On Node-Fault-Injection Training of an RBF Network

John Sum[1], Chi-sing Leung[2], and Kevin Ho[3]*

[1] Institute of E-Commerce, National Chung Hsing University
Taichung 402, Taiwan pfsum@nchu.edu.tw
[2] Department of Electronic Engineering, City University of Hong Kong
Kowloon Tong, KLN, Hong Kong eeleungc@cityu.edu.hk
[3] Department of Computer Science and Communication Engineering,
Providence University, Sha-Lu, Taiwan. ho@pu.edu.tw

Abstract. While injecting fault during training has long been demonstrated as an effective method to improve fault tolerance of a neural network, not much theoretical work has been done to explain these results. In this paper, two different node-fault-injection-based on-line learning algorithms, including (1) injecting multinode fault during training and (2) weight decay with injecting multinode fault, are studied. Their almost sure convergence will be proved and thus their corresponding objective functions are deduced.

1 Introduction

Many methods have been developed throughout the last two decades to improve the fault tolerance of a neural network. Well known methods include injecting random fault during training [20, 4], introducing network redundancy [18], applying weight decay learning [7], formulating the training algorithm as a nonlinear constraint optimization problem [8, 17], bounding weight magnitude during training [5, 12, 14], and adding fault tolerant regularizer [2, 16, 21]. A complete survey on fault tolerant learning methods is exhaustive. Readers please refer to [6] and [23] for reference.

Amongst all, the fault-injection-based on-line learning algorithms are of least theoretical studied. By fault injection, either fault or noise is introduced to a neural network model before each step of training. This fault could either be node fault (stuck-at-zero), weight noise or input noise. As many studies have been reported in the literature on input noise injection [1, 3, 19, 10, 11], the primary focus of this paper is on node fault injection. Our companion paper [13] will be focus on weight noise injection.

Suppose a neural network consists of M weights. Let $\theta \in R^M$ be the weight vector of a neural network model and the update equation is given by $\theta(t+1) = \theta(t) - F(x(t+1), y(t+1), \theta(t))$. The idea of node fault injection is to mimic the network that is suffered from random node fault. Before each step of training,

* Corresponding author.

each node output is set randomly to either normal or zero (stuck-at-zero). Weight update is then based on this perturbed nodes' output. For simplicity, we let $\tilde{F}(\cdot,\cdot,\cdot)$ be the function corresponding to this perturbed network model. The update equation can readily be defined as follows :

$$\theta(t+1) = \theta(t) - \tilde{F}(x(t+1), y(t+1), \theta(t)). \tag{1}$$

Despite the technique of injecting node fault has appeared for almost two decades [4, 7, 20], little theoretical analytical result is known about its convergence behavior, the corresponding objective function to be minimized and its extension by adding weight decay during training an RBF network.

In this paper, two node-fault-injection-based on-line learning algorithms, namely (1) injecting multinode fault [4, 20] during training and (2) weight decay with injecting multinode fault [7], will be analyzed. Analysis on weight-noise-injection-based training will be presented in another paper. Their corresponding objective functions and their convergence properties will be analyzed analytically. The major technique is by applying the Gladyshev Theorem in the theory of Stochastic Approximation [9]. The definition of a RBF model and the node fault injection training algorithms will be introduced in the next section. Then, the main results on their convergence properties and the objective functions will be stated in Section 3. The proof of theorems will be presented in Section 4. Section 5 will give a conclusion.

2 RBF training with node fault injection

Let $\mathcal{M}_0$ be an unknown system to be modeled. The input and output of $\mathcal{M}_0$ are denoted by x and y respectively. The only information we know about $\mathcal{M}_0$ is a set of measurement data $\mathcal{D}$, where $\mathcal{D} = \{(x_k, y_k)\}_{k=1}^N$. Making use of this data set, an estimated model $\hat{\mathcal{M}}$ that is *good* enough to capture the *general behavior* of the unknown system can be obtained. For $k = 1, 2, \cdots, N$

$$\mathcal{M}_0 \ : \quad y_k = f(x_k) + e_k, \tag{2}$$

where (x_k, y_k) is the k^{th} input-output pair that is measured from an unknown deterministic system $f(x)$ with random output noise e_k, $e_k \sim \mathcal{N}(0, S_e)$. To model the unknown system, we assume that $f(x)$ can be realized by an RBF network consisting of M hidden nodes, i.e.

$$y_k = \sum_{i=1}^{M} \theta_i^* \phi_i(x_k) + e_k \tag{3}$$

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

$$\phi_i(x) = \exp\left(-\frac{(x - c_i)^2}{\sigma}\right), \tag{4}$$

where c_is are the centers of the radial basis functions and the positive parameter $\sigma > 0$ controls the width of the radial basis functions. Thus, a model $\mathcal{M}$ in Ω is represented by an M-vector, $\theta^* = (\theta_1^*, \theta_2^*, \cdots, \theta_M^*)^T$ and the model set Ω will be isomorphic to R^M.

2.1 Multinode fault injection training

In conventional training by minimizing MSE, the update equation for $\theta(t)$ is given by

$$\theta(t+1) = \theta(t) + \mu_t(y_t - \phi^T(x_t)\theta(t))\phi(x_t), \tag{5}$$

where μ_t (for $t \geq 1$) is the step size at the t^{th} iteration. While an RBF network is trained by multinode fault injection, the update equation is given by

$$\theta(t+1) = \theta(t) + \mu_t(y_t - \tilde{\phi}^T(x_t)\theta(t))\tilde{\phi}(x_t), \tag{6}$$

$$\tilde{\phi}_i = (1-\beta_i)\phi_i, \quad P(\beta_i = 1) = p, \ \ \forall\, i = 1, \cdots, M. \tag{7}$$

We assume that all nodes are of equal fault rate p, i.e.

$$P(\beta_i) = \begin{cases} p & \text{if } \beta_i = 1 \\ 1-p & \text{if } \beta_i = 0. \end{cases} \tag{8}$$

for $i = 1, 2, \cdots, M$, Besides, $\beta_1, \cdots, \beta_M$ are independent random variables.

2.2 Weight decay-based multinode fault injection training

The update equation for weight decay-based multinode fault injection training is similar to that of simple multinode fault injection, except that a decay term is added. For a RBF network, $f(x_t, \theta(t)) = \phi(x_t)^T\theta(t)$, that is trained by injecting multinode fault during weight decay learning,

$$\theta(t+1) = \theta(t) + \mu_t \left\{ (y_t - \tilde{\phi}^T(x_t)\theta(t))\tilde{\phi}(x_t) - \lambda\theta(t) \right\}, \tag{9}$$

$$\tilde{\phi}_i = (1-\beta_i)\phi_i, \tag{10}$$

for all $i = 1, \cdots, M$. The definition of the random variable β_i is the same as before. $P(\beta_i) = p$ if $\beta_i = 1$ and $(1-p)$ otherwise.

3 Main Results

Theory of stochastic approximation has been developed for more than half a century for the analysis of recursive algorithms. Advanced theoretical works for complicated recursive algorithms have still been under investigation [15]. The theorem applied in this paper is based on Gladyshev Theorem [9].

Theorem 1 (Gladyshev Theorem [9]). *Let $\theta(t)$ and $M(\theta(t),\omega(t))$ for all $t = 0, 1, 2,$ and so on be m-vectors. $\omega(t)$ for all $t = 0, 1, 2,$ and so on are i.i.d. random vectors with probability density function $P(\omega)$ [4]. Consider a recursive algorithm defined as follows :*

$$\theta(t+1) = \theta(t) - \mu_t M(\theta(t), \omega(t)). \tag{11}$$

In which, the expectation of $M(\theta, \omega)$ over ω, i.e.

$$\bar{M}(\theta) = \int M(\theta, \omega) P(\omega) d\omega, \tag{12}$$

has unique solution θ^* such that $\bar{M}(\theta^*) = 0$.

Suppose there exists positive constants κ_1 and κ_2 such that the following conditions are satisfied :

(C1) $\mu_t \geq 0$, $\sum_t \mu_t = \infty$ *and* $\sum_t \mu_t^2 < \infty$.
(C2) $\inf_{\varepsilon < \|\theta - \theta^*\| < \varepsilon^{-1}} (\theta - \theta^*)^T \bar{M}(\theta) > 0$, *for all* $\varepsilon > 0$.
(C3) $\int \|M(\theta, \omega)\|^2 P(\omega) d\omega \leq \kappa_1 + \kappa_2 \|\theta\|^2$.

Then for $t \to \infty$, $\theta(t)$ converges to θ^* with probability one.

Normally, the first condition can easily be satisfied. It is because the step size μ_t could be defined as $\frac{\text{const}}{t}$ for all $t \geq 1$. Therefore, we skip the proof of Condition (C1) in the rest of this section. For the sake of presentation, we let $Y = \frac{1}{N} \sum_{k=1}^{N} y_k \phi(x_k)$. Besides, we have $\bar{M}(\theta) = -h(\theta)$ and ω is a random vector augmenting (x_t, y_t, β).

3.1 Multinode fault injection training

Applying Galdyshev Theorem, the following theorem can be proved for injecting multinode fault training.

Theorem 2. *For injecting multinode fault during training an RBF network, the weight vector $\theta(t)$ will converge with probability one to*

$$\theta^* = [H_\phi + p(Q_g - H_\phi)]^{-1} Y. \tag{13}$$

Besides, the corresponding objective function to be minimized is given by

$$\mathcal{L}(\theta|\mathcal{D}) = \frac{1}{N} \sum_{k=1}^{N} (y_k - f(x_k, \theta))^2 + p\theta^T (Q_g - H_\phi)\theta. \tag{14}$$

[4] In the following convergence proof, $\omega(t) = (x_t, y_t, \beta_t)$. Owing not to confuse the time index t with the element index k, the subscript t is omitted. So that $\omega(t) = (x_t, y_t, \beta)$.

3.2 Weight decay-based multinode fault injection training

For weight decay-based multinode fault injection training, we can have the following theorem.

Theorem 3. *For injecting multinode fault during weight decay training an RBF network, the weight vector $\theta(t)$ will converge with probability one to*

$$\theta^* = \left[H_\phi + p(Q_g - H_\phi) + \frac{\lambda}{1-p} I_{M\times M} \right]^{-1} Y. \tag{15}$$

Besides, the corresponding objective function to be minimized is given by

$$\mathcal{L}(\theta|\mathcal{D}) = \frac{1}{N}\sum_{k=1}^{N}(y_k - f(x_k,\theta))^2 + \theta^T \left\{ p(Q_g - H_\phi) + \frac{\lambda}{1-p} I_{M\times M} \right\} \theta. \tag{16}$$

4 Proof of Theorems

Next, we are going to apply the Gladyshev Theorem for the convergence proof. Normally, the first condition can easily be satisfied. It is because the step size μ_t could be pre-defined. So, we skip the proof of Condition (C1) for simplicity and then prove only the Condition (C2) and (C3).

4.1 Injecting multinode fault (Theorem 2)

To prove the condition (C2), we need to consider the mean update equation $h(\theta(t))$. By taking the expectation of the second part of the Equation (6) with respect to β_i, x_t and y_t, $h(\theta(t))$ will be given by

$$h(\theta(t)) = \left\{ \frac{1}{N}\sum_{k=1}^{N}(y_k - \phi^T(x_k)\theta(t))\phi(x_k) - p(H_\phi - Q_g)\theta(t) \right\}. \tag{17}$$

In which, the solution θ^* is given by

$$\theta^* = \left[H_\phi + p(Q_g - H_\phi) \right]^{-1} Y. \tag{18}$$

Hence, for all $\|\theta - \theta^*\| > 0$, we have

$$-(\theta - \theta^*)^T h(\theta) = -(\theta - \theta^*)^T \left(Y - \left[H_\phi + p(Q_g - H_\phi) \right] \theta \right),$$

which is greater than zero. Therefore, Condition $(C2)$ is satisfied.

For Condition $(C3)$, we consider the Equation (6). By triangle inequality, it is clearly that

$$\|M(\theta,\omega)\|^2 \leq \|y_t^2 \tilde{\phi}^T(x_t)\tilde{\phi}(x_t)\| + \theta^T(\tilde{\phi}(x_t)\tilde{\phi}^T(x_t))^2\theta. \tag{19}$$

Since,

$$\int (\tilde{\phi}^T\tilde{\phi})\tilde{\phi}\tilde{\phi}^T P(\beta)d\beta \leq \phi^T\phi\lambda_{\max}\left\{\int \tilde{\phi}\tilde{\phi}^T P(\beta)d\beta\right\} \leq (\phi^T\phi)^2. \tag{20}$$

Putting Equation (20) into Equation (19), it is clear that

$$\int \left\{\|M(\theta,\omega)\|^2\right\} P(\beta)d\beta \leq \|y_t^2\phi^T(x_t)\phi(x_t)\| + (\phi(x_t)^T\phi(x_t))^2\|\theta\|^2. \tag{21}$$

Further taking the expectation of the above inequality with respect to x_t and y_t, one can readily show that Condition $(C3)$ can be satisfied and the proof is completed.

With reference to Equation (17), the constant factor $(1-p)$ can be put together with μ_t and treated as a new step size. Hence, the objective function of the above algorithm is given by

$$\mathcal{L}(\theta|\mathcal{D}) = \frac{1}{N}\sum_{k=1}^{N}(y_k - f(x_k,\theta))^2 + p\theta^T(Q_g - H_\phi)\theta. \tag{22}$$

The proof for Theorem 2 is completed. **Q.E.D.**

It is worthwhile noted that Equation (14) is also identical to the objective function derived for batch model in [16].

4.2 WD-based multinode fault injection training (Theorem 3)

The corresponding $h(\theta(t))$ will be given by

$$h(\theta(t)) = (1-p)\left\{\frac{1}{N}\sum_{k=1}^{N}(y_k - \phi^T(x_k)\theta(t))\phi(x_k) - p(H_\phi - Q_g)\theta(t)\right\} - \lambda\theta(t). \tag{23}$$

In which, the solution θ^* is given by

$$\theta^* = \left[H_\phi + p(Q_g - H_\phi) + \frac{\lambda}{1-p}I_{M\times M}\right]^{-1} Y. \tag{24}$$

Hence, for all $\|\theta - \theta^*\| > 0$, we have

$$-(\theta-\theta^*)^T h(\theta) = -(\theta-\theta^*)^T\left(Y - \left[H_\phi + p(Q_g - H_\phi) + \frac{\lambda}{1-p}I_{M\times M}\right]\theta\right),$$

which is greater than zero. Therefore, Condition $(C2)$ is satisfied.

For Condition $(C3)$, we consider the Equation (9) and the similar technique as for the case of injecting multinode fault, one can readily show that

$$\int \left\{\|M(\theta,\omega)\|^2\right\} P(\beta)d\beta \leq \|y_t^2\phi^T(x_t)\phi(x_t)\| + \left\{(\phi(x_t)^T\phi(x_t))^2 + \lambda^2\right\}\|\theta\|^2. \tag{25}$$

Further taking the expectation of the above inequality with respect to x_t and y_t, one can readily show that Condition $(C3)$ can be satisfied and the proof is completed.

With reference to Equation (23), the objective function is given by

$$\mathcal{L}(\theta|\mathcal{D}) = \frac{1}{N}\sum_{k=1}^{N}(y_k - f(x_k,\theta))^2 + \theta^T\left\{p(Q_g - H_\phi) + \frac{\lambda}{1-p}I_{M\times M}\right\}\theta. \quad (26)$$

The proof for Theorem 3 is completed. **Q.E.D.**

One should notice that the weight decay effect is scaled up when random node fault is injected.

5 Conclusions

In this paper, proofs on the convergences of two node-fault-injection-based online training RBF methods have been shown and their corresponding objective functions have been deduced. For the injecting multinode-fault training, it is also found that the objective function is identical to the one that is proposed in [16] for batch-mode training an RBF to deal with multinode fault.

For the weight decay-based multinode fault injection training, two additional regularization terms are obtained in the objective function. The first one is identical to the extra term obtained for pure multinode fault injection training. The other is a weight decay term with a constant factor $\lambda/(1-p)$ is depended on the fault rate p. The constant factor can amplify the penalty of weight magnitude if the fault rate is large.

It is worthwhile noted that for λ not equal to zero, regularization effect will still exist in null fault rate situation. Generalization can be improved. So, it is suspected that weight decay-based multimode fault injection training might lead to network model with good generalization and multinode fault tolerance ability. Further investigation along the line should be valuable for future research.

Acknowledgement

The research work reported in this paper is supported in part by Taiwan NSC Research Grant 97-2221-E-005-050-.

References

1. An G. The effects of adding noise during backpropagation training on a generalization performance, *Neural Computation*, Vol.8, 643-674, 1996.
2. Bernier J.L. *et al*, Obtaining fault tolerance multilayer perceptrons using an explicit regularization, *Neural Processing Letters*, Vol.12, 107-113, 2000.
3. Bishop C.M., Training with noise is equivalent to Tikhonov regularization, *Neural Computation*, Vol.7, 108-116, 1995.

4. Bolt G., *Fault tolerant in multi-layer Perceptrons*. PhD Thesis, University of York, UK, 1992.
5. Cavalieri S. and O. Mirabella, A novel learning algorithm which improves the partial fault tolerance of multilayer NNs, *Neural Networks*, Vol.12, 91-106, 1999.
6. 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.
7. Chiu C.T. *et al.*, Modifying training algorithms for improved fault tolerance, ICNN'94 Vol.I, 333-338, 1994.
8. Deodhare D., M. Vidyasagar and S. Sathiya Keerthi, Synthesis of fault-tolerant feedforward neural networks using minimax optimization, *IEEE Transactions on Neural Networks*, Vol.9(5), 891-900, 1998.
9. Gladyshev E., On stochastic approximation, *Theory of Probability and its Applications*, Vol.10, 275-278, 1965.
10. Grandvalet Y., S. Canu, A comment on noise injection into inputs in back-propagation learning, *IEEE Transactions on Systems, Man, and Cybernetics*, 25(4), p.678-681, 1995.
11. Grandvalet Y., S. Canu, S. Boucheron, Noise injection : Theoretical prospects, *Neural Computation*, Vol.9(5), p.1093-1108, 1997.
12. Hammadi N.C. and I. Hideo, A learning algorithm for fault tolerant feedforward neural networks, *IEICE Transactions on Information & Systems*, Vol. E80-D, No.1, 1997.
13. Ho K., C.S. Leung, J. Sum, On weight-noise-injection training, in *Proc. ICONIP'2008*, Springer LNCS, 2009.
14. Kamiura N., *et al*, On a weight limit approach for enhancing fault tolerance of feedforward neural networks, *IEICE Transactions on Information & Systems*, Vol. E83-D, No.11, 2000.
15. Lai T.L., Stochastic approximation, *Annals of Statistics*, Vol. 31, No. 2, 391-406, 2003.
16. Leung C.S., J. Sum, A fault tolerant regularizer for RBF networks, *IEEE Transactions on Neural Networks*, Vol. 19 (3), pp.493-507, 2008.
17. 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.
18. 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.
19. Reed R., R.J. Marks II & S. Oh, Similarities of error regularization, sigmoid gain scaling, target smoothing, and training with jitter, *IEEE Transactions on Neural Networks*, Vol.6(3), 529-538, 1995.
20. Sequin C.H. and R.D. Clay, Fault tolerance in feedforward artificial neural networks, *Neural Networks*, Vol.4, 111-141, 1991.
21. Sum J., C.S. Leung and K. Ho, On objective function, regularizer and prediction error of a learning algorithm for dealing with multiplicative weight noise, accepted for publication in *IEEE Transactions on Neural Networks*.
22. Takase H., H. Kita and T. Hayashi, A study on the simple penalty term to the error function from the viewpoint of fault tolearnt training, *Proc. IJCNN 2004*, 1045-1050, 2004.
23. Tchernev E.B., R.G. Mulvaney, and D.S. Phatak, Investigating the Fault Tolerance of Neural Networks, *Neural Computation*, Vol.17, 1646-1664, 2005.