde{\kappa} := \frac{p_1}{\sqrt{2\pi}} \int_{\frac{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}{2} - \tilde{\xi}}^{\frac{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}{2}} e^{\left(-\frac{u^2}{2}\right)} du - \frac{p_0}{\sqrt{2\pi}} \int_{\frac{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}{2}}^{\frac{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}{2} + \tilde{\xi}} e^{\left(-\frac{u^2}{2}\right)} du, \quad (6.10)$$

and

$$\tilde{\xi} := \frac{\ln p_0 - \ln p_1}{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}. \quad (6.11)$$

2. For a fixed $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ and $\varepsilon \neq 0$, then, $\tilde{\kappa} < 0$ and $\tilde{P}_e$ decreases when $|\varepsilon|$ increases.
3. For a fixed ε , $\tilde{P}_e$ decreases when $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ increases.
4. The probability of error $\tilde{P}_e$ satisfies the following bounds

$$Q\left(\sqrt{1+\eta}\sqrt{\frac{m}{n}}\frac{\|\mathbf{s}\|_2}{2\sigma_{\min}}\right) + \tilde{\kappa} \leq \tilde{P}_e \leq Q\left(\sqrt{1-\eta}\sqrt{\frac{m}{n}}\frac{\|\mathbf{s}\|_2}{2\sigma_{\max}}\right) + \tilde{\kappa} \quad (6.12)$$

Result 1) of Theorem 6.2 gives a general expression for the probability of error by detecting the signal directly with compressed measurements. The detection performance in (6.9) is related to $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ which is dependent on the matrix $\tilde{\Phi} := \Phi\Lambda$ if the signal $\mathbf{s}$ and the covariance matrix of the noise are fixed. Generally, the larger $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ is, the smaller the probability of error is. Moreover, since $\mathbb{H}_{C_w}$ is the orthogonal projection operator on the row space of $\tilde{\Phi}$, the norm $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ is within $[0, \|\Lambda^{-1}\mathbf{s}\|_2]$. Compared with the probability of error by using the original signal shown in (6.3), the probability of error with compressed measurements could be the same as the one with the original signal if $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ approaches $\|\Lambda^{-1}\mathbf{s}\|_2$. However, the probability of error with compressed measurements could also be much larger than the one with the original signal if $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ is far less than $\|\Lambda^{-1}\mathbf{s}\|_2$.

Results 2) and 3) of Theorem 6.2 are analogous to results 1) and 2) of Theorem 6.1. Specifically, result 2) of Theorem 6.2 shows that detecting the signal directly with compressed measurements, the probability of error with equal prior probability of the two hypotheses is larger than the one with unequal prior probabilities when the signal and the covariance matrix of the noise are fixed, and that the probability of error decreases when the difference between prior probabilities increases. Result 3) of Theorem 6.2 states that when the prior probabilities of the two hypotheses are fixed, the probability of error with compressed measurements decreases when $\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2$ increases.

The performance bounds of the probability of error in compressed domain presented in result 4) of Theorem 6.2 tell us that detecting the signal by using compressed measurements directly, the probability of error is within the range whose upper bound and lower bound are related to the noise covariance matrix C_w and

the signal power $\|\mathbf{s}\|_2^2$. Given a known noise covariance matrix, the upper and lower probability of error bounds are mainly dependent on the power of the signal. The larger the signal power is, the smaller the bound of the probability of error tends to have. Specifically, for a sufficiently small η , the probability of error $\tilde{P}_e$ is approximately within the range $Q\left(\sqrt{\frac{m}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma_{\min}}\right) \leq \tilde{P}_e \leq Q\left(\sqrt{\frac{m}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma_{\max}}\right)$ when $p_0 = p_1$. This range is close to the one when directly using the original signal $\mathbf{s}$ in detection. Therefore, as pointed out in [66], how precise a detection result we could get by directly using random measurements is dependent on the ability to solve a detection problem but not on the recovery capability in CS.

Remark 6.1. We now explain why the matrix $\sqrt{n/m}\mathbb{H}_{C_{\mathbf{w}}}$ in Theorem 6.2 satisfies the RIP with parameter η if the covariance matrix of the Gaussian noise is restricted to a diagonal matrix but with possibly unequal diagonal elements. Since $C_{\mathbf{w}} = \Lambda^2$ is known and Φ is given, $\mathbb{H}_{C_{\mathbf{w}}}$ could be decomposed as $(\Phi\Lambda)^T \left((\Phi\Lambda)(\Phi\Lambda)^T \right)^{-1} (\Phi\Lambda)$. It is known [66] that the matrix $\sqrt{n/m}\mathbb{H}_{C_{\mathbf{w}}}$ has the RIP for sparse signals $\mathbf{s} \in \mathcal{S}_k$ if the matrix $\tilde{\Phi} = \Phi\Lambda$ is generated according to certain rules. For example, one could design the sampling matrix $\tilde{\Phi}$ with the property of $\Phi C_{\mathbf{w}} \Phi^T = \tilde{\Phi} \tilde{\Phi}^T = \mathbf{I}$, i.e., $\tilde{\Phi}$ is a random orthogonal projection matrix. Then, $\|\mathbb{H}_{C_{\mathbf{w}}} \Lambda^{-1} \mathbf{s}\|_2 = \|\tilde{\Phi} \Lambda^{-1} \mathbf{s}\|_2 = \|\tilde{\Phi} \tilde{\mathbf{s}}\|_2$ with $\tilde{\mathbf{s}} = \Lambda^{-1} \mathbf{s}$. Hence, the matrix $\sqrt{n/m}\mathbb{H}_{C_{\mathbf{w}}}$ satisfies the RIP with probability 1 [124] and the measurement equation is equivalently given by

$$\mathbf{y} = \tilde{\Phi} \tilde{\mathbf{s}} + \tilde{\Phi} \Lambda^{-1} \mathbf{w}. \quad (6.13)$$

It is obvious that the transformed signal $\tilde{\mathbf{s}}$ still preserves the same sparsity of $\mathbf{s}$ and the reconstruction of the signal $\mathbf{s}$ can be easily obtained by multiplying $\tilde{\mathbf{s}}$ with Λ . Another example is that each element in $\tilde{\Phi}$ is generated with i.i.d. zero mean

Gaussian distribution [66]. In such case, $\sqrt{n/m}\mathbb{H}_{C_{\mathbf{w}}}$ also has the RIP as the distribution of $\|\tilde{\Phi}\tilde{\mathbf{s}}\|_2$ is the same as the one where the matrix $\tilde{\Phi}$ is a random orthogonal projection matrix. Finally, we point out that we are unable to generalize Theorem 6.2 to the case when the noise covariance matrix is not a diagonal matrix. This can be seen by observing that the transformed signal $\tilde{\mathbf{s}} = D^{-\frac{1}{2}}E\mathbf{s}$ is in general not a sparse signal, where $C_{\mathbf{w}} = E^{-1}DE$ is the eigendecomposition of the matrix $C_{\mathbf{w}}$, D is the diagonal matrix whose diagonal elements are the corresponding eigenvalues and E is a unitary matrix whose rows are orthonormal eigenvectors of $C_{\mathbf{w}}$.

We next consider the computational issue for the bounds of probability of error $\tilde{P}_e$ given in (6.12). Generally, for a given sampling matrix Φ , it has been shown in [9, 125] that it is NP-hard problem to evaluate its RIP. That is, for a given matrix, there are no efficient algorithms to calculate the RIP constant η . Thus, a suboptimal constraint called the mutual coherence of the matrix Φ which is easier to calculate than the RIP constant η is usually used when the RIP constant η of the matrix Φ is unknown [126, 127]. For convenience of expression, we assume in the following that each column of Φ has unit ℓ_2 norm.

Definition 1. *The mutual coherence μ of a matrix Φ is defined as*

$$\mu \triangleq \max_{i \neq j} |\phi_i^T \phi_j| \quad (6.14)$$

where ϕ_i and ϕ_j are the i -th and j -th columns of the matrix Φ , respectively.

The relationship of the mutual coherence μ and the RIP of the matrix Φ is discussed in [127]. Exploiting the mutual coherence and recalling the detection bounds in the measurement domain in Theorem 6.2, we next present alternative detection bounds of the probability of error in the measurement domain.

Theorem 6.3. (Appendix 6.8) Consider the same detection problem as in Theorem 6.2. Assume that the mutual coherence of the matrix $\mathbb{H}_{C_w}$ is μ . Then, for any $\mathbf{s} \in \mathbb{S}_k$ with $k < \frac{1}{2} \left(\frac{1}{\mu} + 1 \right)$, the probability of error $\tilde{P}_e$ satisfies the following bounds

$$Q \left(\sqrt{\frac{m[1 + (k-1)\mu]}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma_{\min}} \right) + \tilde{\kappa} \leq \tilde{P}_e \leq Q \left(\sqrt{\frac{m[1 - (k-1)\mu]}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma_{\max}} \right) + \tilde{\kappa} \quad (6.15)$$

The significance of Theorem 6.3 is that for a given sampling matrix Φ , the mutual coherence of Φ can be easily calculated. Assuming that the sparsity of the signal satisfies $k < \frac{1}{2} \left(\frac{1}{\mu} + 1 \right)$, the bounds for the probability of error can be easily calculated by (6.15). However, to calculate the bounds by Theorem 6.2 is computationally much more difficult because computing the RIP constant η is NP-hard.

Since both [66] and this chapter discuss the signal detection problem with compressed measurements, it is natural to ask whether there is a connection between the results presented in this chapter with the corresponding results in [66]. The answer is yes after we restrict our obtained results to the special case where the noise is White Gaussian and the prior probabilities of the two hypotheses are equal. Specially, we can easily derive the following corollary from Theorems 6.2 and 6.3.

Corollary 6.1. Consider the detection problem given in (6.5) where prior probabilities of the two hypotheses are equal, i.e., $p_0 = p_1 = \frac{1}{2}$, and the noise is White Gaussian noise, i.e., $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$. Then, we have

1. The probability of error $\tilde{P}_e$ in detecting the signal $\mathbf{s}$ using compressed measurements is

$$\tilde{P}_e = Q \left(\frac{\|\mathbb{H}\mathbf{s}\|_2}{2\sigma} \right). \quad (6.16)$$

-
2. Assume that $\sqrt{n/m}\mathbb{H}$ satisfies the RIP with parameter $\eta \in (0, 1)$ where $\mathbb{H} := \Phi^T(\Phi\Phi^T)^{-1}\Phi$, the probability of error $\tilde{P}_e$ satisfies the following bounds

$$Q\left(\sqrt{\frac{m(1+\eta)}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma}\right) \leq \tilde{P}_e \leq Q\left(\sqrt{\frac{m(1-\eta)}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma}\right). \quad (6.17)$$

3. Assume that the mutual coherence of matrix $\mathbb{H}$ is μ , the probability of error $\tilde{P}_e$ satisfies

$$Q\left(\sqrt{\frac{m[1+(k-1)\mu]}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma}\right) \leq \tilde{P}_e \leq Q\left(\sqrt{\frac{m[1-(k-1)\mu]}{n}} \frac{\|\mathbf{s}\|_2}{2\sigma}\right). \quad (6.18)$$

Corollary 6.1 shows that when η is sufficiently small, we get the approximate probability of error $\tilde{P}_e \approx Q\left(\sqrt{\frac{m}{n}} \frac{\sqrt{\text{SNR}}}{2}\right)$ where $\text{SNR} = \frac{\|\mathbf{s}\|_2^2}{\sigma^2}$. The probability of error $\tilde{P}_e$ by using compressed measurements decreases when the SNR and/or $\frac{m}{n}$ increases.

Remark 6.2. For the same detection problem in Corollary 6.1, the following results were presented in [66] on the probability of detection $\tilde{P}_D$ using the Neyman-Pearson (NP) detector.

- 1.

$$\tilde{P}_D = Q\left(Q^{-1}(\alpha) - \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma}\right) \quad (6.19)$$

2. $\tilde{P}_D$ satisfies

$$Q\left(Q^{-1}(\alpha) - \sqrt{\frac{m(1-\eta)}{n}} \frac{\|\mathbf{s}\|_2}{\sigma}\right) \leq \tilde{P}_D \leq Q\left(Q^{-1}(\alpha) + \sqrt{\frac{m(1+\eta)}{n}} \frac{\|\mathbf{s}\|_2}{\sigma}\right) \quad (6.20)$$

where α is the probability of false alarm. When $Q^{-1}(\alpha) = \frac{1}{2\sigma} \|\mathbb{H}\mathbf{s}\|_2$, it is straightforward to show that

$$\begin{aligned}
1 - \tilde{P}_D &= Q\left(\frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} - Q^{-1}(\alpha)\right) \\
&= Q\left(\frac{\|\mathbb{H}\mathbf{s}\|_2}{2\sigma}\right) \\
&= \tilde{P}_e.
\end{aligned} \tag{6.21}$$

Hence, in the special case where prior probabilities of the two hypotheses are equal, i.e., $p_0 = p_1 = \frac{1}{2}$, and the noise is White Gaussian noise, i.e., $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$, the NP detector presented in [66] and the Bayesian approach presented in this chapter have the same detection performance.

6.3.2 Approximation of the probability of error and its upper bound with compressed measurements

While Theorem 6.2 gives the general expression for the probability of error $\tilde{P}_e$ for detection with compressed measurements in (6.9), it is expressed in the form of a $Q(\cdot)$ function and hence, numerically rather demanding integration is required. In applications, it is often desirable to find an approximate but simpler expression for the probability of error $\tilde{P}_e$ without doing integration. This has motivated us to investigate in this sub-section the approximation of the probability of error as well as of its upper bound.

Theorem 6.4. (Appendix 6.9) Consider the same detection problem as in Theorem 6.2. For a large $\frac{\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2}{2}$ and a sufficiently small $|\varepsilon|$, the probability of error $\tilde{P}_e$

can be approximated as

$$\bar{P}_e := \frac{2 - \varepsilon^2}{\sqrt{2\pi} \|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2} e^{-\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2^2}{8}}. \quad (6.22)$$

Moreover, when $|\varepsilon| \ll 1$, $\tilde{P}_e$ can be further approximated as

$$\hat{P}_e := \frac{2}{\sqrt{2\pi} \|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2} e^{-\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2^2}{8}}. \quad (6.23)$$

Theorem 6.4 provides a simple way to calculate the probability of error which avoids the calculation of integration. It also shows that compared with the term $Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2}\right)$ in (6.9), the influence on $\tilde{P}_e$ induced by the unequal prior probabilities of the two hypotheses is very little when the difference between prior probabilities $|\varepsilon|$ is sufficiently small. However, the theorem will not work when $|\varepsilon|$ is large. For example, when $|\varepsilon|$ approaches 1, we know that $\tilde{P}_e$ will tend to zero. But neither $\bar{P}_e$ in (6.22) nor $\hat{P}_e$ in (6.23) will tend to zero in such a case.

To study the behavior of the probability of detection $\tilde{P}_D$ with compressed measurements, an approximate but simple detection upper bound was proposed in [66] by using the Chernoff bound [128]: $Q(x) \leq \frac{e^{-\frac{x^2}{2}}}{2}$. As stated in Corollary 6.2, when the false alarm value is $Q^{-1}(\alpha) = \frac{1}{2\sigma} \|\mathbb{H} \mathbf{s}\|_2$, the prior probabilities are equal and the noise is white Gaussian noise, the probability of error $\tilde{P}_e$ derived in this chapter satisfies $\tilde{P}_e = 1 - \tilde{P}_D$. Inspired by the approximate probability of error presented in Theorem 6.4, here we provide a tighter upper bound than the one presented in [66] by utilizing another bound function: $Q(x) < \frac{1}{\sqrt{2\pi}x} e^{-\frac{x^2}{2}}$ for any $x > 0$. The new upper bound is in the form of a piecewise function by combining $Q(x) \leq \frac{e^{-\frac{x^2}{2}}}{2}$ and $Q(x) < \frac{1}{\sqrt{2\pi}x} e^{-\frac{x^2}{2}}$. For a fair comparison with the result in [66], we only consider

the case when the prior probabilities are equal and the noise is white Gaussian noise in the following theorem.

Theorem 6.5. (*Appendix 6.10*) *For the same detection problem in Corollary 6.1, we have the following.*

1. *Suppose that the sampling matrix $\sqrt{n/m}\mathbb{H}$ satisfies the RIP with constant $\eta \in (0, 1)$, then the probability of error satisfies*

$$\left\{ \begin{array}{ll} \tilde{P}_e < \frac{1}{2}e^{-(1-\eta)\text{SNR}\frac{m}{8n}} & \text{if } \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} < \frac{4}{\sqrt{2\pi}} \\ \tilde{P}_e < \frac{2\sqrt{n}}{\sqrt{2m\pi(1-\eta)\text{SNR}}}e^{-(1-\eta)\text{SNR}\frac{m}{8n}} & \text{if } \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} > \frac{4}{\sqrt{2\pi}} \end{array} \right. \quad (6.24a)$$

where $\text{SNR} = \frac{\|\mathbf{s}\|_2^2}{\sigma^2}$.

2. *Suppose that the mutual coherence of matrix $\mathbb{H}$ is μ , the probability of error satisfies*

$$\left\{ \begin{array}{ll} \tilde{P}_e < \frac{1}{2}e^{-[1-(k-1)\mu]\text{SNR}\frac{m}{8n}} & \text{if } \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} < \frac{4}{\sqrt{2\pi}} \\ \tilde{P}_e < \frac{2\sqrt{n}}{\sqrt{2m\pi[1-(k-1)\mu]\text{SNR}}}e^{-[1-(k-1)\mu]\text{SNR}\frac{m}{8n}} & \text{if } \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} > \frac{4}{\sqrt{2\pi}} \end{array} \right. \quad (6.25a)$$

In [66], only (6.24a) is adopted as the upper bound function. It is apparent that the piecewise function in (6.24) is a better approximation than only using the Chernoff bound as when $\frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} > \frac{4}{\sqrt{2\pi}}$, $\frac{2\sqrt{n}}{\sqrt{2m\pi(1-\eta)\text{SNR}}}e^{-(1-\eta)\text{SNR}\frac{m}{8n}}$ is smaller than $\frac{1}{2}e^{-(1-\eta)\text{SNR}\frac{m}{8n}}$. Moreover, with (6.24), we find that the probability of error

approaches 0 exponentially if the number of measurements and/or the SNR increases.

The boundary point of the piecewise function is $\frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} = \frac{4}{\sqrt{2\pi}}$. However, we cannot know the specific SNR at the boundary point if the sampling matrix is not fixed. In the case that $\|\mathbb{H}\mathbf{s}\|_2 = \|\mathbf{s}\|_2$, it is easy to calculate the boundary point at around 4.0616 dB.

6.4 Simulations

In this section, several numerical simulations are presented to illustrate the effectiveness of the new results presented in the previous sections.

6.4.1 General Bayesian signal detection

In this sub-section, we present simulation results on general Bayesian signal detection by obtaining the probability of error P_e with respect to different prior probabilities by using two different values of $\sqrt{\mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}}$. To simplify the discussion, Gaussian noise with zero mean and diagonal but unequal variances is tested, where the maximal and minimal variance values are set to be $\sigma_{\max}^2 = 0.2$ and $\sigma_{\min}^2 = 0.1$, respectively. To compare P_e for two different values of $\sqrt{\mathbf{s}^T C_{\mathbf{w}}^{-1} \mathbf{s}}$, two different signals with 10 dimensions are generated, where

$$\mathbf{s}_1 = 0.6 \times [1 \quad -1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1]^T$$

and

$$\mathbf{s}_2 = 0.58 \times [1 \quad -1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1]^T.$$

Figure 6.1: The simulation results of P_e with respect to different prior probabilities and two different values of $\sqrt{s^T C_w^{-1} s}$.

Fig. 6.1 depicts the simulation results on P_e when the prior probability p_0 of hypothesis $\mathcal{H}_0$ is increasing from 0.5 to 0.9. From the figure, we find that P_e decreases when p_0 increases. This validates result 1) of Theorem 6.1. Moreover, Fig. 6.1 shows that the larger $\sqrt{s^T C_w^{-1} s}$ is, the smaller P_e is. This verifies the second result in Theorem 6.1.

6.4.2 Probability of error $\tilde{P}_e$ in Bayesian detection with compressed measurements

This subsection presents various simulation results on the probability of error $\tilde{P}_e$ in signal detection in the measurement domain using the Bayesian approach.

A. Upper and lower bounds with respect to different prior probabilities

To verify Theorem 6.3, the upper and lower bound of $\tilde{P}_e$ with respect to dif-

Figure 6.2: Upper, lower bounds, and the true value of the probability of error on different prior probabilities.

ferent prior probabilities are tested by simulation here. $\tilde{P}_e$ evaluated using (6.9) is also shown for comparison. In the simulation, 350 measurements are made by the sampling matrix where the entries of the sampling matrix are identically and independently generated from a standard normal distribution and then orthogonalized. The Gaussian noise with zero mean and diagonal but unequal variances is added in the generation of the measurements. The maximal and minimal variances of the noise covariance matrix are set to be 0.02 and 0.01, respectively. The prior probability p_0 is increasing from 0.5 to 0.7. The upper and lower bounds of the probability of error are calculated using (6.15). The mutual coherence μ of the sampling matrix is 0.0705 in our simulation. A sparse signal with 1024 dimensions and 6 nonzero entries is generated as the sparsity of the signal should satisfy $k < \frac{1}{2}(\frac{1}{\mu} + 1) = 7.5922$.

Fig. 6.2 shows the detailed simulation results. In this figure, the solid line represents the true value of the probability of error with (6.9) where the detection is directly using compressed measurements and the Gaussian noise is zero mean and unequal diagonal covariance matrix. The labels “□” and “+” denote the upper and lower bound in (6.15). One can see from this figure that all the probability of errors decrease when the prior probability p_0 is increasing. Moreover, the probability of error obtained with compressed measurements is bounded by the upper and lower bound obtained with Theorem 6.3.

B. Upper and lower bounds with respect to signal power

In this simulation, the prior probabilities are assumed to be equal. The maximal and the minimal variances of the noise are set to be 0.2 and 0.16, respectively. Sparse signal with 6 nonzero entries and different spikes varying from 0.05 to 2 is generated, where the dimension of the signal is set to be 1024. The sampling

Figure 6.3: Upper, lower bounds, and the true value of the probability of error with respect to signal power.

matrix is generated where the entries are identically and independently sampled from a standard normal distribution and then orthogonalized. In the simulation, 350 measurements are used and the mutual coherence of the sampling matrix is $\mu = 0.0724$. Fig. 6.3 depicts the upper and lower bounds by Theorem 6.3 and the true value of $\tilde{P}_e$ by (6.9). It is easy to find from this figure that the probability of error $\tilde{P}_e$ decreases monotonically when the power of signal increases. Moreover, $\tilde{P}_e$ is bounded by the upper and lower bounds, as expected.

C. Approximation of the probability of error

To show the approximate probability of error given in (6.22) of Theorem 6.4, the detection performance with respect to different prior probabilities is tested in this simulation, in comparison with the true expression given in (6.9). In this simulation, a sparse signal with 1024 dimensions and 10 nonzero entries is generated for testing. White Gaussian noise with zero mean and $\sigma^2 = 0.1$ is added.

Fig. 6.4 draws the probability of error calculated with the approximation expression (6.22) and the true value calculated with (6.9). From this figure, one can find that the approximate probability of error is quite close to the true value of probability of error. The relative error between the approximate $\tilde{\tilde{P}}_e$ and the true value $\tilde{P}_e$ divided by the true value $\tilde{P}_e$, the ratio is as low as 6.12%. Moreover, when increasing the prior probability of hypothesis $\mathcal{H}_0$, the probability of error is decreasing for both the true value and the approximate one.

D. Approximation of the upper bound

In this simulation, we compare the approximate upper bound provided by the piecewise function of (6.25) in Theorem 6.5 with the corresponding upper bound given in [66]. The true probability of error by (6.9) and the upper bound of the probability of error shown in (6.18) of Corollary 6.1 are also provided for compar-

Figure 6.4: The probability of errors comparisons with the approximation result $\tilde{P}_e$ and the true value $\tilde{P}_e$ on different prior probabilities.

isons.

In the simulation, a 1024-dimensional sparse signal with 6 nonzero spike values is generated where the localizations of the nonzero entries are randomly assigned. The variance of the White Gaussian noise is set to be $\sigma^2 = 0.2$. The spike value of the nonzero entries in the signal is gradually increased from 0.4 to 2. The sampling matrix is generated where the entries are identically and independently sampled from a standard normal distribution and then orthogonalized. 350 measurements are made by the sampling matrix and the mutual coherence of the sampling matrix is $\mu = 0.0715$.

Fig. 6.5 depicts the true value of probability of error using (6.9), the bound of the probability of error calculated with the method in [66], the upper bound of probability of error in Corollary 6.1 with (6.18) and the approximation upper bound of the probability of error in Theorem 6.5 with (6.25) with respect to different SNR levels. From this figure, one can find that when SNR is sufficiently small which makes $\frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} < \frac{4}{\sqrt{2\pi}}$, the curve of the bound given by our proposed equation in Theorem 6.5 is the same as the one provided in [66] as the first equation of the piecewise function (6.25) is used for calculating the upper bound of the probability of error. However, when SNR increases to a large value which makes $\frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} > \frac{4}{\sqrt{2\pi}}$, the curve of the upper bound calculated by the piecewise function in Theorem 6.5 is lower than the one derived in [66]. Moreover, in such case, the upper bound of the probability of error provided in Theorem 6.5 is tighter than the one proposed in [66]. As shown in Fig. 6.5, the boundary point of the piecewise function in (6.25) is around 5.8 dB in the simulation. Although the boundary point is at $\frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} = \frac{4}{\sqrt{2\pi}}$, the boundary point represented in SNR level is varying as the matrix $\mathbb{H}$ is different for different realizations.

Figure 6.5: Comparisons of performance bounds for the probability of error on different SNR levels.

6.5 Conclusions

We have conducted a detailed investigation on the signal detection problem by processing compressed measurements directly using the Bayesian approach instead of the classical Neyman-Pearson approach adopted in [66]. We have derived a general expression for the probability of error where the prior probabilities could be equal or unequal and the additive noise is uncorrelated Gaussian noise with possibly unequal variances. We have shown that the probability of error is always smaller for unequal prior probabilities than for equal ones. Given a sampling matrix in compressed sensing, we have also derived performance bounds for the probability of error using the *restricted isometry property* constant and then the mutual coherence of the matrix, the latter being computationally more attractive by making use of the relation between the RIP constant and the mutual coherence given in [127]. When the difference between the prior probabilities is sufficiently small and the signal to noise ratio is relatively large, we have obtained an approximate expression of the probability of error which is easier to calculate. Furthermore, an approximate upper bound of the probability of error is derived in terms of a piecewise function and it is shown to be a tighter bound than the one presented in [66] when the signal to noise ratio is relatively large. Numerical simulations have also been provided to illustrate the new theoretical results presented in this chapter.

6.6 Appendix: Proof of Theorem 6.1

1) For convenience of presentation, we let $a := \frac{\sqrt{s^T C_w^{-1} s}}{2} > 0$ and without loss of generality, assume that $p_0 > p_1$. Since $p_0 + p_1 = 1$ and $p_0 - p_1 = \varepsilon > 0$, we have

$p_0 = \frac{1}{2}(1 + \varepsilon)$ and $p_1 = \frac{1}{2}(1 - \varepsilon)$. In such a case, P_e in (6.4) can be equivalently rewritten as a function of ε and a in the following

$$P_e(\varepsilon, a) = Q(a) + \kappa(\varepsilon, a) \quad (6.26)$$

with $Q(a) = \frac{1}{\sqrt{2\pi}} \int_a^\infty e^{-\frac{u^2}{2}} du$ and

$$\kappa(\varepsilon, a) = \frac{1 - \varepsilon}{2\sqrt{2\pi}} \int_{a - \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})}^a e^{-\frac{u^2}{2}} du - \frac{1 + \varepsilon}{2\sqrt{2\pi}} \int_a^{a + \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})} e^{-\frac{u^2}{2}} du, \quad (6.27)$$

For a fixed a , we have $\frac{\partial P_e(\varepsilon, a)}{\partial \varepsilon} = \frac{\partial \kappa(\varepsilon, a)}{\partial \varepsilon}$ and for $\varepsilon > 0$

$$\begin{aligned} \frac{\partial \kappa(\varepsilon, a)}{\partial \varepsilon} &= -\frac{1}{2\sqrt{2\pi}} \int_{a - \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})}^a e^{-\frac{u^2}{2}} du \\ &\quad + \frac{1 - \varepsilon}{2\sqrt{2\pi}} \left[-e^{-\frac{1}{2} \left[a - \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon}) \right]^2} \left(\frac{-1}{a(1 - \varepsilon^2)} \right) \right] \\ &\quad - \frac{1}{2\sqrt{2\pi}} \int_a^{a + \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})} e^{-\frac{u^2}{2}} du \\ &\quad - \frac{1 + \varepsilon}{2\sqrt{2\pi}} \left[e^{-\frac{1}{2} \left[a + \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon}) \right]^2} \left(\frac{1}{a(1 - \varepsilon^2)} \right) \right] \\ &= -\frac{1}{2\sqrt{2\pi}} \int_{a - \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})}^a e^{-\frac{u^2}{2}} du - \frac{1}{2\sqrt{2\pi}} \int_a^{a + \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})} e^{-\frac{u^2}{2}} du \\ &\quad + \frac{1}{2\sqrt{2\pi}} \frac{1}{a(1 - \varepsilon^2)} e^{-\frac{1}{2} \left(a^2 + \left[\frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon}) \right]^2 \right)} \left[(1 - \varepsilon) e^{\frac{1}{2} \ln(\frac{1+\varepsilon}{1-\varepsilon})} - (1 + \varepsilon) e^{-\frac{1}{2} \ln(\frac{1+\varepsilon}{1-\varepsilon})} \right] \\ &= -\frac{1}{2\sqrt{2\pi}} \int_{a - \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})}^a e^{-\frac{u^2}{2}} du - \frac{1}{2\sqrt{2\pi}} \int_a^{a + \frac{1}{2a} \ln(\frac{1+\varepsilon}{1-\varepsilon})} e^{-\frac{u^2}{2}} du. \\ &< 0. \end{aligned} \quad (6.28)$$

As $\kappa(0, a) = 0$ and for $\varepsilon > 0$, $\frac{\partial \kappa(\varepsilon, a)}{\partial \varepsilon} < 0$, $\kappa(\varepsilon, a) < 0$ also.

Since $\frac{\partial P_e(\varepsilon, a)}{\partial \varepsilon} < 0$ for $\varepsilon > 0$ and a fixed a , $P_e(\varepsilon, a)$ is a monotonically decreasing

function of ε . It can be similarly shown that for $\varepsilon < 0$ and a fixed a , $\kappa < 0$ and $P_e(\varepsilon, a)$ is a monotonically increasing function of ε . Therefore, we conclude that for a fixed $\frac{\sqrt{s^T C_w^{-1} s}}{2}$ and $\varepsilon \neq 0$, $\kappa < 0$ and P_e decreases when $|\varepsilon|$ increases.

2) If $p_0 = p_1$, then $\varepsilon = 0$ and $\kappa(0, a) = 0$, P_e in (6.4) reduces to $Q(a)$. Hence,

$$\begin{aligned}
\frac{\partial P_e(0, a)}{\partial a} &= \frac{\partial Q(a)}{\partial a} \\
&= -\frac{1}{\sqrt{2\pi}} e^{-\frac{a^2}{2}} \\
&< 0
\end{aligned} \tag{6.29}$$

Hence, $P_e(0, a)$ is a monotonically decreasing function of a . Thus, when $\varepsilon = 0$, $P_e(0, a)$ is decreasing when a is increasing.

If $p_0 \neq p_1$, without loss of generality, we assume that $p_0 > p_1$. Then, for fixed

p_0 and p_1 , ε is a fixed positive scalar. We have

$$\begin{aligned}
\frac{\partial P_e(\varepsilon, a)}{\partial a} &= -\frac{1}{\sqrt{2\pi}}e^{-\frac{a^2}{2}} + \frac{p_1}{\sqrt{2\pi}} \left(e^{-\frac{a^2}{2}} - e^{-\frac{(a - \frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \left[1 + \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \right) \\
&\quad - \frac{p_0}{\sqrt{2\pi}} \left(e^{-\frac{(a + \frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \left[1 - \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] - e^{-\frac{a^2}{2}} \right) \\
&= -\frac{p_1}{\sqrt{2\pi}} e^{-\frac{(a - \frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \left[1 + \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \\
&\quad - \frac{p_0}{\sqrt{2\pi}} e^{-\frac{(a + \frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \left[1 - \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \\
&= -\frac{p_1}{\sqrt{2\pi}} e^{-\frac{a^2 + (\frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} e^{\frac{1}{2} \ln \frac{p_0}{p_1}} \left[1 + \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \\
&\quad - \frac{p_0}{\sqrt{2\pi}} e^{-\frac{a^2 + (\frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} e^{-\frac{1}{2} \ln \frac{p_0}{p_1}} \left[1 - \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \\
&= -\frac{\sqrt{p_0 p_1}}{\sqrt{2\pi}} e^{-\frac{a^2 + (\frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \left[1 + \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \\
&\quad - \frac{\sqrt{p_0 p_1}}{\sqrt{2\pi}} e^{-\frac{a^2 + (\frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \left[1 - \frac{1}{2a^2} \ln \frac{p_0}{p_1} \right] \\
&= -\frac{2\sqrt{p_0 p_1}}{\sqrt{2\pi}} e^{-\frac{a^2 + (\frac{1}{2a} \ln \frac{p_0}{p_1})^2}{2}} \\
&< 0.
\end{aligned} \tag{6.30}$$

Thus, $P_e(\varepsilon, a)$ is a monotonically decreasing function of a . Therefore, $P_e(\varepsilon, a)$ decreases as $\frac{\sqrt{\mathbf{s}^T \mathbf{C}_w^{-1} \mathbf{s}}}{2}$ increases.

6.7 Appendix: Proof of Theorem 6.2

1) From (6.8), it is easy to show that $\tilde{T}$ has the following PDF

$$\tilde{T} \sim \begin{cases} \mathcal{N}(\mathbf{0}, \|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2^2) & \mathcal{H}_0 \\ \mathcal{N}(\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2^2, \|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2^2) & \mathcal{H}_1 \end{cases} \quad (6.31)$$

where $\mathbb{H}_{C_w} = \tilde{\Phi}^T (\tilde{\Phi} \tilde{\Phi}^T)^{-1} \tilde{\Phi}$ with $\tilde{\Phi} = \Phi \Lambda$ and $\Lambda = C_w^{\frac{1}{2}}$. From (6.6) and (6.31), the probability of error $\tilde{P}_e$ is

$$\begin{aligned} \tilde{P}_e &= P(\mathcal{H}_0) Q\left(\frac{\gamma}{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}\right) + P(\mathcal{H}_1) Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2^2 - \gamma}{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}\right) \\ &= p_0 Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2} + \frac{\ln(\tilde{\delta})}{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}\right) + p_1 Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2} - \frac{\ln(\tilde{\delta})}{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}\right) \\ &= p_0 Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2} + \tilde{\xi}\right) + p_1 Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2} - \tilde{\xi}\right) \\ &= Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2}\right) + \tilde{\kappa} \end{aligned} \quad (6.32)$$

where $\tilde{\kappa}$ is given in (6.10) and $\tilde{\xi}$ in (6.11).

2) and 3) The proofs are similar to the proofs for 1) and 2) of Theorem 6.1 and are hence omitted here.

4) Equation (6.9) or (6.32) can be rewritten as

$$\tilde{P}_e = Q\left(\frac{\|\mathbb{H}_{C_w} \Lambda^{-1} \mathbf{s}\|_2}{2}\right) + \tilde{\kappa} \quad (6.33)$$

Since $\Lambda^{-1} \mathbf{s} \in \mathcal{S}$ is still a sparse signal with the same sparsity as $\mathbf{s}$ and the matrix

$\sqrt{n/m}\mathbb{H}_{C_w}$ satisfies the RIP, we have

$$\sqrt{1-\eta}\|\Lambda^{-1}\mathbf{s}\|_2 \leq \sqrt{n/m}\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2 \leq \sqrt{1+\eta}\|\Lambda^{-1}\mathbf{s}\|_2$$

and furthermore,

$$\sqrt{1-\eta}\sqrt{\frac{m}{n}}\frac{\|\mathbf{s}\|_2}{\sigma_{\max}} \leq \|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2 \leq \sqrt{1+\eta}\sqrt{\frac{m}{n}}\frac{\|\mathbf{s}\|_2}{\sigma_{\min}}. \quad (6.34)$$

The above inequality leads to

$$Q\left(\sqrt{1+\eta}\sqrt{\frac{m}{n}}\frac{\|\mathbf{s}\|_2}{2\sigma_{\min}}\right) \leq Q\left(\frac{\|\mathbb{H}_{C_w}\Lambda^{-1}\mathbf{s}\|_2}{2}\right) \leq Q\left(\sqrt{1-\eta}\sqrt{\frac{m}{n}}\frac{\|\mathbf{s}\|_2}{2\sigma_{\max}}\right). \quad (6.35)$$

From (6.33) and (6.35), we arrive at the bounds for $\tilde{P}_e$ as given in (6.12).

6.8 Appendix: Proof of Theorem 6.3

For any matrix Φ satisfying the RIP, assume that the mutual coherence of the matrix Φ is μ and the sparsity k of the signal satisfies $k < \frac{1}{2}\left(\frac{1}{\mu} + 1\right)$, then, it has been shown in [127] that the RIP constant η of (1.12) satisfies the following equation

$$\eta \leq (k-1)\mu. \quad (6.36)$$

Hence, we have

$$\begin{cases} 1 + \eta \leq 1 + (k-1)\mu & (6.37a) \\ 1 - \eta \geq 1 - (k-1)\mu & (6.37b) \end{cases}$$

Substituting (6.37) into (6.12) of Theorem 6.2, we can obtain the results in Theorem 6.3.

6.9 Appendix: Proof of Theorem 6.4

For a large value of x , say $x \geq 4$, the cumulative distribution function $Q(x)$ can be approximated with the following equation

$$Q(x) \approx \frac{1}{\sqrt{2\pi}x} e^{-\frac{1}{2}x^2}. \quad (6.38)$$

It was stated in [68], the error between the approximation using (6.38) and the true value of $Q(x)$ is smaller than 1.7863×10^{-6} when $x \geq 4$.

Hence, for a relatively large $\frac{\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2}{2}$, the term $Q\left(\frac{\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2}{2}\right)$ in (6.9) can be approximated as

$$Q\left(\frac{\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2}{2}\right) \approx \frac{2}{\sqrt{2\pi} \|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2} e^{-\frac{1}{8}\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2^2} \quad (6.39)$$

Without the loss of generality, we assume that $p_0 > p_1$. The case of $p_0 < p_1$ can be proved similarly and is omitted here. Using the same expressions of p_0 and p_1 as a function of ε and the same argument used in the proof of result 1) of Appendix 6.6, $\tilde{\kappa}$ in (6.9) can be expressed in the same form as (6.27) but with a redefined as $a := \frac{\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2}{2}$ here. Now we use Taylor series [129] to expand $\tilde{\kappa}(\varepsilon, a)$ around the point of $\varepsilon = 0$. In the approximation, we only keep up to the second order term as the higher order terms are very small when $|\varepsilon|$ is sufficiently small. Hence, continuing from (6.28), the second-order partial derivative of $\tilde{\kappa}(\varepsilon, a)$ with respect

to ε can be derived as

$$\begin{aligned}
\frac{\partial \tilde{\kappa}^2(\varepsilon, a)}{\partial \varepsilon^2} &= -\frac{1}{2\sqrt{2\pi}} \left[-e^{-\frac{1}{2}\left[a - \frac{1}{2a} \ln\left(\frac{1+\varepsilon}{1-\varepsilon}\right)\right]^2} \left(\frac{-1}{a(1-\varepsilon^2)}\right) \right] \\
&\quad - \frac{1}{2\sqrt{2\pi}} \left[e^{-\frac{1}{2}\left[a + \frac{1}{2a} \ln\left(\frac{1+\varepsilon}{1-\varepsilon}\right)\right]^2} \left(\frac{1}{a(1-\varepsilon^2)}\right) \right] \\
&= -\frac{1}{2\sqrt{2\pi}} \frac{1}{a(1-\varepsilon^2)} e^{-\frac{1}{2}\left(a^2 + \left[\frac{1}{2a} \ln\left(\frac{1+\varepsilon}{1-\varepsilon}\right)\right]^2\right)} \left[\sqrt{\frac{1+\varepsilon}{1-\varepsilon}} + \sqrt{\frac{1-\varepsilon}{1+\varepsilon}} \right]
\end{aligned} \tag{6.40}$$

From (6.28) and (6.40), we have $\left. \frac{\partial \tilde{\kappa}(\varepsilon, a)}{\partial \varepsilon} \right|_{\varepsilon=0} = 0$ and

$$\left. \frac{\partial \tilde{\kappa}^2(\varepsilon, a)}{\partial \varepsilon^2} \right|_{\varepsilon=0} = -\frac{1}{\sqrt{2\pi}a} e^{-\frac{1}{2}a^2} = -\frac{2}{\sqrt{2\pi} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2} e^{-\frac{1}{8} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2^2}.$$

Hence,

$$\begin{aligned}
\tilde{\kappa}(\varepsilon, a) &\approx \tilde{\kappa}(0, a) + \left. \frac{\partial \tilde{\kappa}(\varepsilon, a)}{\partial \varepsilon} \right|_{\varepsilon=0} (\varepsilon - 0) + \frac{1}{2!} \left. \frac{\partial \tilde{\kappa}^2(\varepsilon, a)}{\partial \varepsilon^2} \right|_{\varepsilon=0} (\varepsilon - 0)^2 \\
&= -\frac{\varepsilon^2}{\sqrt{2\pi} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2} e^{-\frac{1}{8} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2^2}.
\end{aligned} \tag{6.41}$$

Combining (6.39) and (6.41), $\tilde{P}_e$ in (6.9) could be approximated as

$$\tilde{P}_e \approx \bar{P}_e = \frac{2 - \varepsilon^2}{\sqrt{2\pi} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2} e^{-\frac{1}{8} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2^2} \tag{6.42}$$

When $|\varepsilon| \ll 1$, the probability of error $\tilde{P}_e$ can be further approximated as

$$\tilde{P}_e \approx \hat{P}_e = \frac{2}{\sqrt{2\pi} \|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2} e^{-\frac{\|\mathbb{H}_{C\mathbf{w}} \Lambda^{-1} \mathbf{s}\|_2^2}{8}}. \tag{6.43}$$

Note that

$$\lim_{\varepsilon \rightarrow 0} \frac{\hat{\tilde{P}}_e - \tilde{P}_e}{\hat{\tilde{P}}_e} = \lim_{\varepsilon \rightarrow 0} \frac{\varepsilon^2}{2} = 0. \quad (6.44)$$

Therefore, when $\frac{\|\mathbb{H}_{C\mathbf{w}}\Lambda^{-1}\mathbf{s}\|_2}{2}$ is large and the difference between prior probabilities is sufficiently small, $\tilde{P}_e$ is dominated by $\hat{\tilde{P}}_e$.

6.10 Appendix: Proof of Theorem 6.5

1) Comparing with the two functions $Q(x) \leq \frac{1}{2}e^{-\frac{x^2}{2}}$ (for any x) and $Q(x) < \frac{1}{\sqrt{2\pi x}}e^{-\frac{x^2}{2}}$ (for $x > 0$), we find that when $x < \frac{2}{\sqrt{2\pi}}$, $\frac{1}{2}e^{-\frac{x^2}{2}} < \frac{1}{\sqrt{2\pi x}}e^{-\frac{x^2}{2}}$, while when $x > \frac{2}{\sqrt{2\pi}}$, $\frac{1}{2}e^{-\frac{x^2}{2}} > \frac{1}{\sqrt{2\pi x}}e^{-\frac{x^2}{2}}$. Therefore, for the $Q(\cdot)$ function, the following bound function would be more accurate than only using any one of them

$$\begin{cases} Q(x) < \frac{1}{2}e^{-\frac{x^2}{2}} & \text{if } x < \frac{2}{\sqrt{2\pi}} \end{cases} \quad (6.45a)$$

$$\begin{cases} Q(x) < \frac{1}{\sqrt{2\pi x}}e^{-\frac{x^2}{2}} & \text{if } x > \frac{2}{\sqrt{2\pi}} \end{cases} \quad (6.45b)$$

where the boundary point is $x = \frac{2}{\sqrt{2\pi}}$.

For the detection problem with compressed measurements in Corollary 6.1, the boundary point is $x = \frac{\|\mathbb{H}\mathbf{s}\|_2}{2\sigma} = \frac{2}{\sqrt{2\pi}}$. Hence, with (6.45), the probability of error $\tilde{P}_e$ satisfies

$$\begin{cases} \tilde{P}_e < \frac{1}{2}e^{-\frac{\|\mathbb{H}\mathbf{s}\|_2^2}{8\sigma^2}} & \text{if } \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} < \frac{4}{\sqrt{2\pi}} \end{cases} \quad (6.46a)$$

$$\begin{cases} \tilde{P}_e < \frac{2\sigma}{\sqrt{2\pi} \|\mathbb{H}\mathbf{s}\|_2}e^{-\frac{\|\mathbb{H}\mathbf{s}\|_2^2}{8\sigma^2}} & \text{if } \frac{\|\mathbb{H}\mathbf{s}\|_2}{\sigma} > \frac{4}{\sqrt{2\pi}} \end{cases} \quad (6.46b)$$

Since the matrix $\sqrt{n/m}\mathbb{H}$ satisfies the RIP for a small constant η , for sparse signal $\mathbf{s}$, $\|\mathbb{H}\mathbf{s}\|_2$ is within the region $\left[\sqrt{\frac{m(1-\eta)}{n}} \|\mathbf{s}\|_2, \sqrt{\frac{m(1+\eta)}{n}} \|\mathbf{s}\|_2 \right]$ as shown in (6.34).

Hence, combining the result given in (6.46), we can easily get the bound shown in (6.24).

2) For matrix with the mutual coherence value μ , the bound of the probability of error in (6.25) can be obtained with (6.37) and (6.24).

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we have conducted further studies of the extreme learning machine algorithm and the signal detection in compressed sensing. We have presented several improved algorithms based on ELM and proposed some contributions to signal detection in compressed sensing.

In Chapter 2, a novel structure of WNN with extreme learning machine (ELM) named CFWNN-ELM has been first introduced. In CFWNN-ELM, composite functions are applied at the hidden nodes and the learning is done using ELM. The input information is first processed by wavelet functions and then passed through a type of bounded piecewise continuous activation functions. A selection method that takes into account the domain of input space where the wavelets are not zero has been used to initialize the translation and dilation parameters. The formed wavelet neural network is then trained with the computationally efficient ELM algorithm. Experimental results on the regression of some nonlinear functions and real-world data and classifications on several benchmark real-world data sets have shown that

CFWNN-ELM can achieve better performances in most cases than some relevant neural networks and learn much faster than the traditional backpropagation (BP) algorithm and support vector machines (SVM).

Incorporating the DE algorithm into the composite function wavelet neural networks trained with ELM, an improved method to CFWNN-ELM named CWN-E-ELM has been presented in Chapter 3. In CFWNN-ELM, a lot of redundant nodes may exist as the number of hidden neurons are assigned *a priori* and the input weight matrix and bias vector are randomly generated once and never changed. DE has been used to search for the optimal network parameters in CWN-E-ELM. Simulations on several applications have illustrated that the proposed CWN-E-ELM is able to achieve better generalization performances than several relevant methods in general and the network size is much more compact than CFWNN-ELM due to using the DE algorithm which helps choose the optimal hidden node parameters.

To further enhance the classification rate of the ELM algorithm, we have presented an improved algorithm named voting based extreme learning machine (V-ELM) for signal classification in Chapter 4. In V-ELM, the voting method has been incorporated into the extreme learning machine in classification applications. Several individual ELMs with the same network structure have been trained with the same dataset and the final class label of a test sample was determined by the majority voting method on all the results obtained by these independent ELMs. Several propositions have also been proposed to support this algorithm. Simulations on many real world classification datasets have demonstrated that V-ELM generally outperforms the original ELM algorithm as well as several recent state-

of-art classification algorithms, such as SVM, KNN, and MSCC.

In Chapter 5, we have obtained the theoretical detection bound of the probability of error by using the Bayesian approach in compressed sensing where the signal is sparse and is reconstructed from a compressed measurement vector. Based on the oracle estimator in CS, the theoretical bound of the detection error has been provided when the noise in the estimation of CS is white Gaussian noise. Without any additional information in CS, we have shown in Chapter 5 that the probability of error obtained using the signal reconstructed by any recovery algorithm is always larger than the theoretical bound. We have compared our theoretical bound with the detection error obtained by using the signal reconstructed with four recovery algorithms: BPDN algorithm, the Dantzig selector, the OMP method and the CoSaMP algorithm. Simulations have been conducted to show the effectiveness of our result.

In Chapter 6, we have conducted a detailed investigation on the signal detection problem by processing compressed measurements directly using the Bayesian approach instead of the classical Neyman-Pearson approach adopted in [66]. We have derived a general expression for the probability of error where the prior probabilities could be equal or unequal and the additive noise is uncorrelated Gaussian noise with possibly unequal variances. We have shown that the probability of error is always smaller for unequal prior probabilities than for equal ones. Given a sampling matrix in compressed sensing, we have also derived performance bounds for the probability of error using the *restricted isometry property* constant and then the mutual coherence of the matrix, the latter being computationally more

attractive, by making use of the relation between the RIP constant and the mutual coherence given in [127]. When the difference between the prior probabilities is sufficiently small and the signal to noise ratio is relatively large, we have obtained an approximate expression of the probability of error which is easier to calculate. Furthermore, an approximate upper bound of the probability of error is derived in terms of a piecewise function and it is shown to be a tighter bound than the one presented in [66] when the signal to noise ratio is relatively large. Numerical simulations have also been provided to illustrate the new results.

7.2 Future Work

As shown in the above discussions, both the extreme learning machine algorithm and compressed sensing play important roles in signal processing. The topics listed below are worthy of further research.

1. *Combinations of extreme learning machine and compressed sensing*

From discussions shown in this thesis, we find that both the extreme learning machine and compressed sensing are built on a linear model. Moreover, both of them adopt random parameters, where ELM uses random input weights and hidden biases to form the hidden layer output matrix while CS utilizes random ensembles to generate the sampling matrix. The similarities between these two topics open doors for new research activities. Generally, there may exist two ways to combine these two topics.

First, as we have discussed in Chapter 3, incorporating differential evolution algorithm to search the sub-optimal hidden node parameters could efficiently reduce the network size. This also means that there may exist several re-

dundant nodes in ELM and some of the output weights might be very small. Dependent on solving the ℓ_1 optimization problem, compressed sensing is a powerful technique to find the sparse solution in a linear system. Therefore, how to utilize CS to reduce the number of hidden nodes in ELM would be a very interesting and useful future work, especially for hardware implementation. One related work published recently in [130] has shown that ℓ_1 norm optimization has been implemented in ELM for the ELM-based classifier on digital reconfigurable devices. The novel formulation based on ℓ_1 norm optimization scheme for ELM can properly trade off the classifier complexity and the generalization performance, which was further implemented using field-programmable gate arrays (FPGAs) technology and low-end inexpensive devices in [130].

Moreover, it has been recently shown in [131] that in the compressed sensing framework, learning in the measurement domain is possible and it gains both the benefits of compressed sensing and dimensionality reduction simultaneously. However, the authors only proved that the linear kernel SVM classifier in the measurement domain, with high probability, has true accuracy close to the accuracy of the best linear threshold classifier in the data domain. Since the dimensionality reduction problem exists in the machine learning field, research on the connection between compressing sensing and extreme learning machine will be an interesting but challenging future work. To show the feasibility of using the random sampling as a dimension reduction method in neural networks, we have run some simulations to test the performance of learning in the measurement domain in compressed sensing with the traditional gradient descent based algorithm and extreme learning machine in

SLFNs. However, the performance obtained by both the gradient descent based algorithm and the ELM is a little worse than the one by processing directly with the original signal. Therefore, further research on how to efficiently use the compressed measurements directly as the training data is highly desired.

2. *Compressed sensing for signal classification*

As discussed in Chapter 5, the detection bound in compressed sensing based on oracle estimator has been provided. Based on this result, our future work is to investigate the performance bound of classifying signals among multiple hypotheses in compressed sensing by using the Bayesian approach. With the prior information of the support set of the sparse signal, the detection result given in Chapter 5 can be extended to a more general case, that is, classifying signals among multiple hypotheses.

Another future work in this direction is to classify signal directly with compressed measurements with the Bayesian approach. It was shown in [60, 63, 65, 66] that signal recovery from compressed measurements may not be necessary in some signal processing problems, such as detection, classification and filtering. Moreover, we have presented in Chapter 6 that the probability of error by detecting the signal directly with compressed measurements using the Bayesian approach is comparable to the one with the original signal. The merit of using compressed measurements for detection is that we could avoid the reconstruction process for recovering the sparse signal. Since the Bayesian approach is more suitable for handling the multiple hypotheses classification problem than the Neyman-Pearson theorem [67] used in [66], extending the

results presented in Chapter 6 to signal classification with compressed measurements will be a good topic for future research.

3. *Compressed sensing for radar signal processing*

One of the key problems in radar signal processing is to reconstruct a scene from limited measurements of scattered electric fields [132–135]. This is a typical ill-posed linear inverse problem and the recent compressed sensing provides strong solutions for solving underdetermined linear equations with assumptions that the candidate solution is sparse or compressible and the linear system satisfies certain properties. The similarity between models of compressed sensing and the radar reflectivity estimation equation provides a feasibility of adopting developments in compressed sensing into reconstructing the reflectivity signal in radar system [136–147]. Compressive radar image concept has been presented in [136]. A stylized compressed sensing radar with appropriate conditions that can “beat” the classical radar uncertainty principle was proposed in [137]. Compressed sensing has been used in radar sensor network to process Ultra-wide Bandwidth (UWB) radar signals in [138]. In [139], the authors have shown how to reduce the analogue to digital sampling rate in a real digital radar. In [140–145], compressed sensing has been applied to multiple-input multiple-output (MIMO) radar system and its variants for signal estimations. Applications of compressed sensing in synthetic aperture radar (SAR) have been comprehensively studied in [146, 147]. Adaptive compressive sensing and processing method has been proposed for radar tracking problems in [148, 149].

However, we find from above that most prior work focused on using com-

pressed sensing and sparse reconstruction techniques for radar signal acquisition and estimation. Comprehensive studies presented in [65, 66] and Chapter 6 have shown that processing directly with compressed measurements in several general signal processing problems including detection, classification, estimation and filtering can guarantee comparable results to the ones obtained with the original signal. Merits of processing directly in the measurement domain are that it can obtain more significant computational savings and simpler than traditional reconstruct-then-processing methods. Therefore, incorporating the compressed processing techniques by directly considering compressed measurements into radar/SAR signal processing would be a useful and promising research topic.

Author's Publication

Journals:

1. Jiuwen Cao, Zhiping Lin, and G.-B. Huang, "Composite function wavelet neural networks with extreme learning machine," *Neurocomputing*, vol. 73, no. 7-9, pp. 1405-1416, 2010.
2. Jiuwen Cao, Zhiping Lin, and G.-B. Huang, "Composite function wavelet neural networks with differential evolution and extreme learning machine," *Neural Processing Letters*, vol. 33, no. 3, pp. 251-265, 2011.
3. Jiuwen Cao, Zhiping Lin, G.-B. Huang, and Nan Liu, "Voting based extreme learning machine," *Information Sciences*, vol. 185, no. 1, pp. 66-77, 2012.
4. Jiuwen Cao, Zhiping Lin, and G.-B. Huang, "Self-adaptive evolutionary extreme learning machine," *Neural Processing Letter*, vol. 36, no. 3, pp. 285-305, 2012.
5. N. Liu, Zhiping Lin, Jiuwen Cao, Z. Koh, T. Zhang, G.-B. Huang, W. Ser, and Marcus E. H. Ong, "A distance based scoring system for cardiac arrest prediction," *IEEE Transactions on Information Technology in BioMedicine*, vol. 16, no. 6, pp. 1324-1331, 2012.

-
6. Jiuwen Cao and Zhiping Lin, “Bayesian signal detection with compressed measurements,” submitted to *IEEE Transactions on Information Theory*, under review.

Conference:

1. Jiuwen Cao and Zhiping Lin, “The Detection Bound of the Probability of Error in Compressed Sensing Using Bayesian Approach,” in *Proceedings of 2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, Seoul, Korea, May 20-23.
2. Jiuwen Cao, Zhiping Lin, and G.-B. Huang, Voting base Online Sequential Extreme Learning Machine for Multi-class Classification, in *Proceedings of 2013 IEEE International Symposium on Circuits and Systems (ISCAS)*, Beijing, China, May 19-23, accepted for lecture session.

References

- [1] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: a new learning scheme of feedforward neural networks,” in *Proceedings of International Joint Conference on Neural Networks*, pp. 985–990, 2004.
- [2] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: theory and applications,” *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.
- [3] G.-B. Huang, D. Wang, and Y. Lan, “Extreme learning machines: a survey,” *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.
- [4] D. L. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [5] E. J. Candès and T. Tao, “Decoding by linear programming,” *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.
- [6] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

- [7] E. J. Candès and T. Tao, “Near-optimal signal recovery from random projections: universal encoding strategies?,” *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.
- [8] E. J. Candès, “Compressive sampling,” in *Proceedings of the International Congress of Mathematicians*, (Madrid, Spain), August 2006.
- [9] E. J. Candès, J. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.
- [10] Y. C. Eldar and M. Mishali, “Reduce and boost: recovering arbitrary sets of jointly sparse vectors,” *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 4692–4702, 2008.
- [11] Y. C. Eldar, “Compressed sensing of analog signals in shift-invariant spaces,” *IEEE Transactions on Signal Processing*, vol. 57, no. 8, pp. 2986–2997, 2009.
- [12] J. Bernardo and A. Smith, *Bayesian Theory*. John Wiley & Sons Inc.
- [13] S. Haykin, *Neural Networks, A Comprehensive Foundation*. New York, USA: Pearson education Press, 2001.
- [14] D. F. Specht, “A general regression neural network,” *IEEE Transactions on Neural Networks*, vol. 2, no. 6, pp. 568–576, 1991.
- [15] Y.-J. Qu and B.-G. Hu, “Generalized constraint neural network regression model subject to linear priors,” *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 2447–2459, 2011.

- [16] G. P. Zhang, “Neural networks for classification: a survey,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 30, no. 4, pp. 451–462, 2000.
- [17] G. Ou and Y. L. Murphey, “Multi-class pattern classification using neural networks,” *Pattern Recognition*, vol. 40, no. 1, pp. 4–18, 2007.
- [18] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, “Extreme learning machine for regression and multiclass classification,” *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.
- [19] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press, 1995.
- [20] S. Knerr, L. Personnaz, and G. Dreyfus, “Handwritten digit recognition by neural networks with single-layer training,” *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 962–968, 1992.
- [21] R. Lippmann, “Review of neural networks for speech recognition,” *Neural Computation*, vol. 1, pp. 1–38, 1989.
- [22] K. J. Hunt, D. G. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, “Neural networks for control systems - a survey,” *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.
- [23] E. Levin, R. Gewirtzman, and G. F. Inbar, “Neural network architecture for adaptive system modeling and control,” *Neural Networks*, vol. 4, no. 2, pp. 185–191, 1991.

- [24] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
- [25] D. Wang and G.-B. Huang, "Protein sequence classification using extreme learning machine," in *Proceedings of International Joint Conference on Neural Networks*, vol. 3, (Montreal, Canada), pp. 1406–1411, 2005.
- [26] N.-Y. Liang, P. Saratchandran, G.-B. Huang, and N. Sundararajan, "Classification of mental tasks from eeg signals using extreme learning machines," *International Journal of Neural Systems*, vol. 16, no. 1, pp. 29–38, 2006.
- [27] G.-B. Huang, Q.-Y. Zhu, K.-Z. Mao, C.-K. Siew, P. Saratchandran, and N. Sundararajan, "Can threshold networks be trained directly?," *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 3, pp. 187–191, 2006.
- [28] R. Zhang, G.-B. Huang, N. Sundararajan, and P. Saratchandran, "Multi-category classification using extreme learning machine for microarray gene expression cancer diagnosis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, no. 3, pp. 485–495, 2007.
- [29] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate on-line sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.
- [30] H.-J. Rong, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "On-line sequential fuzzy extreme learning machine for function approximation and classification problems," *IEEE Transactions on Systems, Man, and Cybernetics: Part B*, vol. 39, no. 4, pp. 1067–1072, 2009.

REFERENCES

- [31] G.-B. Huang, L. Chen, and C.-K. Siew, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.
- [32] G.-B. Huang and L. Chen, “Convex incremental extreme learning machine,” *Neurocomputing*, vol. 70, no. 16-18, pp. 3056–3062, 2007.
- [33] G.-B. Huang and L. Chen, “Enhanced random search based incremental extreme learning machine,” *Neurocomputing*, vol. 71, no. 16-18, pp. 3460–3468, 2008.
- [34] Y. Lan, Y. C. Soh, and G.-B. Huang, “Ensemble of online sequential extreme learning machine,” *Neurocomputing*, vol. 72, no. 13-15, pp. 3391–3395, 2009.
- [35] J. Cao, Z. Lin, and G.-B. Huang, “Composite function wavelet neural networks with extreme learning machine,” *Neurocomputing*, vol. 73, no. 7-9, pp. 1405–1416, 2010.
- [36] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, “Op-elm: Optimally pruned extreme learning machine,” *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 158–162, 2010.
- [37] J. Cao, Z. Lin, and G.-B. Huang, “Composite function wavelet neural networks with differential evolution and extreme learning machine,” *Neural Processing Letters*, vol. 33, no. 3, pp. 251–265, 2011.
- [38] E. Soria-Olivas, J. Gmez-Sanchs, J. D. Martn-Guerrero, J. Vila-Francis, M. Martnez-Sober, J. R. M. Benedito, and A. J. Serrano, “Belm: Bayesian

- extreme learning machine.,” *IEEE Transactions on Neural Networks*, vol. 22, no. 3, pp. 505–509, 2011.
- [39] J. Cao, Z. Lin, G.-B. Huang, and N. Liu, “Voting based extreme learning machine,” *Information Sciences*, vol. 185, no. 1, pp. 66–77, 2012.
- [40] J. Cao, Z. Lin, and G.-B. Huang, “Self-adaptive evolutionary extreme learning machine,” *Neural Processing Letters*, vol. 36, no. 3, pp. 285–305, 2012.
- [41] V. Vapnik, *The Nature of Statistical Learning Theory*. NY: Springer.
- [42] G.-B. Huang and C.-K. Siew, “Extreme learning machine: Rbf network case,” in *Proceedings of the Eighth International Conference on Control, Automation, Robotics and Vision*, vol. 2, (Kunming, China), pp. 1029–1036, 2004.
- [43] D. Serre, *Matrices: theory and applications*. New York, USA: Springer, 2002.
- [44] H. Nyquist, “Certain topics in telegraph transmission theory,” *Reprint as classic paper in Proceedings of the IEEE*, vol. 90, no. 2, pp. 280–305, 2002.
- [45] C. E. Shannon, “Communication in the presence of noise,” *Reprint as classic paper in Proceedings of the IEEE*, vol. 86, no. 2, pp. 447–457, 1998.
- [46] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” in *Proceeding of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA, USA).
- [47] E. J. Candès and T. Tao, “The dantzig selector: Statistical estimation when p is much larger than n ,” *Annals of Statistics*, vol. 35, no. 6, pp. 2313–2351, 2007.

REFERENCES

- [48] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.
- [49] J. Tropp and A. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [50] D. Needell and J. A. Tropp, “Cosamp: Iterative signal recovery from incomplete and inaccurate samples,” *Communications of the ACM*, “*Research Highlights*.”
- [51] D. Needell and R. Vershynin, “Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit,” *Foundations of Computational Mathematics*, vol. 9, no. 3, pp. 317–334, 2009.
- [52] A. Cohen, R. DeVore, and W. Dahmen, “Compressed sensing and best k -term approximation,” *Journal of the American Mathematical Society*, vol. 22.
- [53] T. Blumensath and M. E. Davies, “Iterative thresholding for sparse approximations,” *The Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 629–654, 2008.
- [54] T. Blumensath and M. E. Davies, “Gradient pursuits,” *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2370–2382, 2008.
- [55] T. Blumensath and M. E. Davies, “Stagewise weak gradient pursuits,” *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4333–4346, 2009.

- [56] G. Peyr, “Best basis compressed sensing,” *IEEE Transactions on Signal Processing*, vol. 58, no. 5, pp. 2613–2622, 2010.
- [57] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, “Least angle regression,” *Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.
- [58] J. Friedman, T. Hastie, and R. Tibshirani, “Sparse inverse covariance estimation with the graphical lasso,” *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.
- [59] J. Tropp and A. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [60] M. Duarte, M. Davenport, M. Wakin, and R. Baraniuk, “Sparse signal detection from incoherent projections,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, (Toulouse).
- [61] R. Castro, J. Haupt, and R. Nowak, “Compressive sensing vs. active learning,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, (Toulouse).
- [62] J. Haupt, R. Castro, R. Nowak, G. Fudge, and A. Yeh, “Compressive sampling for signal classification,” in *Proceedings of Fortieth Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA).
- [63] J. Haupt and R. Nowak, “Compressive sampling for signal detection,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, (Honolulu, HI).

REFERENCES

- [64] J. Wright, A. L. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.
- [65] M. Duarte, M. Davenport, M. Wakin, and R. Baraniuk, “Detection and estimation with compressive measurements,” tech. rep., Department of Electrical and Computer Engineering, Rice University, Houston, TX.
- [66] M. Duarte, M. Davenport, M. Wakin, and R. Baraniuk, “Signal processing with compressive measurements,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 445–460, 2010.
- [67] J. Neyman and E. Pearson, “On the problem of the most efficient tests of statistical hypotheses,” *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 231.
- [68] S. K. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*. Prentice Hall.
- [69] J. Daugmann, “Complete discrete 2-d gabor transforms by neural networks for image analysis and compression,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, no. 7, pp. 1169–1178, 1988.
- [70] Y. C. Pati and P. S. Krishnaprasad, “Analysis and synthesis of feed-forward neural networks using discrete affine wavelet transformations,” *IEEE Transactions on Neural Networks*, vol. 4, no. 1, pp. 73–85, 1992.
- [71] Y. C. Pati and P. S. Krishnaprasad, “Discrete affine wavelet transforms for

- analysis of feed-forward neural networks,” in *Proceedings of the 1990 Conference on Advances in Neural Information Processing Systems 3*, pp. 743–749, 1991.
- [72] Q. Zhang and A. Benveniste, “Wavelet networks,” *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 889–898, 1992.
- [73] H. Szu, B. Telfer, and S. Kadambe, “Neural network adaptive wavelets for signal representation and classification,” *Optical Engineering*, vol. 31, no. 9, pp. 1907–1916, 1992.
- [74] H. Szu, B. Telfer, and J. Garcia, “Wavelet transforms and neural networks for compression and recognition,” *Neural Networks*, vol. 9, no. 4, p. 695C708, 1996.
- [75] H. Szu, C. Hsu, and D. H. Xie, “Continuous speech segmentation determined by blind source separation,” in *Proceedings of the International Society of Optical Engineering*, vol. 3391, (Kunming, China), pp. 396–408, 1998.
- [76] V. Kruger, A. Happe, and G. Sommer, “Affine real-time face tracking using a wavelet network,” in *Proceeding Workshop on Recognition, Analysis and Tracking of Face and Gestures in Real-time Systems*, pp. 141–148, 1999.
- [77] A. Subasi, M. Yilmaz, and H. R. Ozcalik, “Classification of emg signals using wavelet neural network,” *Journal of Neuroscience Methods*, vol. 156, no. 1-2, pp. 360–367, 2006.
- [78] J. Echauz, “Strategies for fast training of wavelet neural networks,” in *Pro-*

- ceedings of the 2nd International Symposium on Soft Computing for Industry, 3rd World Automation Congress*, (Alaska, USA), pp. 1–6, 1998.
- [79] S. S. Rao and B. R. Kumthekar, “Recurrent wavelet networks,” in *Proceedings of IEEE Signal Processing Workshop*, pp. 3143–3147, 1993.
- [80] Y. Yu, S. Tan, J. Vanderwalle, and E. Deprettere, “Near-optimal construction of wavelet networks for nonlinear system modeling,” in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 3, (Atlanta, GA), pp. 48–51, 1996.
- [81] Y. Oussar and G. Dreyfus, “Initialization by selection for wavelet network training,” *Neurocomputing*, vol. 34, no. 1, pp. 131–143, 2000.
- [82] C. W. Hsu and C. J. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.
- [83] B. Zhou, A. Shi, F. Cai, and Y. Zhang, “Wavelet neural networks for non-linear time series analysis,” in *Proceedings of International Symposium on Neural Networks, Lecture Notes in Computer Science*, pp. 165–192, 2004.
- [84] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [85] P. Goupillaud, A. Grossman, and J. Morlet, “Cycle-octave and related transforms in seismic signal analysis,” *Geoexploration*, vol. 23, no. 1, pp. 85–102, 1984.

REFERENCES

- [86] C. J. Lin, “Wavelet neural networks with a hybrid learning approach,” *Journal of Information Science and Engineering*, vol. 22, no. 4, pp. 1367–1387, 2006.
- [87] J. A. Leonard and M. A. Kramer, “Radial basis function networks for classifying process faults,” *IEEE Control Systems Magazine*, vol. 11, no. 3, pp. 31–38, 1991.
- [88] D. R. Wilson and T. R. Martinez, “Heterogeneous radial basis function networks,” in *Proceedings of IEEE International Conference on Neural Networks*, pp. 1263–1267, 1996.
- [89] J. Zhang, G. G. Walter, Y. Miao, and N. W. L. Wan, “Wavelet neural networks for function learning,” *IEEE Transactions on Signal Processing*, vol. 43, no. 6, pp. 1485–1497, 1995.
- [90] E. Romero and R. Alquezar, “A new incremental method for function approximation using feed-forward neural networks,” in *Proceedings of the 2002 International Joint Conference on Neural Networks*, pp. 1968–1973, 2002.
- [91] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” in *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 148–156, 1996.
- [92] C. Blake and C. Merz, “Uci repository of machine learning databases,” in <http://www.ics.uci.edu/mlearn/MLRepository.html>, *Department of Information and Computer Sciences, University of California, Irvine, USA*, 1998.
- [93] R. Storn and K. Price, “Differential evolution - a simple and efficient heuristic

- for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 2004.
- [94] Q.-Y. Zhu, A.-K. Qin, P.-N. Suganthan, and G.-B. Huang, “Evolutionary extreme learning machine,” *Pattern Recognition*, vol. 38, no. 10, pp. 1759–1763, 2005.
- [95] B. Subudhi and D. Jena, “Differential evolution and levenberg marquardt trained neural network scheme for nonlinear system identification,” *Neural Processing Letter*, vol. 27, pp. 285–296, 2008.
- [96] P. L. Bartlett, “The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network,” *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, 1998.
- [97] P. Lingras and C. Butz, “Rough set based 1-v-1 and 1-v-r approaches to support vector machine multi-classification,” *Information Sciences*, vol. 177, no. 18, pp. 3782–3798, 2007.
- [98] C.-W. Hsu and C.-J. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.
- [99] R. Polikar, “Ensemble based systems in decision making,” *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–45, 2006.
- [100] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, “Op-

REFERENCES

- elm: Optimally pruned extreme learning machine,” *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 158–162, 2010.
- [101] K. Levenberg, “A method for the solution of certain non-linear problems in least squares,” *The Quarterly of Applied Mathematics*, vol. 2.
- [102] D. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” *SIAM Journal on Applied Mathematics*, vol. 11.
- [103] D. Bremner, E. Demaine, J. Erickson, J. Iacono, S. Langerman, P. Morin, and G. Toussaint, “Output-sensitive algorithms for computing nearest-neighbor decision boundaries,” *Discrete and Computational Geometry*, vol. 33, no. 4, p. 593C604, 2005.
- [104] T. M. Cover and P. E. Hart, “Nearest neighbor pattern classification,” *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.
- [105] W. L. Cai, S. C. Chen, and D. Q. Zhang, “Robust fuzzy relational classifier incorporating the soft class labels,” *Pattern Recognition Letters*, vol. 28, no. 16, pp. 2250–2263, 2007.
- [106] Z. R. Yang, “A novel radial basis function neural network for discriminant analysis,” *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 604–612, 2006.
- [107] W. L. Cai, S. C. Chen, and D. Q. Zhang, “A multiobjective simultaneous learning framework for clustering and classification,” *IEEE Transactions on Neural Networks*, vol. 21, no. 2, pp. 185–200, 2010.

REFERENCES

- [108] N. Liu and H. Wang, “Ensemble based extreme learning machine,” *IEEE Signal Processing Letters*, vol. 17, no. 8, pp. 754–757, 2010.
- [109] D. Wang, “Elm-based multiple classifier systems,” in *Proceedings of International Conference on 9th Control, Automation, Robotics and Vision*, (Singapore), December 2006.
- [110] R. Walter, *Principles of Mathematical Analysis*. New York, USA: McGraw-Hill, 1976.
- [111] L. Spafford, “Optimum radar signal processing in clutter,” *IEEE Transactions on Information Theory*, vol. 14, no. 5, pp. 734–743, 1968.
- [112] T. Horiuchi and R. Etienne-Cummings, “A time-series processor for sonar mapping and novelty detection,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, (Bangkok, Thailand), May 2003.
- [113] C. C. Wang, C. L. Chen, T. H. Yeh, Y. Hu, and G. N. Sungm, “A high speed transceiver front-end design with fault detection for flexray-based automotive communication systems,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, (Rio de Janeiro, Brazil), May 2011.
- [114] M. Barni, A. Costanzo, and L. Sabatini, “Identification of cut & paste tampering by means of double-jpeg detection and image segmentation,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, (Paris, France), May 2010.
- [115] T. Delbruck, T. Koch, R. Berner, and H. Hermansky, “Fully integrated 500uw

- speech detection wake-up circuit,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, (Paris, France), May 2010.
- [116] J. Cao and Z. Lin, “The detection bound of the probability of error in compressed sensing using bayesian approach,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, (Seoul, Korea), May 2012.
- [117] Z. Ben-Haim, Y. C. Eldar, and M. Elad, “Coherence-based near-oracle performance guarantees for sparse estimation under gaussian noise,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, (Dallas, TX, USA), March 2010.
- [118] Z. Ben-Haim, Y. C. Eldar, and M. Elad, “Coherence-based performance guarantees for estimating a sparse vector under random noise,” *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5030–5043, 2010.
- [119] Z. Ben-Haim and Y. C. Eldar, “The cramèr-rao bound for estimating a sparse parameter vector,” *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3384–3389, 2010.
- [120] B. David and L. Trefethen, *Numerical linear algebra*. Philadelphia, USA: Society for Industrial and Applied Mathematics, 1997.
- [121] S. Ji, Y. Xue, , and L. Carin, “Bayesian compressive sensing,” *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.
- [122] L. He and L. Carin, “Exploiting structure in wavelet-based bayesian compressive sensing,” *IEEE Transactions on Signal Processing*, vol. 57, no. 9, pp. 3488–3497, 2009.

REFERENCES

- [123] S. D. Babacan, R. Molina, and A. K. Katsaggelos, “Bayesian compressive sensing using laplace priors,” *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 53–63, 2010.
- [124] S. Dasgupta and A. Gupta, “An elementary proof of the johnson-lindenstrauss lemma,” tech. rep., Berkeley, CA.
- [125] E. J. Candès, “The restrict isometry property and its implications for compressed sensing,” *Compte Rendus de l’Academie des Sciences, Paris, Serie I*, vol. 346.
- [126] D. L. Donoho and X. Huo, “Uncertainty principles and ideal atomic decomposition,” *IEEE Transactions on Information Theory*, vol. 47, no. 7, pp. 2845–2862, 2001.
- [127] T. Cai, L. Wang, and G. Xu, “Stable recovery of sparse signals and an oracle inequality,” *IEEE Transactions on Information Theory*, vol. 56, no. 7, pp. 3516–3522, 2010.
- [128] H. Chernoff, “A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations,” *Annals of Mathematical Statistics*, vol. 23, no. 4, pp. 493–507, 1952.
- [129] M. Kline, *Mathematical Thought from Ancient to Modern Times*. Oxford University Press.
- [130] S. Decherchi, P. Gastaldo, A. Leoncini, and R. Zunino, “Efficient digital implementation of extreme learning machines for classification,” *IEEE Transac-*

- tions on Circuits and Systems II: Express Briefs*, vol. 59, no. 8, pp. 496–500, 2012.
- [131] R. Calderbank, S. Jafarpour, and R. Schapire, “Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain,” <http://dsp.rice.edu/sites/dsp.rice.edu/files/cs/cl.pdf> (Preprint), 2009.
- [132] M. Richards, *Fundamentals of Radar Signal Processing*. New York, USA: McGraw-Hill, 2005.
- [133] J. Guerci, *Space-Time Adaptive Processing for Radar*. Norwood, MA: Artech House, 2003.
- [134] M. Soumekh, *Synthetic Aperture Radar Signal Processing With MATLAB Algorithms*. New York, USA: J. Wiley, 1999.
- [135] R. J. Purdy, P. E. Blankenship, C. E. Muehe, C. M. Rader, E. Stern, and R. C. Williamson, “Radar signal processing,” *Lincoln Laboratory Journal*, vol. 12, no. 2, pp. 297–320, 2000.
- [136] R. Baraniuk and P. Steeghs, “Compressive radar imaging,” in *Proceedings of IEEE Radar Conference*, (Boston, USA), April 2007.
- [137] M. A. Herman and T. Strohmer, “High-resolution radar via compressed sensing,” *IEEE Transactions on Signal Processing*, vol. 57, no. 6, pp. 2275–2284, 2009.
- [138] Q. Liang, “Compressive sensing for radar sensor networks,” in *Proceedings of the 2010 IEEE Global Telecommunications Conference*, (Miami, USA), Dec 2010.

REFERENCES

- [139] G. E. Smith, T. Diethe, Z. Hussain, J. Shawe-Taylor, and D. R. Hardoon, "Compressed sampling for pulse doppler radar," in *Proceedings of IEEE Radar Conference*, (Washington, USA), May 2010.
- [140] C. R. Berger, S. Zhou, P. Willett, B. Demissie, and J. Heckenbach, "Compressed sensing for ofdm/mimo radar," in *Proceedings of the 42nd Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA), Oct 2008.
- [141] C.-Y. Chen and P. P. Vaidyanathan, "Compressed sensing in mimo radar," in *Proceedings of the 42nd Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA), Oct 2008.
- [142] C. R. Berger, S. Zhou, and P. Willett, "Signal extraction using compressed sensing for passive radar with ofdm signals," in *Proceedings of the 11th International Conference on Information Fusion*, (Cologne), Jun 2008.
- [143] T. Strohmer and B. Friedlander, "Compressed sensing for mimo radar - algorithms and performance," in *Proceedings of the 43rd Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA), Nov 2009.
- [144] S. Gogineni and A. Nehorai, "Target estimation using compressive sensing for distributed mimo radar," in *Proceedings of the 44th Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA), Nov 2010.
- [145] Y. Yu, A. P. Petropulu, and H. V. Poor, "Robust target estimation in compressive sensing based colocated mimo radar," in *Proceedings of the 2010 Military Communications Conference*, (San Jose, CA), Oct 2010.

REFERENCES

- [146] V. M. Patel, G. R. Easley, D. M. Healy, and R. Chellappa, “Compressed sensing for synthetic aperture radar imaging,” in *Proceedings of the 16th IEEE International Conference on Image Processing*, (Cairo), Nov 2009.
- [147] M. T. Alonso, P. Lopez-Dekker, and J. J. Mallorqui, “A novel strategy for radar imaging based on compressive sensing,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 12, pp. 4285–4295, 2012.
- [148] I. Kyriakides, “Ambiguity functions of compressively sensed and processed radar waveforms,” in *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing*, (Prague), May 2011.
- [149] I. Kyriakides, “Adaptive compressive sensing and processing of delay-doppler radar waveforms,” *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 730–739, 2012.