On a multiple nodes fault tolerant training for RBF: Objective function, sensitivity analysis and relation to generalization

John SUM
Department of Information Management, Chung Shan Medical University
Taichung, 402, Taiwan, ROC.

Abstract

Over a decades, although various techniques have been proposed to improve the training of a neural network to against node fault, there is still a lacking of (i) a simple objective function to formalize multiple nodes fault and not much work has been done on understanding of the relation between fault tolerant and generalization. In this paper, an objective function based on the idea of Kullback-Leibler divergence is presented for multiple nodes fault tolerant training. It is essentially the same as a summation of mean square errors plus a regularizer. A simple training algorithm for attaining fault tolerant neural network is presented accordingly and its gracefully performance degradation is shown by simulation results. Besides, the sensitivity of the training algorithm against node fault rate is analyzed and its insensitivity is demonstrated by simulation results. Finally, a discussion on fault tolerant and generalization is presented and the incapability of using regularizers for improving generalization to achieve optimal fault tolerant is commented.

1 Introduction

Obtaining a neural network to tolerate random node fault is of paramount important. It is because node fault is an unavoidable factor while a neural network is implemented [12], either by analog or digital components. Thus could lead to a drastic performance degradation even when a neural network has been well trained. In view of the importance of making a neural network being fault tolerant, many researchers have developed various robust learning algorithm throughout the last decade in order to attain a fault tolerant neural network against random node fault [3, 8].

One approach to robust learning is based on adding heuristic in the training algorithm in order to force the internal representation ability of a neural network distributed widely within the hidden nodes or weights. So that, no single node or single weight is particularly important. A random removal of a node or a weight will only *gracefully degrade* the performance of the network. For this approach, injecting random node fault [17, 1] together with random node deletion and addition [5] during training is one technique. Adding network redundancy by replicating multiple hidden layers after a neural network has been well trained [7, 14] is another one. One more technique is to limit the weight magnitude to small value so that no single weight is extra sensitive within the network. Adding weight decay regularizer [5] and hard bound the weight magnitude during training [2] are two examples. In accordance with simulation results, all these heuristic techniques have demonstrated that the network is able to tolerate against random node fault, either single node fault or multiple nodes fault. As these techniques are heuristics, it is not clear theoretically about the underlying objective function that they are going to achieve. In sequel, analysis and comparison on the similarities and differences between one technique to another can hardly be accomplished except by extensive simulations.

Another approach is to formulate the learning directly as a constraint optimization problem. Neti *et al* [13] defined the problem as a minimax problem in which the objective function to be minimized is maximum of the mean square errors over all fault models. Similarly, Deodhare *et al* [6] formulated the problem by defining the objective function to be minimized as the maximum square error over all fault models and all training samples. As solving these minmax problem is complex, Simon & El-Sherief [18] and Phatak & Tcherner in [16] formulated the learning problem as an unconstraint optimization problem in which the objective function consists of two terms. The first term is the mean square errors of the fault-free model while the second term is the ensemble average of the mean square errors over all fault models. One limitation of these formulations is that the problem being formulated can be very complicated when the number of fault models is large. Extend their formulations to handling multiple nodes fault will become impractical.

Although improving the fault tolerance of a neural network has been researching for more than a decade, research on the relation between fault tolerance and generalization is scares. Only Phatak [15] did explain why adding redundancy can improve node fault tolerant from the VC dimension point of view. Except that, no other work has been done along the line. As one can realize that regularization technique like weight decay can be applied to improve both fault tolerance [5] and generalization [11]. On the other hand, many results have been showing that a neural network being trained to be fault tolerant exhibits improvement in generalization. What actually is the relation between a fault tolerant problem and a generalization problem ? Besides, what is the reason leading to some techniques can be applied to solve both problems ?

In view of the lacking of a simple objective function to formalize multiple nodes fault and the lacking of an understanding of the relation between fault tolerant and generalization, the focus of the paper will be on the following problems :

P1 Derive from the probabilistic approach an objective function for robust training a neural network that can optimally tolerate multiple nodes fault,

P2 Make use of the objective function being derived to study capability of regularization techniques in the multiple nodes fault problem.

Throughout the paper, we will use radial basis function (RBF) network as an example. Extend of the results to other neural network models will not be covered in this paper.

In the next section, the objective function for multiple nodes fault will be derived. The performance of an RBF being trained by the derived objective function will be elucidated and compared with the same network being trained by pseudo inverse in Section 3. The sensitivity of the training method against inaccurate fault rate information will be analyzed in Section 4. Section 5 gives a discussion on the techniques applying in attaining good fault tolerance and good generalization. The conclusion will be presented in Section 6.

2 Multiple nodes fault training

Throughout the paper, we assume that training data set $D_T = \{(x_k, y_k)\}_{k=1}^N$ is extracted from an unknown stochastic system defined as follows : For $k = 1, 2, \cdots, N$

$$y_k = f(x_k) + e_k \tag{1}$$

where x and y are the input and the output of an unknown deterministic system $f(x)$; and e is a random measurement noise. Consider the output of the system is a dependent random variable governed by the input x, the behavior of the system can be denoted by the conditional probability $P_0(y|x)$.

RBF model An RBF network consisting of M hidden nodes is defined as follows :

$$\hat{f}(x, \theta) = \sum_{i=1}^{M} \theta_i \phi_i(x), \tag{2}$$

where $\phi_i(x)$ for all $i = 1, 2, \cdots, M$ are the radial basis functions given by

$$\phi_i(x) = \exp\left(-\frac{(x - c_i)^2}{\sigma}\right); \tag{3}$$

c_is are the radial basis function centers and the positive parameter $\sigma > 0$ controls the width of the radial basis functions. We call the model (2) the fault-free RBF network.

Next, we assume that a node fault is equivalent to permanently set the output of the node zero. Therefore, a faulty RBF $\hat{f}(x, \theta, \beta)$ could be defined by multiplying each $\phi_i(x)$ by a random binary variable β_i :

$$\hat{f}(x, \theta, \beta) = \sum_{i=1}^{M} \beta_i \theta_i \phi_i(x). \tag{4}$$

When $\beta_i = 1$, the i^{th} node is normal. When $\beta_i = 0$, the i^{th} node is fault. We assume that all nodes are of equal fault rate p, i.e. $P(\beta_i = 0) = p$ and $P(\beta_i = 1) = 1 - p$ for $i = 1, 2, \cdots, M$ and $\beta_1, \cdots, \beta_M$ are independent random variables.

In sequel, we approximate the unknown system (1) by

$$y_k = \hat{f}(x_k, \theta, \beta) + e_k, \tag{5}$$

where e is a mean zero Gaussian noise defined in (1). Similarly, we can represent the behavior of this faulty RBF by a conditional probability $P(y|x, \theta, \beta)$. For notational simplicity, we let $\tilde{\theta}_i = \beta_i \theta_i$ and thus the conditional probability of the faulty RBF given x as input is denoted by $P(y|x, \tilde{\theta})$.

Objective function Let $P_0(x)$ be probability distribution of input x, the joint probability distributions for the unknown system and the faulty RBF can be given by

$$\begin{aligned} P_0(x, y) &= P_0(y|x) P_0(x), \\ P(x, y|\tilde{\theta}) &= P(y|x, \tilde{\theta}) P_0(x). \end{aligned}$$

To measure the discrepancy between a faulty RBF $\tilde{\theta}$ and the unknown system, one can apply the Kullback-Leibler divergence [9] :

$$D(P_0||P_{\tilde{\theta}}) = \int\int P_0(x,y) \log \frac{P_0(x,y)}{P(x,y|\tilde{\theta})} dxdy. \quad (6)$$

Since $\tilde{\theta}$ is an unknown and it is depended on the fault-free model θ, the average distance of all fault models (all possible $\beta \in \{0,1\}^M$) with reference to the true distribution $P_0(x,y)$ can be defined as

$$\bar{D}(P_0||P_\theta) = \left\langle \int\int P_0(x,y) \log \frac{P_0(x,y)}{P(x,y|\tilde{\theta})} dxdy \right\rangle_{\Omega_\beta}. \quad (7)$$

Here Ω_β corresponds to the set consisting all the possible β. Due to page limit, the objective function and the corresponding regularizer are stated without proof by the following lemma.

Lemma 1 *The objective function for attaining an optimal fault tolerant RBF against multiple nodes fault with fault rate p is given by*

$$\begin{aligned} E(\theta,p) &= \frac{1}{N}\sum_{k=1}^{N} y_k^2 - 2(1-p)\frac{1}{N}\sum_{k=1}^{N} y_k \phi^T(x_k)\theta \\ &\quad +(1-p)\theta^T \left\{(1-p)\hat{H}_\phi + p\hat{G}\right\}\theta, \end{aligned}$$

where where $\hat{H}_\phi = \frac{1}{N}\sum_{k=1}^{N}\phi(x_k)\phi^T(x_k)$ and $\hat{G} = \mathbf{diag}\left\{\frac{1}{N}\sum_{k=1}^{N}\phi_1^2(x_k), \cdots, \frac{1}{N}\sum_{k=1}^{N}\phi_M^2(x_k)\right\}$. The implicit regularizer is given by $p\theta^T(\hat{G}-\hat{H}_\phi)\theta$.

Taking derivative the $E(\theta,p)$ with respect to θ and setting it to zero, the optimal fault tolerant RBF $\hat{\theta}$ can be obtained as follows :

$$\hat{\theta} = \left(\hat{H}_\phi + p\left(\hat{G} - \hat{H}_\phi\right)\right)^{-1} \frac{1}{N}\sum_{k=1}^{N} y_k \phi(x_k). \quad (8)$$

Observe that $\hat{\theta}$ above is also the solution of

$$J(\theta,p) = \frac{1}{N}\sum_{k=1}^{N}\left(y_k - \phi^T(x_k)\theta\right)^2 + \theta^T\Sigma\theta, \quad (9)$$

where $\Sigma = p(G - H_\phi)$, minimizing $E(\theta,p)$ is equivalent to minimizing the mean square training errors $N^{-1}\sum_{k=1}^{N}\left(y_k - \phi^T(x_k)\theta\right)^2$ plus an additional regularizer term $\theta^T\Sigma\theta$.

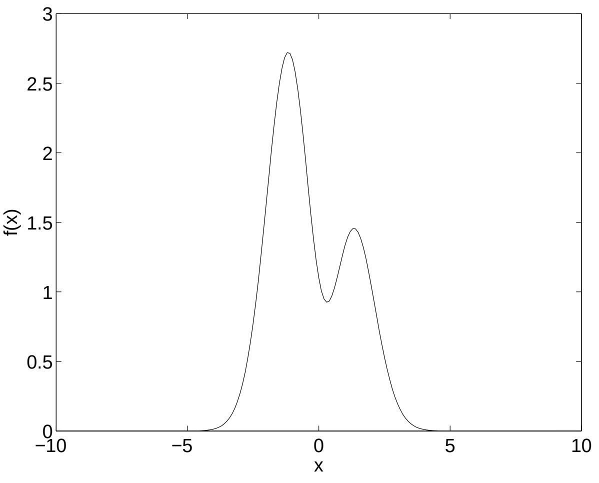

Figure 1: Nonlinear function.

3 Simulation

To demonstrate the performance of the derived algorithm, we use the following Hermite function for illustration.

$$f(x) = 1.1(1 - x + 2x^2)\exp(-x^2/2), \quad (10)$$

where $x \in [-10,10]$. The measured output y_k of a given x_k is generated by adding noise to this noise free function, i.e.

$$y_k = \underbrace{1.1(1 - x_k + 2x_k^2)\exp(-x_k^2/2)}_{f(x_k)} + e_k,$$

where e_k the noise term is a mean zero Gaussian noise of variance S_e. The shape of the noise free function (10) is shown in Figure 1. This function fluctuates mainly in the middle portion around the origin, i.e. $x \in [-5,5]$. For $|x| > 5$, the value of $f(x)$ is almost zero. Figure 2 shows an exemplar training data set of noise variance 0.01. We assume an RBF network is of 37 radial basis functions :

$$\hat{f}(x) = \sum_{i=1}^{37}\theta_i\phi_i(x) = \sum_{i=1}^{37}\theta_i \exp\left(-\frac{(x-c)^2}{\sigma}\right), \quad (11)$$

where $\sigma = 0.49$ and the centers c_is are $\{-9, -8.5, -8, \cdots, 8, 8.5, 9\}$.

Procedure In the simulation, a set of training data $\mathcal{D}_T$ and a set of testing data $\mathcal{D}_F$ have been given. The first set is used for obtaining $\hat{\theta}$ while the second set is used for validation. For presentation clarity, the notations being used is summarized below.

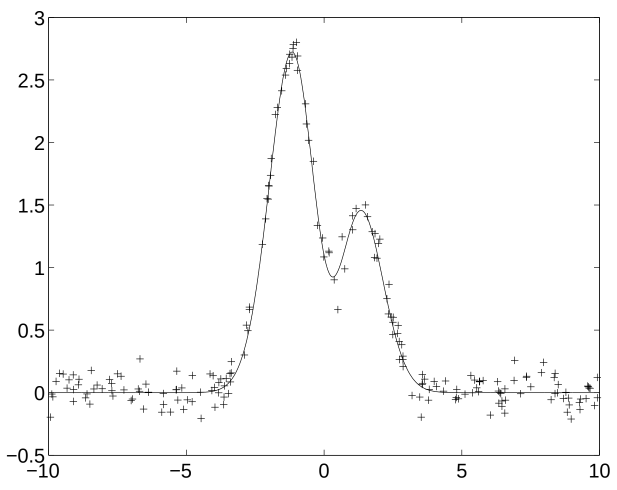

Figure 2: A noisy function with noise variance 0.01 and mean zero.

Notation	Description
$\mathcal{D}_T$	Training data set
$\mathcal{D}_F$	Testing data set
p	Fault rate
M	Number of RBFs
$\mathcal{I}$	Index set for the nodes
$\mathcal{K}$	Set of nodes being removed
$\mathcal{I}-\mathcal{K}$	Remaining node set
$\hat{\theta}(p,\mathcal{I})$	Model obtained by (8)
$\hat{\theta}_{PI}(\mathcal{I})=\hat{\theta}(0,\mathcal{I})$	Model obtained by PI
$\hat{\theta}(p,\mathcal{I}-\mathcal{K})$	$\hat{\theta}(p,\mathcal{I}-\mathcal{K})_i=\hat{\theta}(p,\mathcal{I})_i$ if i is not in $\mathcal{K}$; zero otherwise
$\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K})$	$\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K})_i=\hat{\theta}_{PI}(\mathcal{I})_i$ if i is not in $\mathcal{K}$; zero otherwise
$E(\mathcal{D}_T\vert\hat{\theta}(p,\mathcal{I}-\mathcal{K}))$	Mean square training errors due to $\hat{\theta}(p,\mathcal{I}-\mathcal{K})$
$E(\mathcal{D}_T\vert\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))$	Mean square training errors due to $\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K})$
$E(\mathcal{D}_F\vert\hat{\theta}(p,\mathcal{I}-\mathcal{K}))$	Mean square testing errors due to $\hat{\theta}(p,\mathcal{I}-\mathcal{K})$
$E(\mathcal{D}_F\vert\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))$ due to $\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K})$	Mean square testing errors
$\langle E(\cdot\vert\cdot)\rangle$	Average mean square errors

	$\hat{\theta}(p,\mathcal{I})$	$\hat{\theta}(p,\mathcal{I}-\mathcal{K})$
$\mathcal{D}_T$	$E(\mathcal{D}_T\vert\hat{\theta}(p,\mathcal{I}))$	$E(\mathcal{D}_T\vert\hat{\theta}(p,\mathcal{I}-\mathcal{K}))$
$\mathcal{D}_F$	$E(\mathcal{D}_F\vert\hat{\theta}(p,\mathcal{I}))$	$E(\mathcal{D}_F\vert\hat{\theta}(p,\mathcal{I}-\mathcal{K}))$

To show its tolerance effectiveness, the performance of the RBF generated by robust learning (8) is compared with the network generated by pseudo inverse, i.e.

$$\hat{\theta}_{PI}(\mathcal{I})=\left(\sum_{k=1}^{N}\phi(x_k)\phi^T(x_k)\right)^{-1}\left(\sum_{k=1}^{N}f(x_k)\phi(x_k)\right).$$

To observe the fault tolerance behavior of the RBF being trained by pseudo inverse and the RBF being trained by our method, the procedure shown in Figure 3 is taken.

Results The resultant training errors $\langle E(\mathcal{D}_T|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))\rangle$ and $\langle E(\mathcal{D}_T|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))\rangle$ are plotted against p and shown in Figure 4. Here, $S_e=0.04$ are shown. While the resultant testing errors $\langle E(\mathcal{D}_F|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))\rangle$ and $\langle E(\mathcal{D}_F|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))\rangle$ are plotted against p and shown in Figure 5. It is observed that whenever p increase, the performance of the RBF obtained by the robust learning only gracefully degrades. This can be explained by comparing the weight magnitude of the models obtained by pseudo inverse and robust learning.

On the other hand, the RBF network obtained by our robust learning does not suffer from large weight magnitude problem. The regularizer has the proper controlled the weight magnitude to small value but not very small so as to maintain a good shape of the reconstructed function. We have also found that the magnitude is still controlled to a small value even though the measurement noise variance is large.

4 Sensitivity on fault rate p

In the above simulation, we have demonstrated that the RBF obtained by the robust learning can tolerate fault due to random node removal. The assumption is that the fault rate p is given. As we know that the fault rate p sometimes cannot be given precisely, it will be necessary to analyze its performance change against the uncertainty of p. Let $\bar{p}$ be the actual fault rate and $\hat{\theta}$ be the estimator obtained by assuming the fault rate is p. Let $E(D_F,\hat{\theta}(p,\mathcal{I}-\mathcal{K}),\bar{p})$ be the prediction error for the RBF trained under the assumption that the fault rate is p but the actual fault rate is $\bar{p}$. We can have the following lemma.

Lemma 2
$E(D_F,\hat{\theta}(p,\mathcal{I}-\mathcal{K}),\bar{p})\leq E(D_F,\hat{\theta}(p,\mathcal{I}-\mathcal{K}),p)$ if

$$\bar{p}<p<\frac{3+(M-1)\bar{p}}{M+1}.$$

(Proof) For simplicity, we denote $E(\hat{\theta},\bar{p})$ be $E(D_F,\hat{\theta}(p,\mathcal{I}-\mathcal{K}),\bar{p})$.

$$\begin{aligned}
& E(\hat{\theta},\bar{p})-E(\hat{\theta},p)\\
= \;& 2\left((1-p)-(1-\bar{p})\right)\left\langle f(x)\phi^T(x)\right\rangle_{\Omega_x}\hat{\theta}\\
& +(1-\bar{p})\hat{\theta}^T\left\{(1-\bar{p})H_\phi+\bar{p}G\right\}\hat{\theta}\\
& -(1-p)\hat{\theta}^T\left\{(1-p)H_\phi+pG\right\}\hat{\theta}. \qquad (12)
\end{aligned}$$

```
1 Generate $\hat{\theta}_{PI}(\mathcal{I})$ ($\hat{\theta}(0,\mathcal{I})$) using $D_T$
2 Calculate training error $E(\mathcal{D}_T|\hat{\theta}_{PI}(\mathcal{I}))$ ( $E(\mathcal{D}_T|\hat{\theta}(0,\mathcal{I}))$ )
3 Calculate testing error $E(\mathcal{D}_F|\hat{\theta}_{PI}(\mathcal{I}))$ ($E(\mathcal{D}_F|\hat{\theta}(p,\mathcal{I}))$ )
4 FOR $p = \{0, 0.01, 0.02, \cdots, 0.19, 0.2\}$
    – Generate $\hat{\theta}(p,\mathcal{I})$ using $D_T$ [**Robust learning**]
    – REPEAT 100000 times
        ∗ Set $\mathcal{K}$ to an empty list
        ∗ Generate random numbers $r_1, r_2, \cdots, r_M$ from uniform distribution $U[0,1]$
        ∗ Including $i \in \mathcal{K}$ if $r_i \le p$, for all $i = 1, 2, \cdots, M$
        ∗ FOR $i = 1, \cdots, M$,
              $\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K})_i = \begin{cases} \hat{\theta}_{PI}(\mathcal{I})_i & \text{if } i \notin \mathcal{K} \\ 0 & \text{otherwise} \end{cases}$
              $\hat{\theta}(p,\mathcal{I}-\mathcal{K})_i = \begin{cases} \hat{\theta}(p,\mathcal{I})_i & \text{if } i \notin \mathcal{K} \\ 0 & \text{otherwise} \end{cases}$
        ∗ Evaluate $E(\mathcal{D}_T|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))$, $E(\mathcal{D}_F|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))$
        ∗ Evaluate $E(\mathcal{D}_T|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))$, $E(\mathcal{D}_F|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))$
    – Evaluate the average errors $\langle E(\mathcal{D}_T|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))\rangle$, $\langle E(\mathcal{D}_F|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))\rangle$
    – Evaluate the average errors $\langle E(\mathcal{D}_T|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))\rangle$, $\langle E(\mathcal{D}_F|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))\rangle$
```

Figure 3: Simulation procedure. Note that the total number of faulty models can be as large as 66045 and 10.3×10^6 for the number of fault nodes are 4 ($p \approx 0.1$) and 7 ($p \approx 0.2$) respectively. Repeating 100000 times is only sufficient for $p \le 0.14$.

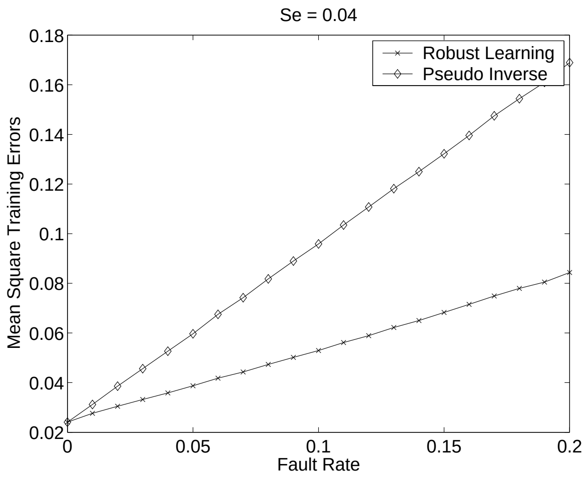

Figure 4: Comparison between the average mean square training error of the RBF network obtained by the proposed algorithm and by pseudo inverse. The solid line with diamonds corresponds to the result from pseudo inverse $\langle E(\mathcal{D}_T|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))\rangle$ while the solid line with crosses corresponds to the result from robust learning $\langle E(\mathcal{D}_T|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))\rangle$. In this simulation, $S_e = 0.04$.

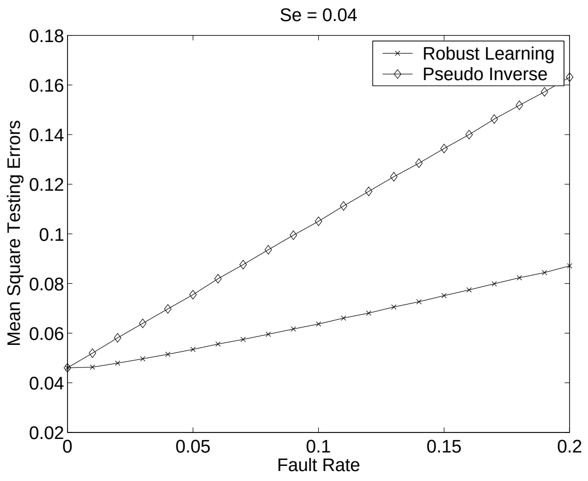

Figure 5: Comparison between the average mean square testing errors of the RBF network obtained by the proposed algorithm and by pseudo inverse. The solid line with diamonds corresponds to the result from pseudo inverse $\langle E(\mathcal{D}_F|\hat{\theta}_{PI}(\mathcal{I}-\mathcal{K}))\rangle$ while the solid line with crosses corresponds to the result from robust learning $\langle E(\mathcal{D}_F|\hat{\theta}(p,\mathcal{I}-\mathcal{K}))\rangle$. In this simulation, $S_e = 0.04$.

Write $\left\langle f(x)\phi^T(x)\right\rangle_{\Omega_x}\hat{\theta}$ in the first term of (12) as $\hat{\theta}^T\left\{(1-p)H_\phi + pG\right\}\hat{\theta}$, the difference between $E(\hat{\theta},\bar{p}) - E(\hat{\theta},p)$ can be given by

$$\begin{aligned} & E(\hat{\theta},\bar{p}) - E(\hat{\theta},p) \\ = \; & (\bar{p}-p)\hat{\theta}^T\left((\bar{p}-p)H_\phi + (3-(\bar{p}+p))G\right)\hat{\theta}. \end{aligned} \quad (13)$$

As we assume that $\bar{p}$ or p is not a large value. For $\bar{p}$ or p is too large, the network is basically very faulty. It is useless. Therefore, $E(\hat{\theta},\bar{p}) \geq E(\hat{\theta},p)$ whenever $\bar{p} \geq p$ as $\bar{p}+p$ cannot be larger than one. And the equality holds if and only if $\bar{p}=p$. For $\bar{p}<p$, it can readily be shown that

$$\begin{aligned} & \bar{\theta}^T\left[(\bar{p}-p)H_\phi + (3-(\bar{p}+p))G\right]\bar{\theta} \\ = \; & \bar{\theta}^T\left[(\bar{p}-p)(H_\phi - G) + (3-2p)G\right]\bar{\theta} \\ \leq \; & \bar{\theta}^T\left[(\bar{p}-p)(M-1)G + (3-2p)G\right]\bar{\theta} \end{aligned}$$

since

$$\begin{aligned} \hat{\theta}^T(H_\phi - G)\hat{\theta} &= \sum_{i=1}^{M}\sum_{j\neq i}\hat{\theta}_i\hat{\theta}_j\phi_i\phi_j \\ &\leq (M-1)\sum_{i=1}^{M}\left(\hat{\theta}_i\phi_i\right)^2 \\ &\leq (M-1)\hat{\theta}^T G\hat{\theta}. \end{aligned}$$

The condition for which $E(\hat{\theta},\bar{p}) \leq E(\hat{\theta},p)$ will be given by

$$\bar{p} < p < \frac{3+(M-1)\bar{p}}{M+1}. \quad (14)$$

And the proof is completed. **Q.E.D.**

For $M \gg 1$, the upper bound can be approximated by $3/M + \bar{p}$. The condition can thus be written as $\bar{p} < p < 3/M + \bar{p}$. That is to say, our robust learning is not quite sensitive to the value of p if the estimated p is in a range of about $3/M$ above the true $\bar{p}$. Figure 6 shows the results for the cases when measurement noise are 0.01 respectively and $\bar{p} = 0.8p$. Whenever the true $\bar{p}$ is smaller than p, it is found that $E(\hat{\theta}(p),p) > E(\hat{\theta}(p),0.8p) > E(\hat{\theta}(0.8p),0.8p)$ and $E(\hat{\theta}(p),0.8p) \approx E(\hat{\theta}(0.8p),0.8p)$. The difference between $E(\hat{\theta}(p),\bar{p})$ and $E(\hat{\theta}(\bar{p}),\bar{p})$ increase as the difference between p and $\bar{p}$ increases or S_e increases.

Furthermore, we have observed that the bound $(3+(M-1)\bar{p})/(M+1)$ in fact is a rather tight bound. In practice, $\hat{\theta}^T(H_\phi - G)\hat{\theta}$ is much smaller than $(M-1)\hat{\theta}^T G\hat{\theta}$. Hence, one can almost set p to a much larger value during training. After training, the performance of the network under node fault can be evaluated by simulation. The mean square testing errors can thus be used as an upper bound on the performance of the network in actual situation.

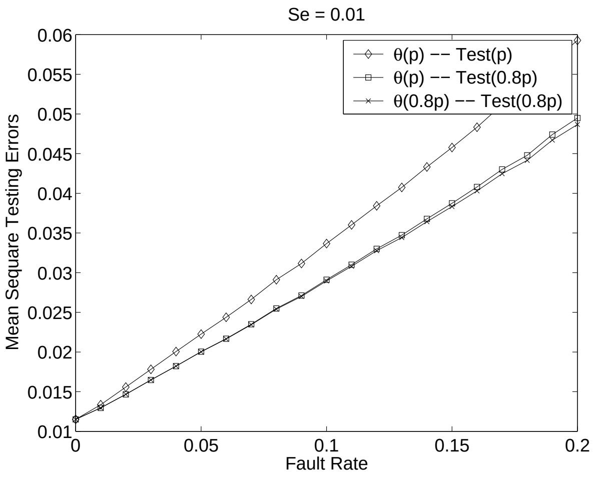

Figure 6: The performance of the robust learning whenever the true fault rate probability $\bar{p}$ is unknown but below p. The line with diamonds corresponds to $E(D_F,\hat{\theta}(p,\mathcal{I}-\mathcal{K}),p)$. The line with squares corresponds to $E(D_F,\hat{\theta}(p,\mathcal{I}-\mathcal{K}),\bar{p})$. The line with crosses corresponds to $E(D_F,\hat{\theta}(\bar{p},\mathcal{I}-\mathcal{K}),\bar{p})$.

5 FT and generalization

Improving generalization ability of a neural network has been researching for more than a decade. A common technique is to add a regularizer term, such as weight decay [11] and local regularizer [4, 19], to mean square errors term. We have found that the objective function for node fault tolerant is of the same form – the additional term $p\theta^T(G-H_\phi)\theta$ is a regularizer term. So, it is interesting to ask whether there is any regularizer for optimizing generalization can also be used to optimize node fault tolerant.

Regularizer for node fault tolerant By inspecting the regularizer in the robust learning (9) $\theta^T(\hat{G}-\hat{H}_\phi)\theta$, one will see that it is neither non-positive nor nonnegative. $(\hat{G}-\hat{H}_\phi)$ is of the form :

$$\frac{1}{N}\sum_{k=1}^{N}\left[\begin{array}{ccc} 0 & \cdots & -\phi_1(x_k)\phi_M(x_k) \\ -\phi_2(x_k)\phi_1(x_k) & \cdots & -\phi_2(x_k)\phi_M(x_k) \\ \cdots & \cdots & \cdots \\ -\phi_M(x_k)\phi_1(x_k) & \cdots & 0 \end{array}\right].$$

Since $\phi_i(x_k)\phi_j(x_k) \geq 0$ for all $i,j = 1,2,\cdots,M$ and $k = 1,2,\cdots,N$, the off-diagonal elements are all nonpositive. This matrix consists of both positive and negative eigenvalues as stated in the following lemma.

Lemma 3 *The matrix $\hat{G}-\hat{H}_\phi$ consists of both positive and negative eigenvalues.*

(Proof) Since the diagonal elements of symmetric $\hat{G} - \hat{H}_\phi$ are all zeros, the summation of all its eigenvalues must be zero and the eigenvalues must be real. Then it is readily shown by contradiction that $\hat{G} - \hat{H}_\phi$ must consist of both positive and negative eigenvalues or else all the eigenvalues are zeros. **Q.E.D**

The positive eigenvalues control the weight magnitudes to smaller values while the negative eigenvalues seem to control the weight magnitudes to larger values. If we consider weight magnitude as a measure to network complexity (like effective number of parameters), the negative eigenvalues of $(\hat{G} - \hat{H}_\phi)$ correspond to increase the network complexity while the positive eigenvalues correspond to decrease the network complexity.

Tipping regularizer Tipping local regularizer is basically a generalized form weight decay defined as follows :

$$\theta^T \begin{bmatrix} \lambda_i & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \lambda_M \end{bmatrix} \theta \qquad (15)$$

Each weight is controlled by an independent hyperparameter that can be obtained by the idea of evidence maximization [10]. It can readily shown that these hyperparameters are all nonnegative.

Lemma 4 *Matrix* $\mathbf{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_M\}$ *consists of only nonnegative eigenvalues.*

(Proof) In accordance with the re-estimation technique shown in p.654 of [19], the parameter λ_i satisfies the condition $\lambda_i = \langle \theta_i^2 \rangle^{-1}$, where the expectation is taken over the posterior probability of θ given training data set and the regularizer. Obviously, $\lambda_i \geq 0$ and the proof is completed. **Q.E.D.**

Therefore similar to that weight decay, Tipping's local regularizer is a nonnegative regularizer. It is able to penalize the weight magnitude but it is unlikely to train an RBF network to be optimally node fault tolerant.

Chen regularizer Inspired by Tipping's local regularization, Chen in [4] extended the idea of orthogonal least square (OLS) method by introducing a regularizer term with M hyperparameters. Consider the training data set consisting of N data, $\{x_k, y_k\}_{k=1}^N$, the objective function to be minimized in Chen's paper is defined as follows :

$$\sum_{k=}^{N} (y_k - \phi(x_k)^T \theta)^2 + \theta^T A^T Z A \theta, \qquad (16)$$

where A is an upper triangular matrix satisfies the following condition :

$$\begin{bmatrix} \phi(x_1)^T \\ \phi(x_2)^T \\ \cdots \\ \phi(x_N)^T \end{bmatrix} = \underbrace{\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1M} \\ w_{21} & w_{22} & \cdots & w_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NM} \end{bmatrix}}_{W} \times \underbrace{\begin{bmatrix} 1 & a_{12} & \cdots & a_{1M} \\ 0 & 1 & \cdots & a_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{A} \qquad (17)$$

with vectors $w_i = (w_{1i}, \cdots, w_{Ni})^T \in R^N$ for all $i = 1, 2, \cdots, M$ are orthogonal basis, i.e. $w_i^T w_j = 0$ if $i \neq j$. $W \in R^{NM}$ and $A \in R^{MM}$ are obtained by a modified Gram-Schmitt procedure called OLS method. Z is a diagonal matrix, $\mathbf{diag}\{z_1, z_2, \cdots, z_M\}$. of elements $z_1, z_2, \cdots, z_M$ the hyperparameters satisfying the following equalities. For all $i = 1, \cdots, M$,

$$z_i = \frac{\gamma_i \sum_{k=}^{N} (y_k - \phi(x_k)^T \theta)^2}{\left(N - \sum_{j=1}^{M} \gamma_j\right) (A\theta)_i^2} \qquad (18)$$

$$\gamma_i = \frac{w_i^T w_i}{z_i + w_i^T w_i}, \qquad (19)$$

where $(A\theta)_i$ is the i^{th} element of the vector $A\theta$. It can readily be shown that the eigenvalues of the matrix $A^T Z A$ are all nonnegative, as stated in the following lemma.

Lemma 5 *Matrix* $A^T Z A$ *defined in Chen's LROLS consists of only nonnegative eigenvalues.*

(Proof) From (18), we can see that $z_i \geq 0$ for all $i = 1, \cdots, M$. Hence, $\theta^T A^T Z A \theta \geq 0$. Therefore, the matrix $A^T Z A$ is positive semidefinite. **Q.E.D.**

Similar to Tipping's local regularizer, it is able to penalize the weight magnitude but it is unlikely to train an RBF network to be optimally node fault tolerant.

By comparing the eigenvalues of the respective regularizers, the purpose of Tipping's regularizer and Chen's local regularizer is only to reduce the complexity of a network. While the implicit regularizer is somehow not only to reduce the complexity, but also to increase the network complexity (similar to adding network redundancy). Therefore, we anticipate that the regularizer solely for improving generalization is unlikely be applicable for attaining an RBF with optimal node fault tolerance.

6 Conclusion

Assuming all the nodes have equal fault rate and their faults are independently random, we have derived from Kullback-Leibler divergence an objective function for robust training an RBF network that can optimally tolerate multiple nodes fault. Simulation results have demonstrated that the performance of the fault tolerant RBF is only degraded gracefully compared with the one trained by pseudo inverse. Besides, we have also shown that the performance of such a fault tolerant RBF is insensitive to the uncertainty of the fault rate p. Observing that the objective function is in a form of a MSE plus a regularizer term, we have also compared the property of this implicit regularizer with other regularizer for achieving good generalization. It is found that such regularizers are not suitable for optimal fault tolerant training. The last finding suggests Finally, we would like to point out that the results presented in this paper provide only a partial picture to the relation between fault tolerance and generalization. A lot more work will be needed for a complete picture for this problem. The objective function derived and the property of the implicit regularizer might serve as a vehicle for further investigation.

References

[1] Bolt G., *Fault tolerant in multi-layer Perceptrons.* PhD Thesis, University of York, UK, 1992.

[2] Cavalieri S. and O. Mirabella, A novel learning algorithm which improves the partial fault tolerance of multilayer neural networks, *Neural Networks*, Vol.12, 91-106, 1999.

[3] Chandra P. and Y. Singh, Fault tolerance of feedforward artificial neural networks – A framework of study, *Proceedings of IJCNN'03* Vol.1 489-494, 2003.

[4] Chen S., Local regularization assisted orthogonal least squares regression, *International Journal of Control*, in press.

[5] Chiu C.T. *et al.*, Modifying training algorithms for improved fault tolerance, ICNN'94 Vol.I, 333-338, 1994.

[6] Deodhare D., M. Vidyasagar and S. Sathiya Keerthi, Sythesis of fault-tolerant feedforward neural networks using minimax optimization, *IEEE Transactions on Neural Networks*, Vol.9(5), 891-900, 1998.

[7] Emmerson M.D. and R.I. Damper, Determining and improving the fault tolerance of multilayer perceptrons in a pattern-recognition application, *IEEE Transactions on Neural Networks*, Vol.4, 788-793, 1993.

[8] Fontenla-Romero O. *et al*, A measure of fault tolerance for functional networks, *Neurocomputing*, Vol.62, 327-347, 2004.

[9] Kullback S., *Information Theory and Statistics*, Wiley, 1959.

[10] Mackay D.J.C. (1992), A Practical Bayesian Framework for Backprop Networks, *Neural Computation*, Vol.4(3) 448-472.

[11] Moody J.E., Note on generalization, regularization, and architecture selection in nonlinear learning systems, *First IEEE-SP Workshop on Neural Networks for Signal Processing*, 1991.

[12] Murray A.F. and P.J. Edwards, Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training, *IEEE Transactions on Neural Networks*, Vol.5(5), 792-802, 1994.

[13] Neti C. M.H. Schneider and E.D. Young, Maximally fault tolerance neural networks, *IEEE Transactions on Neural Networks*, Vol.3(1), 14-23, 1992.

[14] Phatak D.S. and I. Koren, Complete and partial fault tolerance of feedforward neural nets., *IEEE Transactions on Neural Networks*, Vol.6, 446-456, 1995.

[15] Phatak D.S., Relationship between fault tolerance, generalization and the Vapnik-Cervonenkis (VC) dimension of feedforward ANNs, *IJCNN'99*, Vol.1, 705-709, 1999.

[16] Phatak D.S. and E. Tcherner, Synthesis of fault tolerance neural networks, *Proc. IJCNN'02*, 1475-1480, 2002.

[17] Sequin C.H. and R.D. Clay, Fault tolerance in feedforward artificial neural networks, *Neural Networks*, Vol.4, 111-141, 1991.

[18] Simon D. and H. El-Sherief, Fault-tolerance training for optimal interpolative nets, *IEEE Transactions on Neural Networks*, Vol.6, 1531-1535, 1995.

[19] Tipping, M.E., The relevance vector machine, *Advances in Neural Information Processing Systems 12*, p.652-658, MIT Press, 2000.