Neural Comput & Applic (2017) 28 (Suppl 1):S365–S378
DOI 10.1007/s00521-016-2350-4

REVIEW

Reducing the complexity of an adaptive radial basis function network with a histogram algorithm

Pey Yun Goh[1] · Shing Chiang Tan[1] · Wooi Ping Cheah[1] · Chee Peng Lim[2]

Received: 7 April 2015 / Accepted: 10 May 2016 / Published online: 26 May 2016

Abstract In this paper, a constructive training technique known as the dynamic decay adjustment (DDA) algorithm is combined with an information density estimation method to develop a new variant of the radial basis function (RBF) network. The RBF network trained with the DDA algorithm (i.e. RBFNDDA) is able to learn information incrementally by creating new hidden units whenever it is necessary. However, RBFNDDA exhibits a greedy insertion behaviour that absorbs both useful and non-useful information during its learning process, therefore increasing its network complexity unnecessarily. As such, we propose to integrate RBFNDDA with a histogram (HIST) algorithm to reduce the network complexity. The HIST algorithm is used to compute distribution of information in the trained RBFNDDA network. Then, hidden nodes with non-useful information are identified and pruned. The effectiveness of the proposed model, namely RBFNDDA-HIST, is evaluated using a number of benchmark data sets. A performance comparison study between RBFNDDA-HIST and other classification methods is conducted. The proposed RBFNDDA-HIST model is also applied to a real-world condition monitoring problem in a power generation plant. The results are analysed and discussed. The outcome indicates that RBFNDDA-HIST not only can reduce the number of hidden nodes significantly without requiring a long training time but also can produce promising accuracy rates.

✉ Pey Yun Goh
pygoh@mmu.edu.my

[1] Faculty of Information Science and Technology, Multimedia University, Melaka 75450, Malaysia

[2] Centre for Intelligent Systems Research, Deakin University, Waurn Ponds, VIC 3216, Australia

Keywords Radial basis function network · Dynamic decay adjustment · Pruning · Histogram · Nodes reduction

1 Introduction

An artificial neural network (ANN) is one of the important machine learning models for performing data classification [1]. It is capable of learning information/knowledge from data sets. According to Karnin [2], some researchers prefer to employ a large network to solve a problem, because a small network may not have sufficient information to make accurate predictions. In addition, a large initial network structure can learn the underlying problem with a fast convergence speed [2–4]. However, other researchers argue that a large network structure can result in a high computation cost [5]. An oversized ANN leads to poor performance when the ANN tends to remember or capture spurious information [6, 7]. Therefore, the key question is how to determine an adequate network size for a trained ANN to recognise unseen data in a satisfactory manner. One of the possible solutions is to apply pruning algorithms to produce a parsimonious ANN structure [4, 5, 8]. Indeed, the issue of identifying an optimal ANN structure still remains an open question for investigation in the machine learning community [8–10]. As such, finding an appropriate network structure for solving a specific problem is usually accomplished by trial-and-error [5], and it is a challenge to identify the best network structure of an ANN model for achieving good classification performance.

In this study, a new variant of the radial basis function (RBF) network is developed, with the aim of reducing the number of hidden nodes automatically using a statistical method. RBF is chosen due to its universal function approximation capabilities [10]. In addition, it has been successfully used to solve many real-world problems, e.g. industrial control [11], image retrieval [10], and medical disease diagnosis [12]. A typical RBF network consists of three layers, i.e. the input, hidden, and output layers, as shown in Fig. 1. In a conventional RBF network, the number of hidden nodes must be pre-determined before training. From the viewpoint of machine learning, such prerequisite is not desirable because if too few hidden nodes are used, the RBF network would have poor generalisation due to limited flexibility. On the other hand, if too many hidden nodes are used, the RBF network would suffer from the over-fitting problem [13]. Another drawback of the RBF network is its offline learning property. When a new data sample is provided, the RBF network needs to be re-trained using all new and old data samples; otherwise, previously learned information can be corrupted [14]. This is known as the stability–plasticity dilemma. The dilemma poses the question how can a machine learning model keeps on learning information (i.e. plastic) without corrupting or erasing previously learned information (i.e. stable) [14]. One way to overcome this stability–plasticity dilemma is online or incremental learning. In essence, an incremental learning model is able to learn online and on-the-fly with respect to the incoming data samples one by one, without requiring iterative learning through all data samples. Since this incremental learning capability does not require all data samples to be kept for learning purposes, it is, therefore, flexible and scalable in tackling the volume and velocity issues in today's big-data era [15]. The dynamic decay adjustment (DDA) coupled with the RBF network imparts the incremental learning property for the resulting model, i.e. RBFNDDA [16]. We focus on RBFNDDA because it can tackle the stability–plasticity dilemma and adapt its network structure (i.e. the number of hidden nodes) incrementally. In other words, new hidden units are inserted into the network structure whenever they are required during the learning process. Besides that, the learning process of RBFNDDA is also very fast, since information is absorbed incrementally.

Despite the aforementioned favourable features, a drawback of RBFNDDA is that its incremental learning feature can lead to a greedy insertion behaviour, as a result of absorbing useful information as well as redundant, outlier, and noisy information [17]. In this case, the network size can become large. In this regard, an appropriate network size can be determined by applying growing (constructive) or pruning techniques, or a combination of both growing and pruning techniques [18]. Regularisation techniques are another option to determine a parsimonious network structure [19]. Regularisation techniques use certain criteria to identify unnecessary weights or nodes during the training process of an ANN [19]. In our work, a pruning method is adopted from a statistical perspective to reduce the number of hidden neurons of RBFNDDA. Both Garcia-Pedrajas and Ortiz-Boyer [20] and Ma and Khorasani [21] commented that pruning algorithms cannot be feasibly used in real-world applications under the following conditions: (1) when it is difficult to determine the initial network size; (2) when it is difficult to measure the relevance of network hidden nodes; and (3) when it is necessary to re-initiate the training session following a pruning process (which incurs a high computational cost). By considering these comments, in this research, a histogram (HIST) is applied to prune the hidden nodes of RBFNDDA. HIST is adopted because of its simplicity and high computational efficiency [22]. In addition, a

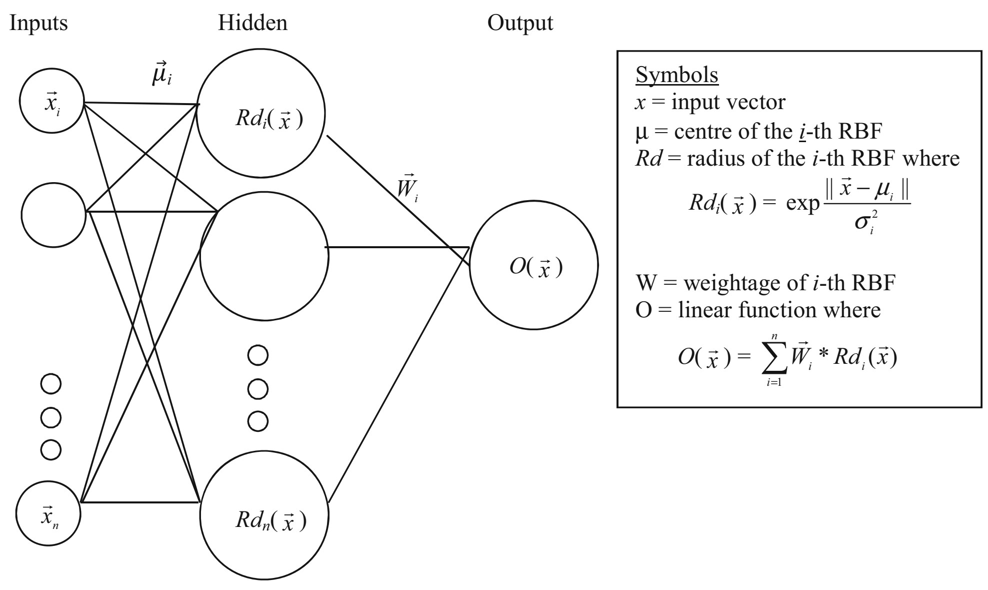
Inputs
Hidden
Output
$\vec{x}_i$
$\vec{\mu}_i$
$Rd_i(\vec{x})$
$\vec{W}_i$
$O(\vec{x})$
$\vec{x}_n$
$Rd_n(\vec{x})$
Symbols
x = input vector
μ = centre of the i-th RBF
Rd = radius of the i-th RBF where
$Rd_i(\vec{x}) = \exp\frac{\|\vec{x}-\mu_i\|}{\sigma_i^2}$
W = weightage of i-th RBF
O = linear function where
$O(\vec{x}) = \sum_{i=1}^{n} \vec{W}_i * Rd_i(\vec{x})$

Fig. 1 The architecture of an RBF network

histogram computes the distribution of information and provides the relevance measure of hidden nodes before pruning. Therefore, a hybrid model of RBFNDDA-HIST is devised, in which the use of HIST for pruning is not susceptible to the aforementioned problems.

The organisation of this paper is as follows. In Sect. 2, a literature review is provided. RBFNDDA, HIST, and the proposed RBFNDDA-HIST model are explained in Sect. 3. In Sect. 4, a study using a number of benchmark data sets from the University of California Irvine Machine Learning (UCI) Repository [23] is conducted. The experimental set-up and the characteristics of the benchmark data sets are described. The proposed RBFNDDA-HIST model is also applied to a real-world condition monitoring problem in a power generation plant. The performance of RBFNDDA-HIST is analysed, compared, and discussed. In Sect. 5, conclusions and suggestions for future work are provided.

2 Literature review

The literature review consists of two parts. The first part describes different types of pruning algorithms for ANNs. The second part presents the histogram techniques in different research areas.

2.1 Pruning methods

A pruning method is used to trim an ANN model that has a large structure. The main purpose of pruning is to remove unimportant nodes or weights [24]. Reed [3] categorised pruning methods into two groups: sensitivity and penalty (regularisation) methods. Recently, new pruning approaches to ANNs such as the genetic algorithm (GA) [23, 24], magnitude [25, 26] and cross-validation [27] methods have been introduced. Some researchers [28, 29] combined both growing and pruning methods in their studies.

A GA-based pruning method is an evolutionary approach inspired by the biological evolution principle such as mutation and reproduction for refining an ANN. Heo and Oh [25] used the GA to prune and optimise the input and hidden nodes of a multilayer perceptron (MLP) network. The correlation between the input and hidden nodes is considered during the optimisation process. Each chromosome represents the conditions from the input to hidden nodes using bit values, whereby a 0 indicates the absence/removal of an input or hidden node while a 1 indicates the presence of a node. In each generation, a new population of chromosomes is formed after going through the roulette-wheel selection, crossover, and mutation processes. Each MLP (formulated as a chromosome) is trained by the back-propagation algorithm. The chromosomes are evolved with the GA under different pruning rates. The GA was also utilised by Kaylani et al. [26] to optimise the architecture of the fuzzy adaptive resonance theory mapping (FAM) network, ellipsoidal adaptive resonance theory mapping (ARTMAP) network, and Gaussian ARTMAP network. Each of these ANNs goes through a learning process. The learned ANN models are converted into a group of chromosomes to form an initial GA population. A weighted sum approach is used to define the fitness level of the chromosomes such that fit chromosomes would be selected for reproduction. The mutation operator is employed to reduce the number of hidden nodes in the network structure.

A magnitude-based method can be used to control the magnitude of the trained weight to be small [28] or to identify the relevant weight magnitude through the use of regression [29]. As an example, Leung, Wang, and Sum [28] proposed two mean prediction error (MPE) formulae to determine the best trained fault-tolerant network with a good generalisation capability. These formulae measure weight noise and weight fault, respectively. According to [25], weight noise occurs in the weight encoding process, while weight fault occurs when certain hidden and output nodes are disconnected during the training process. The proposed method [25] applies the optimal brain damage (OBD) mechanism to rank and select the smallest structure from the pruned RBF network. A huge RBF network is trained using a weight-decay method. The optimal weight-decay parameter is used to avoid over-fitting of the trained network. The importance of the RBF nodes is ranked using the OBD concept. Then, the RBF node is deleted one by one with the MPE formula as the test error estimation procedure to produce a parsimonious trained network. A suitable model can only be selected after the construction of a few RBF networks. As such, this approach is time-consuming.

A sensitivity-based technique analyses the importance of each weight or hidden node of an ANN. The least important weight or hidden node is pruned by measuring the sensitivity rate of the error function [3]. The sensitivity-based technique was applied to prune an incremental ANN [3, 28, 29]. Huang et al. [30] proposed a sequential growing and pruning method for building an RBF network. The network grows by adding a hidden node when the information content of the node (measured by a concept known as “significance”) can contribute to the overall learning accuracy. When a training sample is provided, the significance measure of the nearest hidden node with respect to the training sample is computed. If the significance measure is more than the pre-set learning accuracy, it is retained. Otherwise, the node is pruned. The network can be slow in speed when dealing with large data sets. Another sensitivity-based technique was used by Abbas [31].

Pruning is applied when a node has the least contribution towards the performance of an MLP network trained with back-propagation. The correlation between the hidden nodes is examined, and a weight-update procedure using the anti-Hebbian rule is adopted. The rule is normally used to de-correlate the output of two neurons. After the training and weight-update process, the average correlation coefficient of each hidden node is computed, which represents the contribution of each hidden node in reducing the output error. The node that has the least contribution towards error reduction is removed. The proposed model is yet to be applied to any real-world data sets.

Leung and Tsoi [32] used the recursive least square (RLS) algorithm to train and prune a recurrent RBF network. A uniform grid of centres and spreads of the RBF nodes is established before the learning process begins. The parameter settings are estimated using the RLS algorithms. Unnecessary nodes are deleted based on the error covariance matrix. However, Hsu et al. [11] commented that the computational cost of the RLS-based RBF network could be high. Chan et al. [33] proposed a growing and pruning method based on the sensitivity of neighbours of a hidden node. This sensitivity measure is computed using square root of the distance between a hidden node and its nearest node in the data space. Three criteria are used to determine the allocation of a new hidden node with respect to a training sample: (1) the sensitivity measure of the new hidden node is smaller than the sensitivity threshold; (2) the newly added hidden node should be far enough from the nearest hidden node; and (3) the network gives a low response to the training sample. A hidden node is pruned if its sensitivity measure is larger than a sensitivity threshold. However, the problem is that the neighbourhood of a hidden node needs to be calculated again before pruning during the training process. Therefore, additional time and storage are needed [33]. Another sensitivity-based method was proposed by Medeiro and Barreto [7]. They used the correlation between the error signal of the nodes in a given layer and the error signal propagated back to the previous layer as a guide to determine irrelevant weights. Weights that are smaller error correlations than a pre-defined error tolerance are pruned from the network.

Some researchers combined a few pruning techniques together. As an example, Huynh and Setiono [27] proposed a pruning method that first used a penalty term and then a cross-validation method to improve the performance of an ANN trained by back-propagation. The proposed method has two phases. In the first phase, a penalty term is added to the error function. The magnitudes of the connection weights and the accuracy rates of training and cross-validation are examined. Connections that satisfy the pruning conditions are pruned. In the second phase, possible removal of hidden nodes is examined by considering their impacts on the classification rate. Hidden nodes that have the least impact are removed, and the network is re-trained after removal of a node. This process continues until no further nodes can be pruned. As the network needs to be re-trained, additional computational effort is required.

In summary, different types of pruning algorithms are available in the literature. Most of the methods [3, 12, 25, 28, 29, 33] have high computational costs and are slow in speed. Some algorithms [23, 24, 26, 28] use a trial-and-error approach to find suitable parameters or thresholds for different problems. In this paper, we focus on a statistical pruning method, i.e. histogram, for removing unwanted hidden nodes from RBFNDDA. Histogram is a useful technique for visualising statistical approximation of data [34]. Since the histogram method is computationally efficient [22], we use it to prune the hidden nodes of RBFNDDA based on distribution of information. It should be noted that histogram is commonly used in signal processing but not machine learning (see the survey in Sect. 2.2 for more details). Therefore, we innovatively employ the histogram as a magnitude-based pruning method to tackle the problem associated with a large RBFNDDA network in this study. The proposed RBFNDDA-HIST model is explained in detail in Sect. 3.3.

2.2 Histogram

Histogram is a useful statistic method for data exploration. It can be used to summarise the distribution of data using density approximation. In principle, histogram, which is a non-parametric estimator [35], is able to approximate any density accurately as the sample size approaches infinity. Histogram has been used in a number of studies successfully, e.g. to represent the summary of multi-dimensional data [36], projective clustering [34, 35], signal processing, computer vision [37], and anomaly detection [38].

Histogram is useful for pruning in signal processing problems. It can be used to accelerate the search process [37, 39]. One of the important applications of histogram is to construct a large vocabulary for a continuous speech recognition system. Steinbiss et al. [40] used a histogram pruning method to limit the number of active hypotheses generated during the word recognition phase. A histogram score of active states is computed. If the number of active hypotheses is more than the threshold, the algorithm retains only the hypotheses with the best scores and prunes other hypotheses. Kashino et al. [41] proposed a feature histogram pruning method to search for the similarity between the query and stored signals in audio and video signal processing problems. The occurrence of each feature vector from the query and stored signals is computed to form the histogram. A distance measure is used to determine the similarity degree. If the similarity degree exceeds a pre-defined threshold, the query signal is retained as a stored signal [41]. Zhang and Liu [42] compared the feature histogram between the query and stored signals in a similarity-based audio retrieval process. The pruning algorithm does not require computation of the similarity measure between the query and stored signals.

Compared with signal processing, histogram has not been widely used in machine learning. Bors and Pitas [43] proposed a clustering algorithm to determine the parameter settings of an RBF network, known as median RBF. The centres and variances are found based on a method similar to learning vector quantisation. The network is then trained by using a robust statistical-based algorithm, where marginal median is used to estimate the centres of the RBF nodes, and median of absolute deviations (MAD) is used to estimate the covariance matrix. A data sample histogram is used to locate the centre median as well as to accelerate the calculation of median and MAD during the training process. Wei et al. [44] proposed the use of histogram-like plots to post-process the network outputs by assessing graphically the accuracy of posterior estimates with respect to individual classes. In this case, the histogram-like plots are used to re-map the sigmoid outputs of an ANN to improve the posterior probability estimation.

In this study, histogram is employed to refine the RBFNDDA structure. To the best of our knowledge, no histogram-based methods have been researched to generate a histogram of recognition nodes for reducing the RBF network complexity. This motivates us to integrate histogram into the RBFNDDA learning process to redress the negative effect of the network's greedy insertion behaviour. The details are presented in the next section.

3 The proposed model

We first describe the training algorithms of RBFNDDA and the histogram method. Then, how the histogram method is integrated into the RBFNDDA network is explained.

3.1 RBFNDDA

Berthold and Diamond [16] proposed DDA as the training algorithm to build an RBF network. It adopts the incremental learning capability of the Probabilistic Restricted Coulomb Energy (P-RCE) model, in which a new hidden node is introduced incrementally when a new pattern is wrongly classified by a conflicting class. Two user-defined parameters, i.e. the positive threshold (θ^+) and negative threshold (θ^-), are required to determine the width of an RBF node and also to distinguish the node from its neighbours of other classes. The θ^+ indicates the minimum correct-classification probability of a node for the correct class. The θ^- indicates the highest probability of a node that results in misclassification. The recommended settings of these two parameters are: $\theta^+ = 0.4$ and $\theta^- = 0.2$ [42, 43].

During each training iteration, all weights of the hidden nodes are initialised to zero. Given a training sample, if more than one RBF nodes have the activation level higher than θ^+, the weight of the hidden node with the highest activation level is increased. If the activation level of all hidden nodes is below θ^+, a new hidden node is introduced. This new hidden node uses the training sample as its centre, and its weight is set to one. Width shrinking is performed for all hidden nodes of different classes in the next step when their activation levels on the training sample are above θ^-. The training process continues until all training samples have been presented. Algorithm 1 shows a single training iteration of the RBFNDDA network.

Algorithm 1 Training algorithm of RBFNDDA

```
(1) For all hidden nodes $P_i^b$ for $i = 1, 2, \ldots, m_b,\ b = 1, 2, \ldots, B$, set
    the weight of each hidden node to zero, i.e. $w_i^b = 0$.
(2) Consider a training input, $x$, that belongs to class $b$, and assume
    that $P_i^b$ nodes in the network. If the Gaussian activation level
    of $P_i^b$, i.e. $R_i^b \geq \theta^+$ (the input is correctly classified), increase
    the weight of the hidden node with the largest $R_i^b$ by one, i.e.
    $w_i^b = w_i^b + 1$.
    Otherwise (when none of $P_i^b$ fulfils the condition), update the
    number of class $b$ nodes, $m_b = m_b + 1$;
    {
        Introduce input $\boldsymbol{x}$ as a new node $P_{m_b}^b$
        Set $w_{m_b}^b = 1$
        Set the centre of $P_{m_b}^b$, $r_{m_b}^b = x$
        Set the width of
    $P_{m_b}^b$, $wd_{m_b}^b = \min_{\substack{j \neq b \\ 1 \leq a \leq m_j}} \left\{ \sqrt{\frac{\|r_a^j - r_{m_b}^b\|^2}{\ln \theta^-}} \right\}$
    }
(3) Adjust the width of all conflicting nodes for $j \neq b,\ 1 \leq a \leq m_j$
    $wd_a^j = \min\left\{ wd_a^j, \sqrt{\frac{\|r_a^j - r_{m_b}^b\|^2}{\ln \theta^-}} \right\}$
(4) Steps (2) to (3) are repeated until all the training samples have
    been presented.
```

3.2 The histogram (HIST) algorithm

The HIST algorithm was proposed by Shimazaki and Shinomoto [45] to select the optimum bin size of a peri-stimulus time histogram (PSTH) and to determine the number of experiments required for finding a meaningful presentation in terms of resolution of the firing rate. It is a neurophysiology study of information transmission. This histogram is used to present the firing rate (or the rate of spike) of a neuron's onset of stimuli. The time duration is divided into several discrete bins. The height of a bin indicates the number of spikes. It is important to identify the number of spikes distributed within an appropriate set of bins. However, there is no systematic way for identifying the most appropriate number of bins in the existing PSTH experiments. As such, Shimazaki and Shinomoto [45] proposed a procedure to estimate an appropriate number of bins (bin size) and the interval of histogram (bin width) by minimising a cost function, $C_n(\Delta)$, as follows:

$$\begin{aligned} C_n(\Delta) &= \frac{2}{n\Delta}\left\langle E\hat{\theta}\right\rangle - \left\langle E\left(\hat{\theta} - \left\langle E\hat{\theta}\right\rangle\right)^2\right\rangle \\ &= \frac{2}{n\Delta}\frac{\bar{k}}{n\Delta} - \frac{1}{(n\Delta)^2}\left\langle (k_i - \bar{k})^2\right\rangle \\ &= \frac{2\bar{k} - v}{(n\Delta)^2} \end{aligned} \tag{1}$$

where $n =$ the number of sequence to obtain the firing rate; $\Delta =$ the bin width that represents the observed time period; $E\hat{\theta} =$ average spike count; $\hat{\theta} = \frac{k_i}{n\Delta}$ (the total number of spikes, k_i, that enters each bin i); and $v =$ variance. The algorithm can adapt the bin size and bin width to the problems under scrutiny. Algorithm 2 shows the procedure for bin size-optimisation:

Algorithm 2: Algorithm to optimise the bin size

(1) The observed time period is divided into N bins with width, Δ (N is the number of bins, $2 \leq N \leq 50$). The sequence, n, is set to 30. The number of spikes, k_i, that enters the ith bin is computed.

(2) The mean, $\bar{k}$, and variance, v, of the number of spikes are computed as follows

$$\bar{k} \equiv \frac{1}{N}\sum_{i=1}^{N} k_i \tag{2}$$

$$v \equiv \frac{1}{N}\sum_{i=1}^{N} (k_i - \bar{k})^2 \tag{3}$$

(3) The cost function (C_n) is computed as

$$C_n(\Delta) = \frac{2\bar{k} - v}{(n\Delta)^2} \tag{4}$$

(4) Steps (1) to (3) are repeated by varying the bin width, Δ, to search for the optimal bin size and bin width that minimise C_n.

3.3 The proposed RBFNDDA-HIST model

We employ the HIST algorithm [45] to identify and prune less important hidden nodes of RBFNDDA. As mentioned in [20, 21], it is a challenge to measure the relevance of network hidden nodes, while pruning always leads to a high computational cost. The proposed RBFNDDA-HIST model is able to preserve the nature of RBFNDDA, in which the number of hidden nodes grows automatically according to the requirement of the task in hand and HIST helps measure the relevance of hidden nodes by computing the distribution of information before pruning. In addition, HIST is simple, straightforward, and highly efficient in terms of computational cost [22].

We now explain the learning–pruning procedure of RBFNDDA-HIST. Firstly, RBFNDDA is built by applying the DDA algorithm. RBFNDDA comprises a group of hidden nodes, P, that belong to B classes, as follows.

$$P = \left\langle r_{1,\ldots,m_b}, w_{1,\ldots,m_b}, wd_{1,\ldots,m_b}\right\rangle, \quad b = 1, 2, \ldots, B \tag{5}$$

where r_i is the reference vector, w_i is the weight, and wd_i is the width of each hidden node. The hidden nodes in (5) can be re-organised according to their classes.

$$P' = \bigcup\nolimits_1^B P^b \tag{6}$$

Then, the hidden nodes, P', are propagated from RBFNDDA to HIST for the purpose of identifying and pruning superfluous nodes. While the nodes are in a multi-dimensional form, the HIST algorithm processes one-dimensional information. Therefore, the centre of each hidden node is transformed into a single-dimensional data sample ($D =$ number of dimensions) using an aggregate sum, as follows:

$$\text{Input}, X_i^b = \sum_{d=1}^{D} r_{d_i}^b \tag{7}$$

The HIST algorithm processes all Input X^b that belong to class b. In this study, the bins of the histogram have the same width, which is computed as follows:

$$\Delta = \frac{\text{Input}X_{\max}^b - \text{Input } X_{\min}^b}{N} \tag{8}$$

where N is the number of bins between 3 and 50. All inputs X^b and their corresponding w from RBFNDDA are divided into N bins with Δ, by varying the value of N incrementally from 3 to 50 according to Algorithm 2. All data samples to HIST are available after RBFNDDA training, and they are assembled in a single sequence. Therefore, the sequence, n, is set to 1. The optimum bin size is identified by referring to the smallest cost computed using Eq. 4. The histogram can be constructed based on the optimal bin size. However, which bin(s) is (are) less important and should be removed from the histogram? To address this issue, we extend Algorithm 2 by introducing the expected value $E\left(W_z^b\right)$ per bin z,

$$E\left(W_z^b\right) = \sum_{i=1}^{m} W_i p(W_i) \tag{9}$$

where

$$p(W_i) = \frac{\text{the number of Input } X^b \text{ from the category of } W_i \text{ within } a \text{ bin } z}{\text{total number of Input } X^b \text{ per bin } z} \quad (10)$$

m denotes the number of different W_i categories of input X^b in a bin. Each category refers to the RBFNDDA weight values, where weight w_i^b is considered as a category. As an example, three bins are used to form a histogram of all input X^b that belong to class b. In bin 1 (i.e. $z = 1$), the total number of input X^b is 10, and the W_i category of the RBFNDDA nodes includes 1, 3, and 5. In this case, $m = 3$. Furthermore, assume that 3 input X^b have $W_1 = 1$; 3 input X^b have $W_2 = 3$; and 4 input X^b have $W_3 = 5$. The calculation of $E(W_1^b)$, $E(W_2^b)$, and $E(W_3^b)$ based on the information of bins 1, 2, and 3 of class b is illustrated in Table 1. $E(W_z^b)$ of each bin is then normalised using a min–max method and is compared against a pruning threshold. $E(W_z^b)$ lower than the pruning threshold is deleted. Equations (7)–(10) are applied to construct the histogram of hidden nodes from other classes before pruning. The nodes, which are retrieved from the remaining bins of the histograms of all classes after the pruning process, are retained in the RBFNDDA network.

In this study, two versions of RBFNDDA-HIST are proposed:

- RBFNDDA-HIST1: RBFNDDA is trained with multiple epochs. Upon completion of the RBFNDDA learning process in multiple epochs, the HIST algorithm is applied to prune the hidden nodes. The purpose of this method is to remove all unwanted nodes at once before the pruned network is evaluated with the test samples. The details are shown in Algorithm 3.
- RBFNDDA-HIST2: Pruning is applied after each RBFNDDA training epoch. The less important nodes are pruned by the HIST algorithm before a new training epoch begins. In other words, RBFNDDA-HIST2 performs learning and pruning in each training epoch.

Algorithm 3 The training procedure of RBFNDDA-HIST1

```
(1) After RBFNDDA is trained, separate hidden nodes $P^b$ according
    to class $b$.
(2) Transform the centre of each hidden node of class $b$ into a one-
    dimensional input using Eq. 7.
(3) Find the optimal bin size that minimises the cost function as in
    Algorithm 2.
(4) A histogram is constructed by using the optimal bin size and
    width settings as in Algorithm 2.
(5) Compute the probability $p(W)$ for each bin according to Eq. 10.
(6) Obtain $E(W)$ for each bin using Eq. 9, and apply normalisation.
(7) Delete all bins with their normalised $E(W)$ lower than a pre-set
    threshold.
(8) Repeat Steps (2) to (7) for the hidden nodes of other classes.
(9) The pruned model proceeds to the test phase.
```

4 Experiments

Two sets of experiments were conducted to evaluate the effectiveness of RBFNDDA-HIST. The first experiment compared between RBFNDDA-HIST and RBFNDDA as well as with other methods [44, 46]. We used the benchmark data sets from the UCI repository [23]. The second experiment examined the usefulness of RBFNDDA-HIST in a real-world application, i.e. the circulating water (CW) system. The results were discussed and analysed, as in Sects. 4.1 and 4.2.

In all the experiments, the threshold parameters of RBFNDDA were set in accordance with the settings recommended by Berthold and Diamonds [16], i.e. $\theta^+ = 0.4$, $\theta^- = 0.2$, and the maximum training epoch = 6. For RBFNDDA-HIST, in addition to $\theta^+ = 0.4$, $\theta^- = 0.2$, N was set between 3 and 50, and the pruning threshold was set to 0.25. The original setting of N recommended by Shimazaki and Shinomoto [45] was any number of bins between 2 and 50. However, node reduction in RBFNDDA-HIST might not be beneficial due to limited

Table 1 An example of $E(W)$ computation

	Histogram of class b							
	Bin 1 ($m = 3$)			Bin 2 ($m = 2$)		Bin 3 ($m = 3$)		
W	1	3	5	1	2	2	5	9
$p(W)$	3/10	3/10	4/10	5/8	3/8	2/4	1/4	1/4
$E(W)$	3.20			1.38		4.50		
Normalised $E(W)$	0.58			0.00[a]		1.00		

[a] Remark: With a pruning threshold of 0.25, bin 2 is deleted because its normalised $E(W)$ is < 0.25

amount of information if the number of bins in a histogram were too small (i.e. 2). As such, N was set between 3 and 50 in this study. For the pruning threshold, after some trials, 0.25 was identified as the best setting for RBFNDDA-HIST in our experiments. The computing platform used to run experiments consisted of Windows 7, Intel Core i5-2410 M, and 4.0 GB RAM.

4.1 Benchmark study

4.1.1 Performance comparison between RBFNDDA and RBFNDDA-HIST

Nine benchmark data sets from the UCI repository, i.e. Parkinson, Wisconsin PBC, Wisconsin DBC, Blood transfuse, Thyroid, Banknote, Breast tissue, Planning-relax, and Indian liver, were used. Table 2 shows the details of the data sets. Most data sets have only two classes except Thyroid and Breast tissue that have three and four classes, respectively. The data samples were normalised between 0 and 1, and the missing data samples (in Wisconsin PBC) were replaced with zero. For each data set, 50 % of the data samples were used for training and the remaining for testing. The experiment was repeated 30 runs. The average results are presented in Table 3.

By referring to Tables 3 and 4, both RBFNDDA-HIST models reduced more than 50 % of nodes in Parkinson, Wisconsin PBC, and Wisconsin DBC, as compared with RBFNDDA. Both RBFNDDA-HIST models achieved 40–49 % of node reduction for Thyroid and Blood transfuse, 20–30 % for Breast tissue and Banknote, and about 20 % for Indian liver and Planning-relax data sets, respectively.

Table 2 Characteristics of the data sets

Data set	No. of class	Sample size	Attributes	No. of training	No. of testing
Parkinson	2	197	22	98	97
Wisconsin PBC	2	198	34	99	99
Wisconsin DBC	2	569	32	285	284
Blood transfuse	2	748	5	374	374
Thyroid	3	215	5	108	107
Banknote	2	1372	4	686	686
Breast tissue	4	106	9	53	53
Planning-relax	2	182	13	91	91
Indian liver	2	583	8	292	291

Table 3 Performance comparison between RBFNDDA and RBFNDDA-HIST (standard deviations are shown in parentheses)

Data set	RBFNDDA			RBFNDDA-HIST1			RBFNDDA-HIST2		
	Acc.	# Nodes	Time (s)	Acc.	# Nodes	Time (s)	Acc.	# Nodes	Time (s)
Parkinson	83.47	57.00	0.40	83.92	22.00	0.62	83.57	23.00	0.47
	(3.01)	(6.00)	(0.06)	(3.21)	(5.00)	(0.09)	(3.17)	(7.00)	(0.07)
Wisconsin PBC	76.03	81.00	0.54	75.52	34.00	0.50	75.49	34.00	0.59
	(2.67)	(12.00)	(0.08)	(3.17)	(10.00)	(0.07)	(3.13)	(10.00)	(0.10)
Wisconsin DBC	94.37	90.00	0.63	94.35	43.00	0.54	94.35	43.00	0.47
	(1.36)	(9.10)	(0.05)	(1.17)	(9.30)	(0.06)	(1.17)	(9.30)	(0.05)
Blood transfuse	61.56	654.00	19.09	62.36	329.00	13.63	61.92	353.00	10.06
	(2.68)	(55.00)	(1.04)	(3.06)	(37.00)	(1.20)	(2.83)	(43.00)	(1.06)
Thyroid	91.25	41.97	0.39	91.28	23.50	0.37	91.28	24.10	0.50
	(3.03)	(4.43)	(0.07)	(3.32)	(5.92)	(0.06)	(3.38)	(4.60)	(0.07)
Banknote	95.33	167.10	8.08	94.18	120.83	7.57	94.54	120.27	6.36
	(1.39)	(16.09)	(0.63)	(1.48)	(19.89)	(0.77)	(1.64)	(19.45)	(0.63)
Breast tissue	79.12	28.23	0.14	79.25	22.00	0.15	79.50	21.47	0.16
	(4.17)	(2.92)	(0.02)	(4.21)	(4.00)	(0.04)	(4.51)	(4.52)	(0.01)
Planning-relax	64.80	84.67	0.53	64.69	68.03	0.60	64.58	68.27	0.83
	(3.10)	(2.70)	(0.21)	(2.69)	(4.94)	(0.24)	(3.26)	(6.53)	(0.33)
Indian liver	74.19	262.62	4.96	74.39	213.36	4.71	74.27	218.94	3.92
	(2.25)	(5.42)	(2.88)	(2.33)	(7.41)	(2.65)	(2.22)	(10.28)	(1.38)

Table 4 Node reduction (in percentage)

	RBFNDDA-HIST1	RBFNDDA-HIST2
Parkinson	61.40	59.65
Wisconsin PBC	58.02	58.02
Wisconsin DBC	52.22	52.22
Blood transfuse	49.69	46.02
Thyroid	44.01	42.58
Banknote	27.69	28.03
Breast tissue	22.07	23.95
Planning-relax	19.65	19.37
Indian liver	18.76	16.63

Overall, we can conclude that both RBFNDDA-HIST models are able to reduce the number of nodes from RBFNDDA significantly. By referring to Fig. 2, RBFNDDA-HIST1 achieved a slightly higher average node reduction rate (i.e. 39.28 %) than that of RBFNDDA-HIST2 (i.e. 38.50 %) from all nine classification tasks. This is because some nodes removed earlier were re-inserted into RBFNDDA-HIST2 during the subsequent training cycles.

A paired t test was conducted to compare statistically the performances between RBFNDDA and RBFNDDA-HIST in terms of accuracy, number of nodes, and execution time. Table 5 lists the paired t test results. The performance of both models (i.e. RBFNDDA-HIST1 vs RBFNDDA and RBFNDDA-HIST2 vs RBFNDDA) was compared at the significant level of $\alpha = 0.05$. The average accuracy rates of RBFNDDA-HIST1 are significantly different from those of RBFNDDA in Blood transfuse, Banknote and Indian liver because the p values are lower than 0.05. By referring to Table 3, the average accuracy rates of RBFNDDA-HIST1 are higher than those of RBFNDDA in Blood transfuse and Indian liver. However, RBFNDDA has higher accuracy rates than RBFNDDA-HIST1 in Banknote. The accuracy rates of RBFNDDA-HIST2 in all classification tasks except Banknote are statistically the same as those of RBFNDDA.

The p values between the number of nodes of RBFNDDA versus RBFNDDA-HIST1 as well as

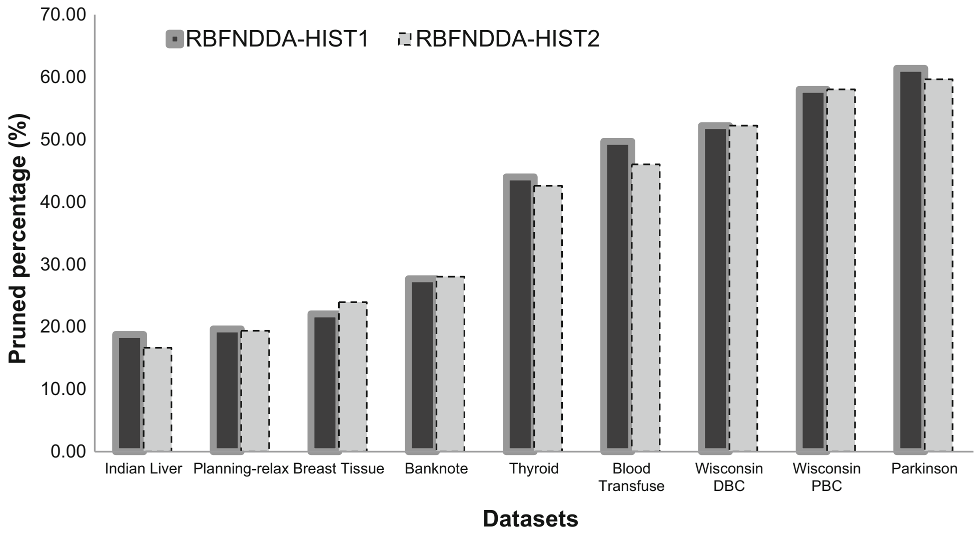

Fig. 2 Node reduction rates of RBFNDDA-HIST

Table 5 The p values of the paired t test

Hypothesis	RBFNDDA ≠ RBFNDDA-HIST1			RBFNDDA ≠ RBFNDDA-HIST2		
Data set	Acc.	# Nodes	Time (s)	Acc.	# Nodes	Time (s)
Parkinson	0.051[a]	0.000	0.879[a]	0.638[a]	0.000	0.001
Wisconsin PBC	0.122[a]	0.000	0.000	0.103[a]	0.000	0.006
Wisconsin DBC	0.930[a]	0.000	0.000	0.930[a]	0.000	0.000
Blood transfuse	0.010	0.000	0.000	0.149[a]	0.000	0.000
Thyroid	0.877[a]	0.000	0.026	0.893[a]	0.000	0.000
Banknote	0.000	0.000	0.000	0.001	0.000	0.000
Breast tissue	0.625[a]	0.000	0.078[a]	0.441[a]	0.000	0.000
Planning-relax	0.648[a]	0.000	0.100[a]	0.462[a]	0.000	0.000
Indian liver	0.048	0.000	0.620[a]	0.258[a]	0.000	0.025

[a] No significant difference

RBFNDDA versus RBFNDDA-HIST2 are all below 0.05. These results ascertain that the network sizes of both RBFNDDA-HIST models are significantly smaller than those of RBFNDDA.

In terms of the execution time, the paired *t* test reveals that RBFNDDA-HIST1 is significantly faster than RBFNDDA in Wisconsin PBC, Wisconsin DBC, Blood transfuse, Thyroid, and Banknote. For RBFNDDA-HIST2, we assume that the execution time is longer as pruning has been conducted in each epoch. However, RBFNDDA-HIST2 is significantly faster than RBFNDDA in Wisconsin DBC, Blood transfuse, Banknote, and Indian liver. These results show that both RBFNDDA-HIST have good computational efficiency.

In general, both RBFNDDA-HIST models are as accurate as RBFNDDA, except for the Blood transfuse, Indian liver, and Banknote data sets. The results in Tables 3 and 5 show that both RBFNDDA-HIST models can reduce unwanted nodes significantly. In addition, computational efficiency of RBFNDDA-HIST2 is better than that of RBFNDDA.

4.1.2 *Performance comparison with other methods*

In this section, we compare the classification performance of RBFNDDA-HIST with other ANNs, which include RBFN and Genetic algorithm RBFN (GA-RBFN) [47], and Probabilistic neural network (PNN) enhanced with structural minimisation methods [48], i.e. PNN-Kmeans and PNN-support vector machine (PNN-SVM). The data sets used in both [47, 48] are summarised in Table 6, which include Iris, Wisconsin BC, Pima Indian diabetes (Diabetes), Haberman's survival (Haberman), Cardiotocography (CTG), Thyroid 7200 (Thy7200), Dermatology, and Wisconsin DBC from the UCI machine learning repository.

Ding et al. [47] applied the GA to optimise both the weights and structure of an RBF network. A combination of binary and real encoding scheme was proposed to formulate the chromosomes. The binary encoding scheme was used to encode the network architecture, while the real encoding scheme was used to encode the weights between the hidden and output nodes of the RBF network. Both the binary and real parts of the selected chromosomes performed single-point crossover separately. The binary part of the chromosome performed bit-flipping mutation, whereas the real part employed Gaussian mutation. The chromosome with the lowest error rate was selected and was further trained using either the pseudo-inverse method or the least-mean-square method.

A number of RBFN models, as shown in Table 7, were evaluated using the Iris data set. The experiment was repeated 100 runs. To make a fair comparison, we followed the same experimental set-up in [47] for RBFNDDA-HIST. The results are listed in Table 8. RBFNDDA-HIST1 achieves the highest accuracy rate. RBFNDDA-HIST2 ranks third, after RBFNDDA-HIST1 and GA-RBFN-PI. Both RBFNDDA-HIST models have a smaller network size (in terms of the number of hidden nodes) than those reported in [47].

Both RBFNDDA-HIST models were also compared with two PNN networks, i.e. PNN-Kmeans and PNN-SVM [48]. The PNN is constructed using the Bayesian decision

Table 6 Characteristics of the benchmark data sets

Data set	No. of class	Sample size	Attributes	No. of training	No. of testing
Iris	3	150	4	105	45
Wisconsin BC	2	683	9	546	137
Diabetes	2	768	8	614	154
Haberman	2	306	3	245	61
CTG	3	2126	23	1701	425
Thy7200	3	7200	21	5760	1440
Dermatology	6	358	34	286	72
Wisconsin DBC	2	569	32	456	113

Table 7 RBFN-based ANNs reported in [47]

Algorithm	Definition
RBFN-PI	Original RBFN; PI = Pseudo-Inverse, PI is used to learn the weight
RBFN-LMS	Original RBFN; LMS = Least-Mean-Square, LMS is used to learn the weight
GA-RBFN	GA is used to learn the network structure and weight of RBFN
GA-RBFN-PI	GA finds the optimal network structure of RBFN and PI is then used to learn the weight
GA-RBF-LMS	GA finds the optimal network structure of RBFN and LMS is then used to learn the weight

Table 8 Performance comparison between RBFNDDA-HIST and the methods reported in [47]

Algorithm	Acc.	# Nodes
RBFNDDA-HIST1	95.10	23
RBFNDDA-HIST2	94.89	22
RBFN-PI	90.91	50
RBFN-LMS	91.80	50
GA-RBFN	86.71	32
GA-RBFN-PI	95.04	32
GA-RBFN-LMS	93.42	32

strategy [47, 48]. It learns very fast, but requires one node to represent each training sample [49]. This problem motivated Kusy and Kluska [48] to propose two alternative methods that could reduce the network complexity of PNN. PNN-Kmeans determined the optimal number of nodes using the k-means algorithm. PNN-SVM applied SVM to generate support vectors as patterns in the PNN hidden layer. Two parameters of PNN-SVM, i.e. *C* constraint and *sc* spread constant, were determined empirically. The performance varied with different parameter settings.

Table 9 presents the overall comparison results. It is clear that both RBFNDDA-HIST models perform better, or as good as, PNN-Kmeans and PNN-SVM. Table 9 shows that number of nodes in RBFNDDA-HIST is smaller than those of PNN-Kmeans and PNN-SVM in Wisconsin BC, Diabetes, CTG, Thy7200, and Dermatology. For Wisconsin DBC, both RBFNDDA-HIST models create fewer hidden nodes as compared with those of PNN-Kmeans, but more hidden nodes than those of PNN-SVM. Although both RBFNDDA-HIST models show inferior results in Haberman, on average, RBFNDDA-HIST1 and RBFNDDA-HIST2 contain fewer hidden nodes, i.e. 224 and 229, while PNN-Kmeans and PNN-SVM have 670 and 363 hidden nodes, respectively. The average classification rates of RBFNDDA-HIST are comparable with those of PNN-Kmeans and PNN-SVM as well. In summary, both RBFNDDA-HIST models are able to achieve comparable classification performance as those of PNN-Kmeans and PNN-SVM, but with smaller network sizes.

4.2 Circulating water system

The performance of RBFNDDA-HIST was evaluated using a real-world condition monitoring problem of a circulating water (CW) system in a power generation plant [50]. The CW system (see Fig. 3) is responsible to supply an adequate volume of cooling water from the sea to remove heat from steam in the condenser. The heat-transfer efficiency and blockage level of the condenser tubes can affect the operating conditions of the CW system. If the heat transfer process is efficient, the back-pressure of the turbine can be maintained at a low level, and this allows power to be generated with high efficiency. Besides that, solid materials such as shells and sands from the seawater accumulate in the tubes, resulting in blockage in the condenser and affecting heat transfer efficiency.

In this study, RBFNDDA-HIST was used to classify the health conditions of the CW system. A total of 2812 data samples with twelve attributes that indicated four different operation conditions of the CW system were collected, as follows.

- efficient heat transfer in the condenser and no significant blockage in the CW system
- inefficient heat transfer in the condenser but no significant blockage in the CW system
- efficient heat transfer in the condenser but significant blockage in the CW system

Table 9 Performance comparison between RBFNDDA-HIST and PNN-Kmeans and PNN-SVM

Algorithm	Wisconsin BC	Diabetes	Haberman	CTG	Thy7200	Dermatology	Wisconsin DBC	Average
RBFNDDA-HIST1								
Acc.	97.23	72.77	70.38	91.81	92.71	92.04	93.93	87.27
# Nodes	21	352	190	248	493	170	95	224
RBFNDDA-HIST2								
Acc.	97.23	73.90	71.97	91.34	92.71	91.39	93.87	87.49
# Nodes	24	352	168	254	541	170	98	229
PNN-Kmeans								
Acc.	93.43	71.43	75.41	88.47	93.54	91.67	90.27	86.32
# Nodes	55	368	25	1021	2881	201	137	670
PNN-SVM								
Acc.	91.97	68.83	73.77	92.94	93.13	93.06	92.04	86.53
# Nodes	244	551	135	288	1103	186	36	363

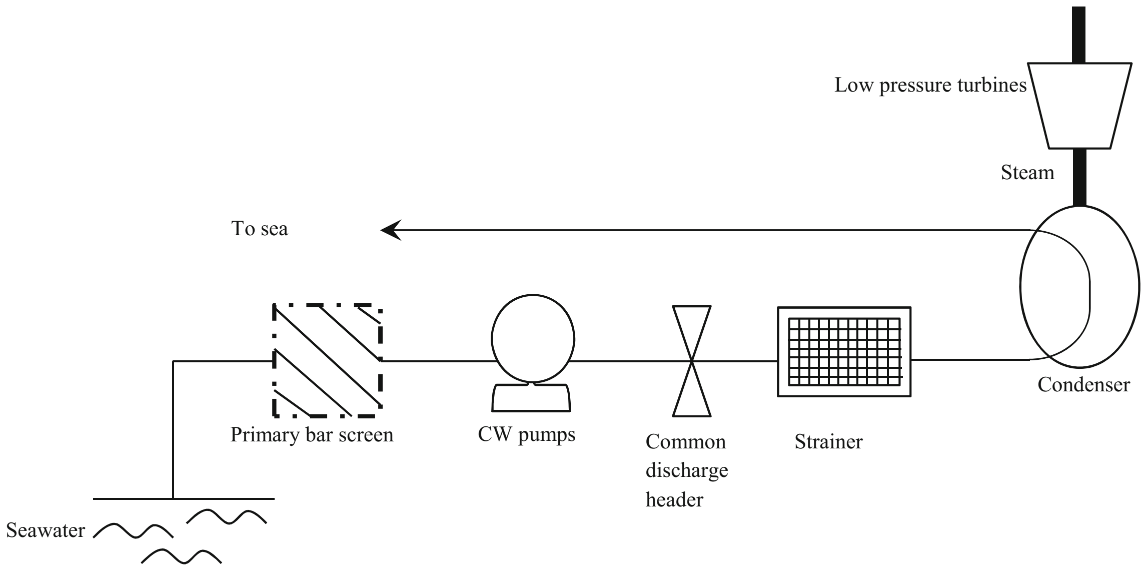

Fig. 3 The CW system

Table 10 Classification performances among RBFNDDA, RBFNDDA-HIST1, and RBFNDDA-HIST2

Algorithm	Acc.	# Nodes	Time (s)
RBFNDDA	94.81	310.38	38.94
	(1.02)	(16.35)	(19.68)
RBFNDDA-HIST1	94.36	90.66	34.67
	(0.97)	(23.99)	(11.52)
RBFNDDA-HIST2	93.56	102.44	25.07
	(1.25)	(23.98)	(12.55)

Table 11 Paired *t* test

Hypothesis	Acc.	# Nodes	Time (s)
RBFNDDA $\neq$ RBFNDDA-HIST1	0.066[a]	0.000	0.518[a]
RBFNDDA $\neq$ RBFNDDA-HIST2	0.000	0.000	0.002

[a] No significant difference

- inefficient heat transfer in the condenser and significant blockage in the CW system

In this experiment, 50 % of the data samples were allocated for training and the remaining for testing. The same parameter settings as in Sect. 4 were used. The experiment was repeated 30 times. Table 10 shows the results of RBFNDDA-HIST and RBFNDDA. Table 11 presents the results from the paired *t* test between each RBFNDDA-HIST model and RBFNDDA. Based on the results in Tables 10 and 11, both RBFNDDA-HIST models have significantly fewer hidden nodes than those of RBFNDDA, i.e. with 67 % of node reduction.

A data retention ratio (DRR) [51] is used to calculate the ratio of the number of hidden nodes to the total number of training samples, i.e.

$$\mathrm{DRR} = \frac{\text{no. of nodes}}{\text{no. of training samples}} \tag{12}$$

The DRR score of RBFNDDA was 0.22, while RBFNDDA-HIST1 achieved 0.06 and RBFNDDA-HIST2 achieved 0.07, respectively. In other words, both RBFNDDA-HIST models recruited fewer hidden nodes to represent 1406 training samples. The accuracy rates of RBFNDDA-HIST1 and RBFNDDA are statistically the same. The accuracy rate of RBFNDDA-HIST2 is slightly inferior by 1.25 % as compared with that of RBFNDDA. The execution time of RBFNDDA-HIST2 is significantly faster (25.07 s) than that of RBFNDDA (38.94 s). In general, comparing with RBFNDDA, both RBFNDDA-HIST models are able to tackle this condition monitoring problem with fewer nodes and with comparable accuracy rate.

5 Summary

In this paper, the RBFNDDA network has been integrated with a histogram algorithm to form RBFNDDA-HIST, with the purpose of reducing the network complexity of RBFNDDA in terms of hidden nodes. Two RBFNDDA-HIST models have been proposed, i.e. RBFNDDA-HIST1 and RBFNDDA-HIST2. Both RBFNDDA-HIST models have been evaluated using benchmark data sets from the UCI machine learning repository. The performances of both RBFNDDA-HIST models have been benchmarked against different classifiers, i.e. RBFN-PI, RBFN-LMS, GA-RBFN, GA-RBFN-PI, GA-RBFN-LMS, PNN-Kmeans, and PNN-SVM. The applicability of RBFNDDA-HIST models has also been demonstrated using a real-world condition monitoring problem in a power generation plant. Based on the experimental results, both RBFNDDA-HIST models are able to reduce the number of hidden nodes without deteriorating classification accuracy as compared with the original RBFNDDA network. Besides that, the computational time of RBFNDDA-HIST2 is generally more efficient. In short, both RBFNDDA-HIST models achieve improved results in terms of classification accuracy and network complexity as compared with those from the original RBFNDDA network.

For further work, both RBFNDDA-HIST models can be improved by equipping them with the capability of identifying abnormal data samples (e.g. outliers) in data-driven problems. The ultimate goal is to develop an RBFNDDA-based model capable of differentiating normal and abnormal data samples in order to achieve better generalisation. Then, the applicability of the resulting model to a variety of real-world data classification problems can be explored.

References

1. Zhang GP (2000) Neural networks for classification: a survey. IEEE Trans Syst Man Cybern Part C Appl Rev 30(4):451–462
2. Karnin ED (1990) A simple procedure for pruning back-propagation trained neural networks. IEEE Trans Neural Netw 1(2):239–242
3. Reed R (1993) Pruning algorithms—a survey. IEEE Trans Neural Netw 4(5):740–747
4. Yu H (2010) Network complexity analysis of multilayer feedforward artificial neural networks. In: Schumann J, Liu Y (eds) Applications of neural networks in high assurance systems se—3, vol 268., SpringerBerlin, Heidelberg, pp 41–55
5. Sabo D, Yu X-H (2008) Neural network dimension selection for dynamical system identification. In: IEEE international conference on control applications, pp 972–977
6. Zhang GP (2007) Avoiding pitfalls in neural network research. IEEE Trans Syst Man Cybern Part C Appl Rev 37(1):3–16
7. Medeiros CS, Barreto G (2013) A novel weight pruning method for MLP classifiers based on the MAXCORE principle. Neural Comput Appl 22(1):71–84
8. Abid S, Chtourou M, Djemel M (2013) Pseudo-entropy based pruning algorithm for feed forward neural networks. Aust J Basic Appl Sci 7(8):214–223
9. Yu H, Reiner PD, Xie T, Bartczak T, Wilamowski BM (2014) An incremental design of radial basis function networks. IEEE Trans Neural Netw Learn Syst 25(10):1793–1803
10. Montazer GA, Giveki D (2015) An improved radial basis function neural network for object image retrieval. Neurocomputing. doi:10.1016/j.neucom.2015.05.104
11. Hsu C-F, Lin C-M, Yeh R-G (2013) Supervisory adaptive dynamic RBF-based neural-fuzzy control system design for unknown nonlinear systems. Appl Soft Comput 13(4):1620–1626
12. Qasem SN, Shamsuddin SM (2011) Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. Appl Soft Comput 11(1):1427–1438
13. Ghodsi A, Schuurmans D (2003) Automatic basis selection techniques for RBF networks. Neural Netw 16(5–6):809–816
14. Polikar R, Upda L, Upda SS, Honavar V (2001) Learn ++: an incremental learning algorithm for supervised neural networks. IEEE Trans Syst Man Cybern Part C Appl Rev 31(4):497–508
15. Buschermohle A, Schoenke J, Rosemann N, Brockmann W (2013) The incremental risk functional: basics of a novel incremental learning approach. In: IEEE international conference on systems, man, and cybernetics (SMC), pp 1500–1505
16. Berthold MR, Diamond J (1995) Boosting the performance of RBF networks with dynamic decay adjustment. In: Tesauro G, Touretzky D, Alspector (eds) Advances in neural information processing, vol 7, p 8
17. Paetz J (2004) Reducing the number of neurons in radial basis function networks with dynamic decay adjustment. Neurocomputing 62:79–91
18. Narasimha PL, Delashmit WH, Manry MT, Li J, Maldonado F (2008) An integrated growing-pruning method for feedforward network training. Neurocomputing 71(13–15):2831–2847
19. Attik M, Bougrain L, Alexandre F (2005) Neural network topology optimization. In: Duch W, Kacprzyk J, Oja E, Zadrożny S (eds) Artificial neural networks: formal models and their applications—ICANN 2005 SE—9, vol 3697. Springer, Berlin, pp 53–58
20. García-Pedrajas N, Ortiz-Boyer D (2007) A cooperative constructive method for neural networks for pattern recognition. Pattern Recogn 40(1):80–98
21. Ma L, Khorasani K (2004) New training strategies for constructive neural networks with application to regression problems. Neural Netw 17(4):589–609
22. Legg PA, Rosin PL, Marshall D, Morgan JE (2007) Improving accuracy and efficiency of registration by mutual information using Sturges' histogram rule. In: Medical image understanding and analysis, pp 26–30
23. Bache K, Lichman M (2013) UCI machine learning repository. http://archive.ics.uci.edu/ml/
24. Anders U, Korn O (1999) Model selection in neural networks. Neural Netw 12(2):309–323
25. Heo G-S, Oh I-S (2008) Simultaneous node pruning of input and hidden layers using genetic algorithms. In: 2008 international conference on machine learning and cybernetics, vol 6, pp 3428–3433
26. Kaylani A, Georgiopoulos M, Mollaghasemi M, Anagnostopoulos GC (2009) AG-ART: an adaptive approach to evolving ART architectures. Neurocomputing 72(10–12):2079–2092
27. Huynh TQ, Setiono R (2005) Effective neural network pruning using cross-validation. In: IEEE international joint conference on neural networks (IJCNN), vol 2, pp 972–977
28. Leung CS, Wang H-J, Sum J (2010) On the selection of weight decay parameter for faulty networks. IEEE Trans Neural Netw 21(8):1232–1244
29. Martínez-Martínez JM, Escandell-Montero P, Soria-Olivas E, Martín-Guerrero JD, Magdalena-Benedito R, Gómez-Sanchis J (2011) Regularized extreme learning machine for regression problems. Neurocomputing 74(17):3716–3721
30. Huang G-B, Saratchandran P, Sundararajan N (2005) A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. IEEE Trans Neural Netw 16(1):57–67
31. Abbas H (2014) A decorrelation approach for pruning of multilayer perceptron networks. In: El Gayar N, Schwenker F, Suen C (eds) Artificial neural networks in pattern recognition SE 2, vol 8774. Springer, Berlin, pp 12–22

32. Leung C, Tsoi A (2005) Combined learning and pruning for recurrent radial basis function networks based on recursive least square algorithms. Neural Comput Appl 15(1):62–78
33. Chan PPK, Wu X-R, Ng WWY, Yeung DS (2011) Sensitivity based growing and pruning method for RBF network in online learning environments. In: 2011 international conference on machine learning and cybernetics (ICMLC), vol 3, pp 1107–1112
34. Ioannidis Y (2003) The history of histogram. In: Proceedings of the 29th international conference on very large data bases, vol 29, pp 19–30
35. Scott DW, Sain SR (2005) Multi-dimensional density estimation. In: Rao CR, Wegman EJ, Solka JL (eds) Handbook of statistics 24: data mining and visualization. Elsevier, Amsterdam, pp 229–261
36. Mazzeo GM, Sacca D, Sirangelo C (2005) Hierarchical binary histograms for summarizing multi-dimensional data. In: Proceedings of the 2005 ACM symposium on applied computing, pp 598–603
37. Tjahyadi R, Liu W, Svetha Venkatesh (2004) Application of the DCT energy histogram for face recognition. In: 2nd international conference on information technology for application (ICITA), pp 305–310
38. Goldstein M, Dengel A. (2012) Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm. In: 35th German conference on artificial intelligence (KI-2012), pp 59– 63
39. Gao S, Zhang C, Chen W-B. (2010) A variable bin width histogram based image clustering algorithm. In: 2010 IEEE fourth international conference on semantic computing (ICSC), pp 166–171
40. Steinbiss V, Tran B-H, Ney H (1994) Improvement in beam search. In: International conference on spoken language processing, pp 2143–2146
41. Kashino K, Kurozumi T, Murase H (2003) A quick search method for audio and video signals based on histogram pruning. IEEE Trans Multimed 5(3):348–357
42. Zhang WQ, Liu J (2007) Two-stage method for specific audio retrieval. In: IEEE International conference on acoustics, speech and signal processing (ICASSP 2007), vol 4, pp IV–85–IV–88
43. Bors AG, Pitas I (1996) Median radial basis function neural network. IEEE Trans Neural Netw 7(6):1351–1364
44. Wei W, Leen TK, Barnard E (1999) A fast histogram-based postprocessor that improves posterior probability estimates. Neural Comput 11(5):1235–1248
45. Shimazaki H, Shinomoto S (2007) A method for selecting the bin size of a time histogram. J Neural Comput 19(6):1503–1527
46. Paetz J (2006) Evolution of real valued weights for RBF-DDA networks. In: International joint conference on neural networks (IJCNN), pp 2907–2913
47. Ding S, Xu L, Su C, Jin F (2012) An optimizing method of RBF neural network based on genetic algorithm. Neural Comput Appl 21(2):333–336
48. Kusy M, Kluska J (2013) Probabilistic neural network structure reduction for medical data classification. In: Rutkowski L, Korytkowski M, Scherer R, Tadeusiewicz R, Zadeh L, Zurada J (eds) Artificial intelligence and soft computing SE—11, vol 7894. Springer, Berlin, pp 118–129
49. Specht DF (1992) Enhancements to probabilistic neural networks. In: International Joint Conference on Neural Networks (IJCNN), vol 1, pp 761–768
50. Tenaga National Sdn Bhd Malaysia (1999) System description and operating procedures, vol 14
51. Lam W, Keung C-K, Danyu Liu (2002) Discovering useful concepts prototypes for classification based on filtering and abstration. IEEE Trans Pattern Anal Mach Intell 24(8):1075–1090