Noise Resistant Training for Extreme Learning Machine

Yik Lam Lui, Hiu Tung Wong, Chi-Sing Leung(✉), and Sam Kwong

City University of Hong Kong, Kowloon Tong, Hong Kong
eeleungc@cityu.edu.hk

Abstract. The extreme learning machine (ELM) concept provides some effective training algorithms to construct single hidden layer feedforward networks (SHLFNs). However, the conventional ELM algorithms were designed for the noiseless situation only, in which the outputs of the hidden nodes are not contaminated by noise. This paper presents two noise-resistant training algorithms, namely noise-resistant incremental ELM (NRI-ELM) and noise-resistant convex incremental ELM (NRCI-ELM). For NRI-ELM, its noise-resistant ability is better than that of the conventional incremented ELM algorithms. To further enhance the noise resistant ability, the NRCI-ELM algorithm is proposed. The convergent properties of the two proposed noise resistant algorithms are also presented.

Keywords: Node noise · Extreme learning machines · Incremental algorithm

1 Introduction

Single hidden layer feedforward networks (SHLFNs) can act as universal approximators [1]. With the traditional training algorithms, such as backpropagation based algorithms, we need to estimate all the connection weights, including the input weights from the input layer to the hidden layer, and the output weights from the hidden layer to the output node. Training all the connection weights may have some problems, such as local minimum. Huang et al. [2] proposed the extreme learning machine concept, where the hidden nodes are generated randomly. Besides, they showed that SHLFNs with the ELM concept can act as universal approximators too. In [2,3], Huang et al. developed the incremental ELM (I-ELM) [2] algorithm and the convex incremental ELM (CI-ELM) algorithm [3]. The mean square error (MSE) performances of these two algorithms are very well under the noiseless situation, where there is no node noise in the implementation.

In the implementation of neural networks, noise take place unavoidably [4]. When we use the finite precision technology to implement a trained network, multiplicative noise or additive noise would be introduced [5]. Also, when the implementation is at the nano-scale, transient noise may occur [6]. For traditional

F. Cong et al. (Eds.): ISNN 2017, Part II, LNCS 10262, pp. 257–265, 2017.
DOI: 10.1007/978-3-319-59081-3_31

neural network models, some batch mode learning algorithms for trained neural networks under the imperfection situation were reported [8]. To the best of our knowledge, there are not many literatures related to the noise-resistant ELMs.

This paper considers the multiplicative node noise and the additive node noise as the imperfect conditions for the SHLFN model. We first derive the training set error expression of noisy SHLFNs. Afterwards, we develop two noise-resistant incremental ELM algorithms, namely noise-resistant I-ELM (NRI-ELM) and noise-resistant CI-ELM (NRCI-ELM). For the NRI-ELM algorithm, we keep all the previously trained weights unchanged, and we adjust the output weight of the newly inserted node. The noise-resistant performance of the NRI-ELM algorithm is better than that of I-ELM and CI-ELM. For the NRCI-ELM algorithm, we use a simple rule to update all the previously trained weights, and we estimate the output weight of the new node to maximize the reduction in the training set error of noisy SHLFNs. The noise-resistant ability of the NRCI-ELM algorithm is much better than that of I-ELM, CI-ELM, and NRI-ELM. In addition, we prove that in terms of the training set error of noisy SHLFNs, the NRCI-ELM algorithm and the NRCI-ELM algorithm converges.

The rest of this paper is organized as follows. Section 2 presents the background of the ELM concept and the node noise models. Section 3 derives the two proposed noise resistant incremental ELM algorithms. Section 4 presents the simulation result. Section 5 concludes the paper.

2 ELM and Node Noise

The nonlinear regression problem is considered in this paper. The training set is denoted as $\mathbb{D}_t = \left\{(\boldsymbol{x}_k, o_k) : \boldsymbol{x}_k \in \mathbb{R}^M, o_k \in \mathbb{R}, k = 1, \ldots, N\right\}$, where $\boldsymbol{x}_k$ and o_k are the input and the target output of the k-th sample, respectively. The test set is denoted as $\mathbb{D}_f = \left\{(\boldsymbol{x}'_{k'}, o'_{k'}) : \boldsymbol{x}'_{k'} \in \mathbb{R}^M, o'_{k'} \in \Re, k' = 1, \ldots, N'\right\}$. In a SHLFN with n hidden nodes, the network output is given by $f_n(\boldsymbol{x}) = \sum_{i=1}^{n} \beta_i h_i(\boldsymbol{x})$, where $h_i(\boldsymbol{x})$ is the output of the ith hidden node, and β_i is the output weight of the ith hidden node. In this paper, we use the sigmoid function as the activation function. Hence the output of the ith hidden node is given by

$$h_i(\boldsymbol{x}) = \frac{1}{1 + \exp\{-(\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{x} + b_i)\}}, \tag{1}$$

where b_i is the input bias of the ith hidden node, and $\boldsymbol{w}_i$ is the input weight vector of the ith hidden node.

In the ELM approach [2,3], the bias terms b_i's and the input weight vectors $\boldsymbol{w}_i$'s are randomly generated. We only need to estimate the output weights β_i's. For a trained SHLFN, the training set error is given by

$$\mathcal{E} = \sum_{k=1}^{N} (y_k - \sum_{i=1}^{n} \beta_i h_i(\boldsymbol{x}_k))^2 = \left\| \boldsymbol{o} - \sum_{i=1}^{n} \beta_i \boldsymbol{h}_i \right\|_2^2, \tag{2}$$

where $\boldsymbol{o} = [o_1, \ldots, o_N]^{\mathrm{T}}$, and $\boldsymbol{h}_i = [h_i(\boldsymbol{x}_1), \ldots, h_i(\boldsymbol{x}_N)]^{\mathrm{T}}$.

In the implementation of a network, node noise may not be avoided. When we use the digital implementation, finite precision can be modelled as multiplicative node noise or additive node noise [5]. When we use the floating point approach, the round-off error can be modelled as multiplicative noise. On the other hand, when we use the fixed point approach, the round-off error can be modelled as additive noise.

Given the kth input vector, when a hidden node is affected by the multiplicative noise and additive noise concurrently, its output can be modelled as

$$\tilde{h}_i(\boldsymbol{x}_k) = (1 + \delta_{ik})h_i(\boldsymbol{x}_k) + \epsilon_{ik}, \forall i = 1, \ldots, n \text{ and } \forall k = 1, \ldots, N, \tag{3}$$

where δ_{ik}'s are the noise factors that describe the deviation due to the multiplicative node noise, and ϵ_{ik}'s are the noise factors that describe the deviation due to the additive node noise. Note that in the multiplicative noise case, the magnitude of the noise component "$\delta_{ik}h_i(\boldsymbol{x})$" is proportional to the magnitude of the output $h_i(\boldsymbol{x})$. This paper assumes that the noise factors δ_{ik}'s and ϵ_{ik}'s are zero-mean identically independently distributed random variables with variances equal to σ_δ^2 and σ_ϵ^2, respectively.

From [3], the CI-ELM algorithm works very well for the noiseless situation. For example, as shown in Fig. 1(a), the network outputs fit the training samples very well. However, when node noise exists, the network outputs contain a lot of noise with large magnitude, as shown in Fig. 1(b). When our proposed NRCI-ELM is used, the noise in the network outputs can be greatly suppressed, as shown in Fig. 1(c) and (d).

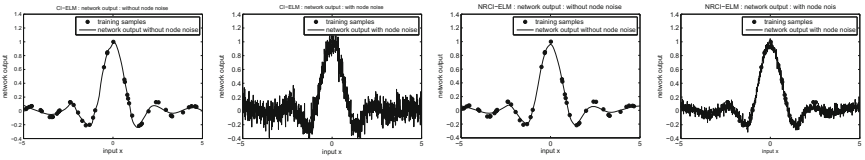

Fig. 1. Illustration of the noise resistant ability of CI-ELM and NRCI-ELM. (a) The network output of a noiseless network with CI-ELM. (b) The network output of a noisy network with CI-ELM. (c) The network output of a noiseless network with NRCI-ELM. (d) The network output of a noisy network with NRCI-ELM. In this example, $\sigma_\epsilon^2 = \sigma_\delta^2 = 0.01$.

3 Noise Resistant Incremental Learning

For a SHLFN with a particular noise pattern, the training set error can be expressed as

$$\tilde{\mathcal{E}} = \sum_{k=1}^{N} \left(o_k - \sum_{i=1}^{n} \beta_i \tilde{h}_i(\boldsymbol{x}_k) \right)^2. \tag{4}$$

According to the properties of δ_{ik}'s and ϵ_{ik}'s, the statistics of $\tilde{h}_i(\boldsymbol{x}_k)$'s are given by

$$\langle \tilde{h}_i(\boldsymbol{x}_k) \rangle = h_i(\boldsymbol{x}_k), \tag{5}$$

$$\langle \tilde{h}_i^2(\boldsymbol{x}_k) \rangle = (1+\sigma_\delta^2) h_i^2(\boldsymbol{x}_k) + \sigma_\epsilon^2, \tag{6}$$

$$\langle \tilde{h}_i(\boldsymbol{x}_k) \tilde{h}_j(\boldsymbol{x}_k) \rangle = h_i(\boldsymbol{x}_k) h_j(\boldsymbol{x}_k), \forall\, i \neq j. \tag{7}$$

Taking the expectation over all possible noise patterns, we obtain the training set error of noisy SHLFNs, given by

$$\left\langle \tilde{\mathcal{E}} \right\rangle = \left\langle \sum_{k=1}^{N} \left(o_k - \sum_{i=1}^{n} \beta_i \Big((1+\delta_{ik}) h_i(\boldsymbol{x}_k) + \epsilon_{ik} \Big) \right)^2 \right\rangle. \tag{8}$$

From (5)–(7), Eq. (8) becomes

$$\left\langle \tilde{\mathcal{E}} \right\rangle = \left\| \boldsymbol{o} - \sum_{i=1}^{n} \beta_i \boldsymbol{h}_i \right\|_2^2 + \sigma_\delta^2 \sum_{i=1}^{n} \beta_i^2 \|\boldsymbol{h}_i\|_2^2 + \sigma_\epsilon^2 N \sum_{i=1}^{n} \beta_i^2. \tag{9}$$

Similarly, we can obtain the test set error of noisy SHLFNs, given by

$$\left\langle \tilde{\mathcal{E}}_t \right\rangle = \left\| \boldsymbol{o}' - \sum_{i=1}^{n} \beta_i \boldsymbol{h}_i' \right\|_2^2 + \sigma_\delta^2 \sum_{i=1}^{n} \beta_i^2 \|\boldsymbol{h}_i'\|_2^2 + \sigma_\epsilon^2 N' \sum_{i=1}^{n} \beta_i^2, \tag{10}$$

where $\boldsymbol{o}' = [o_1', \ldots, o_N']^{\mathrm{T}}$, and $\boldsymbol{h}_i = [h_i(\boldsymbol{x}_1'), \ldots, h_i(\boldsymbol{x}_{N'}')]^{\mathrm{T}}$.

For the NRI-ELM, at the nth iteration, a new hidden node $h_n(\cdot)$, whose input bias and input weight vector are randomly generated, is inserted into the network. We keep the output weights $\{\beta_1, \ldots, \beta_{n-1}\}$ of the previously inserted hidden nodes unchanged. We need to estimate the output weight β_n of the nth hidden node. From (9), the training set error of the noisy networks at the nth iteration is

$$L_n = \left\| \boldsymbol{o} - \sum_{i=1}^{n} \beta_i \boldsymbol{h}_i \right\|_2^2 + \sigma_\delta^2 \sum_{i=1}^{n} \beta_i^2 \|\boldsymbol{h}_i\|_2^2 + \sigma_\epsilon^2 N \sum_{i=1}^{n} \beta_i^2. \tag{11}$$

Define

$$\boldsymbol{f} = \sum_{i=1}^{n} \beta_i \boldsymbol{h}_i, \quad \boldsymbol{e}_n = \boldsymbol{o} - \sum_{i=1}^{n} \beta_i \boldsymbol{h}_i, \quad v_n = \sum_{i=1}^{n} \beta_i^2 \|\boldsymbol{h}_i\|_2^2, \quad u_n = N \sum_{i=1}^{n} \beta_i^2. \tag{12}$$

From (12), Eq. (11) can be rewritten as

$$L_n = \|\boldsymbol{e}_n\|_2^2 + \sigma_\delta^2 v_n + \sigma_\epsilon^2 u_n. \tag{13}$$

From (13), the change in the training set error between the nth-iteration and $(n-1)$th-iteration is given by

$$\triangle_n = L_n - L_{n-1} = -2\beta_n \boldsymbol{e}_{n-1}^{\mathrm{T}} \boldsymbol{h}_n + (1+\sigma_\delta^2) \beta_n^2 \|\boldsymbol{h}_n\|_2^2 + \sigma_\epsilon^2 \beta_n^2 N. \tag{14}$$

Since $\triangle_n$ is a quadratic function of β_n with a minimum value equal to a negative value, the optimal value of β_n to maximize the decrease in the training set error is given by

$$\beta_n = \frac{\boldsymbol{e}_{n-1}^{\mathrm{T}} \boldsymbol{h}_n}{(1+\sigma_\delta^2)\|\boldsymbol{h}_n\|_2^2 + N\sigma_\epsilon^2}. \tag{15}$$

With (15), the change in the training set error between two consecutive iterations is

$$\triangle_n = -\frac{\left(\boldsymbol{e}_{n-1}^{\mathrm{T}} \boldsymbol{h}_n\right)^2}{(1+\sigma_\delta^2)\,\|\boldsymbol{h}_n\|_2^2 + N\sigma_\epsilon^2}. \tag{16}$$

Equation (16) means that when we inserted a new hidden node, the training set error of noisy networks decreases. That means, in terms of the training set MSE of noisy network, the NRI-ELM algorithm converges. Algorithm 1 shows the proposed NRI-ELM algorithm. From Steps (5)–(8) in Algorithm 1, for the NRI-ELM algorithm, the computational complexity is $O(N)$ for each iteration.

Algorithm 1. NRI-ELM

1: Set n equal to zero ($n = 0$), $\boldsymbol{e}_0 = \boldsymbol{y}$, and $\boldsymbol{f}_0 = \boldsymbol{0}$.
2: **while** $n \leq n_{\max}$ **do**
3: $n = n + 1$.
4: Insert a new hidden node.
5: Compute the output vector $\boldsymbol{h}_n$ of this hidden node.
6: Compute the output weight of the newly inserted node: $\beta_n = \frac{\boldsymbol{e}_{n-1}^{\mathrm{T}} \boldsymbol{h}_n}{(1+\sigma_\delta^2)\|\boldsymbol{h}_n\|_2^2 + N\sigma_\epsilon^2}$.
7: $\boldsymbol{f}_n = \boldsymbol{f}_{n-1} + \beta_n \boldsymbol{h}_n$.
8: $\boldsymbol{e}_n = \boldsymbol{y} - \boldsymbol{f}_n$.
9: **end while**

In [3], the CI-ELM algorithm was proposed. Under the noiseless situation [3], the training set error of the original CI-ELM algorithm is better than that of I-ELM algorithm. However, as shown in Sect. 4, the original CI-ELM algorithm has a very poor noise resistant ability. Hence it is interesting to develop a noise resistant version of CI-ELM, namely *NRCI-ELM*.

In the NRCI-ELM case, after we estimate the output weight β_n at the nth iteration, we update all the previously trained weights by

$$\beta_i^{new} = (1-\beta_n)\beta_i, \tag{17}$$

for $i = 1$ to $n-1$. Hence we have the recursive definitions for $\boldsymbol{f}_n$, $\boldsymbol{e}_n$, v_n and u_n

$$\boldsymbol{f}_n = (1-\beta_n)\boldsymbol{f}_{n-1} + \beta_n \boldsymbol{h}_n, \quad \boldsymbol{e}_n = \boldsymbol{y} - \boldsymbol{f}_n,$$
$$v_n = (1-\beta_n)^2 v_{n-1} + \beta_n^2 \|\boldsymbol{h}_n\|_2^2, \; u_n = (1-\beta_n)^2 u_{n-1} + \beta_n^2 N,$$

where $\boldsymbol{f}_0 = \boldsymbol{o}$, $\boldsymbol{e}_0 = \boldsymbol{y}$, $v_0 = 0$, and $u_0 = 0$. With this new updating scheme for the previously trained output weights, the change in the training set error between the nth-iteration and $(n-1)$th-iteration is given by

$$\begin{aligned}\triangle_n = L_n - L_{n-1} = &-2\beta_n\Big(\boldsymbol{e}_{n-1}^{\mathrm{T}}\boldsymbol{r}_n + \sigma_\delta^2 v_{n-1} + \sigma_\epsilon^2 u_{n-1}\Big) \\ &+\beta_n^2\Big(\|\boldsymbol{r}_n\|_2^2 + \sigma_\delta^2(v_{n-1} + \|\boldsymbol{h}_n\|_2^2) + \sigma_\epsilon^2(u_{n-1} + N)\Big), \quad (18)\end{aligned}$$

where $\boldsymbol{r}_n = \boldsymbol{h}_n - \boldsymbol{f}_{n-1}$.

Similar to the NRI-ELM case, to maximize the decrease in the training set error of noisy networks, β_n should be given by

$$\beta_n = \frac{\boldsymbol{e}_{n-1}^{\mathrm{T}}\boldsymbol{r}_n + \sigma_\delta^2 v_{n-1} + \sigma_\epsilon^2 u_{n-1}}{\|\boldsymbol{r}_n\|_2^2 + \sigma_\delta^2(v_{n-1} + \|\boldsymbol{h}_n\|_2^2) + \sigma_\epsilon^2(u_{n-1} + N)}. \quad (19)$$

With (19), the change in the training set error between two consecutive iterations is

$$\triangle_n = -\frac{\left(\boldsymbol{e}_{n-1}^{\mathrm{T}}\boldsymbol{r}_n + \sigma_\delta^2 v_{n-1} + \sigma_\epsilon^2 u_{n-1}\right)^2}{\|\boldsymbol{r}_n\|_2^2 + \sigma_\delta^2(v_{n-1} + \|\boldsymbol{h}_n\|_2^2) + \sigma_\epsilon^2(u_{n-1} + N)}. \quad (20)$$

Equation (20) means that when we insert a new hidden node, the training set error of noisy networks decreases. That means, in terms of the training set error of noisy network, the NRCI-ELM algorithm converges too. Algorithm 2 shows the proposed NRCI-ELM algorithm. At each each iteration, the complexity of the NRCI-ELM algorithm is "$O(n) + O(N)$". Compared to the NRI-ELM case whose complexity is equal to $O(N)$, the additional complexity $O(n)$ is due to the update of the previous weights.

Algorithm 2. NRCI-ELM

1: Set $n = 0$, $\boldsymbol{e}_0 = \boldsymbol{y}$, $\boldsymbol{f}_0 = \boldsymbol{0}$, $v_0 = 0$, $v_0 = 0$ and $\boldsymbol{r}_0 = \boldsymbol{0}$.
2: **while** $n \leq n_{\max}$ **do**
3: $n = n + 1$.
4: Insert a new hidden node whose b_n and $\boldsymbol{w}_n$ are randomly generated.
5: Compute the output vector $\boldsymbol{h}_n$ for this new hidden node.
6: Compute $\boldsymbol{r}_n = \boldsymbol{h}_n - \boldsymbol{f}_{n-1}$.
7: Compute the new weight: $\beta_n = \frac{\boldsymbol{e}_{n-1}^{\mathrm{T}}\boldsymbol{r}_n + \sigma_\delta^2 v_{n-1} + \sigma_\epsilon^2 u_{n-1}}{\|\boldsymbol{r}_n\|_2^2 + \sigma_\delta^2(v_{n-1} + \|\boldsymbol{h}_n\|_2^2) + \sigma_\epsilon^2(u_{n-1} + N)}$.
8: $\boldsymbol{f}_n = (1 - \beta_n)\boldsymbol{f}_{n-1} + \beta_n\boldsymbol{h}_n$.
9: $\boldsymbol{e}_n = \boldsymbol{y} - \boldsymbol{f}_n$.
10: $v_n = (1 - \beta_n)^2 v_{n-1} + \beta_n^2\|\boldsymbol{h}_n\|_2^2$.
11: $u_n = (1 - \beta_n)^2 u_{n-1} + \beta_n^2 N$.
12: $\beta_i = (1 - \beta_n)\beta_i$, for all $i = 1, \ldots, n-1$.
13: **end while**

4 Simulation

Two real life datasets from the UCI data repository are used. They are Abalone [9] and Housing Price [10]. The Abalone dataset has 4,177 samples. Each sample has eight inputs and one output. Two thousand samples are randomly taken as the training set. The other 2,177 samples are used as the test set. The Housing Price dataset has 506 samples. Each sample has 13 inputs and one output. The training set contains 250 samples, while the test set has 256 samples.

This section considers four incremented algorithms. They are the original I-ELM algorithm, the original CI-ELM algorithm, the proposed NRI-ELM algorithm, and the proposed NRCI-ELM algorithm, respectively. Figure 2 shows the MSE performance versus the number of hidden nodes, where the noise level is equal to $\sigma_\epsilon^2 = \sigma_\delta^2 = 0.09$. It can be seen that the proposed NRI-ELM algorithm is better than the two original incremental algorithms. Also, the MSE performance of the original CI-ELM algorithm is very poor. When we use more hidden nodes, the performance of the CI-ELM algorithm suddenly becomes very poor. To sum up, the proposed NRCI-ELM algorithm is much better than the original I-ELM algorithm, the original CI-ELM algorithm and the proposed NRI-ELM algorithm.

Table 1 shows the average test set MSE values of noisy networks over 100 trials for various node noise levels. In Table 1, the number of hidden nodes is equal to 500. It can be seen that the performance of the CI-ELM algorithm is very poor. The noise resistant ability of the NRI-ELM algorithm is better than that of the I-ELM algorithm. In addition, the NRCI-ELM algorithm is much better than the other three algorithms. For instance, in the Abalone dataset with noise level $\sigma_\epsilon^2 = \sigma_\delta^2 = 0.01$, the test set MSE of I-ELM is equal to 0.01421. When the NRI-ELM is used, the test set error is reduced to 0.01367. The NRCI-ELM algorithm can further reduce the test set error to is 0.00815.

For high noise levels, the improvement of the NRCI-ELM algorithm is more significant. For instance, with the node noise level equal to $\sigma_\epsilon^2 = \sigma_\delta^2 = 0.09$ The test set MSE of the I-ELM algorithm is equal to 0.05855. With the

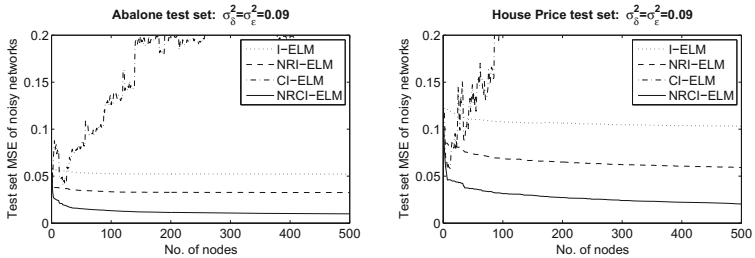

Fig. 2. The performance of the four incremental methods versus the number of additive nodes. The noise level is $\sigma_\epsilon^2 = \sigma_\delta^2 = 0.09$. The Abalone dataset is considered.

Table 1. Average test set MSEs of noisy networks. The average values are taken over 100 trials. There are 500 hidden nodes.

	Node noise level $\sigma_\epsilon^2, \sigma_\delta^2$	I-ELM mean(std)	NRI-ELM mean(std)	CI-ELM mean(std)	NRCI-ELM mean(std)
Abalone	**0.01, 0.01**	**0.01421(0.00180)**	**0.01367(0.00142)**	**0.06153(0.03175)**	**0.00815(0.00007)**
	0.09, 0.09	**0.05855(0.01561)**	**0.03386(0.00312)**	**0.49954(0.28633)**	**0.01002(0.00009)**
	0.25, 0.25	**0.14723(0.04334)**	**0.04648(0.00202)**	**1.37555(0.79548)**	**0.01174(0.00015)**
Housing	0.01, 0.01	0.02558(0.00649)	0.02425(0.00440)	0.05488(0.02510)	0.01478(0.00029)
	0.09, 0.09	0.11266(0.05269)	0.05921(0.00706)	0.39941(0.22699)	0.02026(0.00044)
	0.25. 0.25	0.28682(0.14524)	0.08081(0.00450)	1.08848(0.63079)	0.02528(0.00050)

NRI-ELM algorithm, the test set MSE is reduced to 0.03386. When the NRCI-ELM algorithm is used, the test MSE is reduced to 0.01002.

Another interesting property of the NRCI-ELM algorithm is that the test set error is insensitive to the node noise level. In the Abalone dataset, when the noise level is $\sigma_\delta^2 = \sigma_\epsilon^2 = 0.01$, the test set error of the NRCI-ELM algorithm is equal to 0.00815. When the noise level is greatly increased to $\sigma_\delta^2 = \sigma_\epsilon^2 = 0.25$, the test set error of the NRCI-ELM algorithm is slightly increased to 0.01174 only.

One may suggest that we should use the NRCI-ELM algorithm only because its test set error of noisy network is the best. The difference between the NRCI-ELM algorithm and the NRI-ELM algorithm is the computation complexity. For the NRI-ELM algorithm, the complexity is $O(N)$. But for the NRCI-ELM algorithm, the computation complexity is "$O(N) + O(n)$".

5 Conclusion

This paper proposed two incremental ELM algorithms, namely NRI-ELM and NRCI-ELM, for handling node noise. They insert the randomly generated hidden nodes into the network in the one-by-one manner. The NRI-ELM algorithm adjusts the output weight of the newly inserted hidden node only. Its noise-resistant ability is better than that of the original I-ELM algorithm and the original CI-ELM algorithm. Besides, we proposed the NRCI-ELM algorithm. It estimates the output weight of the newly additive node, and uses a single rule to modify the previously trained output weights. In addition, we prove that for the two proposed algorithms, the training set MSE of noisy networks converges. Simulation examples illustrate that the noise resistant ability of the NRI-ELM algorithm and NRCI-ELM algorithm is better than that of I-ELM and CI-ELM. In addition, the NRCI-ELM algorithm has the best noise resistant ability, compared to other three incremental algorithms. For the NRI-ELM algorithm, the complexity is $O(N)$. For the NRCI-ELM algorithm, the computation complexity is "$O(N) + O(n)$".

Acknowledgment. The work was supported by a research grant from the Government of the Hong Kong Special Administrative Region (CityU 11259516).

References

1. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
2. Huang, G.B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural Netw. **17**(4), 879–892 (2006)
3. Huang, G.B., Chen, L.: Convex incremental extreme learning machine. Neurocomputing **70**, 3056–3062 (2007)
4. Burr, J.: Digital neural network implementations. In: Neural Networks, Concepts, Applications, and Implementations. Prentice Hall, Englewood Cliffs, NJ (1995)
5. Liu, B., Kaneko, K.: Error analysis of digital filter realized with floating-point arithmetic. Proc. IEEE **57**(10), 1735–1747 (1969)
6. Mahvash, M., Parker, A.C.: Synaptic variability in a cortical neuromorphic circuit. IEEE Trans. Neural Netw. Learn. Syst. **24**(3), 397–409 (2013)
7. Leung, C.-S., Wang, H., Sum, J.: On the selection of weight decay parameter for faulty networks. IEEE Trans. Neural Netw. **21**(8), 1232–1244 (2010)
8. Leung, C.-S., Sum, J.: RBF networks under the concurrent fault situation. IEEE Trans. Neural Netw. Learn. Syst. **23**(7), 1148–1155 (2012)
9. Sugiyama, M., Ogawa, H.: Optimal design of regularization term and regularization parameter by subspace information criterion. Neural Netw. **15**(3), 349–361 (2002)
10. Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml