Nuclear Engineering and Design 241 (2011) 2549–2557

Contents lists available at ScienceDirect

Nuclear Engineering and Design

journal homepage: www.elsevier.com/locate/nucengdes

Optimal artificial neural network architecture selection for performance prediction of compact heat exchanger with the EBaLM-OTR technique

Dumidu Wijayasekara[a,*], Milos Manic[a], Piyush Sabharwall[b], Vivek Utgikar[c]

[a] Department of Computer Science, University of Idaho, 1776 Science Center Drive, Idaho Falls, ID 83402, USA
[b] Idaho National Laboratory, Idaho Falls, ID, USA
[c] Department of Chemical Engineering, University of Idaho, Idaho Falls, ID 83402, USA

ARTICLE INFO

Article history:
Received 19 October 2010
Received in revised form 23 April 2011
Accepted 26 April 2011

ABSTRACT

Artificial Neural Networks (ANN) have been used in the past to predict the performance of printed circuit heat exchangers (PCHE) with satisfactory accuracy. Typically published literature has focused on optimizing ANN using a training dataset to train the network and a testing dataset to evaluate it. Although this may produce outputs that agree with experimental results, there is a risk of over-training or over-learning the network rather than generalizing it, which should be the ultimate goal. An over-trained network is able to produce good results with the training dataset but fails when new datasets with subtle changes are introduced. In this paper we present EBaLM-OTR (error back propagation and Levenberg-Marquardt algorithms for over training resilience) technique, which is based on a previously discussed method of selecting neural network architecture that uses a separate validation set to evaluate different network architectures based on mean square error (MSE), and standard deviation of MSE. The method uses *k*-fold cross validation. Therefore in order to select the optimal architecture for the problem, the dataset is divided into three parts which are used to train, validate and test each network architecture. Then each architecture is evaluated according to their generalization capability and capability to conform to original data. The method proved to be a comprehensive tool in identifying the weaknesses and advantages of different network architectures. The method also highlighted the fact that the architecture with the lowest training error is not always the most generalized and therefore not the optimal. Using the method the testing error achieved was in the order of magnitude of within 10^{-5}–10^{-3}. It was also show that the absolute error achieved by EBaLM-OTR was an order of magnitude better than the lowest error achieved by EBaLM-THP.

1. Introduction

Compact heat exchangers such as printed circuit heat exchangers (PCHE) are widely considered to be the next step in heat exchanger technology in nuclear power plants. The US Department Of Energy is focusing on Very High Temperature Reactors (VHTR) and Gas-Cooled Fast Reactors (GFR), both of which benefit greatly from advances in heat exchanger technology (Gezelius et al., 2004; Kim et al., 2008).

These next generation reactors will be used in Next Generation Nuclear Plant (NGNP) (Sabharwall et al., 2009), which will most likely produce electricity and process heat for various different industries (Sabharwall et al., 2010). To utilize the generated heat efficiently a thermal device is needed to transfer the thermal energy (Sabharwall, 2009).

High allowable pressure and temperature limits along with the compactness make PCHE a better choice for heat exchanger applications than other compact heat exchangers. The manufacturing process of PCHE ensures no gaskets or braze material is used, thus reducing the risk and fluid incompatibility substantially. The construction also allows greater number of flow configurations, which increases the versatility of PCHE. The compact size of PCHE also makes it more cost effective. Compared to standard heat exchangers designed for the same thermal duty and pressure drop PCHE is 4–6 times smaller (Sabharwall, 2009).

Artificial neural networks (ANNs) have been used in the past to accurately predict thermohydraulic models (Gongnan et al., 2009; Su et al., 2002; Tan et al., 2009; Yang and McClain, 1999). They have also been used to predict heat exchanger performance with minimal error (Diaz et al., 2001; Ermis, 2008; Mandavgane and Pandharipande, 2006; Patra et al., 2010; Ridluan et al., 2009). Recently, Ridluan et al., 2009 introduced EBaLM-THP algorithm that combines two well known algorithms (Error Back Propagation for robustness and Levenberg-Marquardt for speed) to model the per-

* Corresponding author. Tel.: +1 208 533 8158.
E-mail address: wija2589@vandals.uidaho.edu (D. Wijayasekara).

0029-5493/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.nucengdes.2011.04.045

formance of PCHE. In his work Ridluan et al. demonstrated superior performance of ANNs relative to comparable approaches, when it comes to predictive modeling of PCHE.

Ridluan's work however is not addressing the important issue of over-training of neural networks. Over-training or over-fitting is a common phenomenon in ANN and can lead to some misleading results (Binos, 2003; Wright and Manic, 2010). A network is over-trained when it conforms exactly to the training data rather than generalizing it. This produces an optimal network with very low approximation errors, but only for the training dataset. Testing an over-trained network on a similar dataset will also produce very low error, but when data with subtle differences are introduced to such a network, the errors start to dramatically increase (Wright and Manic, 2010).

The chance of over-training of a network increases with the number of neurons and number of training runs. A single neuron which has one input and one output is capable of modeling a single behavior of the input which can be represented by $y = mx + c$. With increasing number of neurons and connections the complexity of the behavior increases. Therefore with a higher number of neurons, there is a higher risk of the network learning exactly the training data and not generalizing. Similarly as the learning process progresses the network tries to reduce the approximation error, and given enough training epochs, the network will conform to the training outputs exactly thus resulting in an over-trained network.

Previous work in ANN architecture selection proposes numerous methods for architecture selection (Hopp and Prechelt, 1998; Mazrou, 2009; Wright and Manic, 2010), but these methods are either inadequate or too complex for performance prediction of PCHE since the available dataset is small. Wright and Manic (2010) focus on architecture selection for a data clustering problem, thus for a modeling problem the parameters used by Wright and Manic (2010) cannot be used. The process used by Hopp and Prechelt, 1998 modifies the network topology during training, which is extremely complicated and requires high processing power. Mazrou, 2009 focuses on determining the learning algorithm and transfer functions of the networks based on convergence time and mean square error.

The EBaLM-OTR (error back propagation and Levenberg-Marquardt algorithms for over training resilience) technique presented in this paper uses a variation of the method proposed by Wright and Manic (2010) to select the optimum architecture for this problem. The presented method uses mean square errors (MSE) of training, validating and testing errors to evaluate architectures according to their generalization capabilities as well as their ability to conform to experimental data. This facilitates the selection of the networks that are not over-trained or under-trained. The methodology also focuses on the stability of the outputs by evaluating standard deviations of MSE.

This paper also focuses on the fact that the performance of a network highly depends on the weight initialization and the adequate selection of training dataset. If the initial weights of a neuron happen to be chosen in such a way that network produces output values far from the desired ones, then the network takes longer time to reach the optimal value (or it might not reach the optimal value at all). Regarding the training data, if the selected dataset does not reflect the total behavior of all the possible values of the problem space then the ANN cannot learn the total behavior of that problem.

Therefore when selecting the optimal ANN architecture we must consider different initialization weight sets and different training datasets in order to find the neural network architecture, optimal relative to these factors but also network resilient to over-training. The proposed method of selecting the optimal neural network architecture can be used for a problem that has a small set of initial data and can overcome errors of weight initialization and selecting initial dataset.

In order to determine the optimal architecture, the architectures with varying numbers of neurons in the hidden layers have been investigated. These were trained using k-fold cross validation (Setiono, 2001) and tested with a separate dataset in order to evaluate them. In k-fold cross validation, the dataset is divided into k sets of same size. Then k-1 sets are used for training and the remaining set is used for validation. This is iterated k times using a different fold for validation in each iteration (Setiono, 2001; Wright and Manic, 2010).

In order to guarantee the stability of the performance of networks, each architecture was trained multiple times and mean MSE and standard deviation of MSE of each architecture was used as a selection criterion. Different vectors were used to rank architectures according to their conforming capabilities and generalization capabilities. The standard deviation was used as a stability measure when evaluating architectures.

The rest of the paper is organized as follows: Section 2 provides a brief description on artificial neural networks and the EBaLM-THP algorithm, Section 3 describes PCHE and summarizes the dataset used, Section 4 describes the EBaLM-OTR architecture selection algorithm, Section 5 discusses the results of the analysis and Section 6 provides the conclusion.

2. Artificial neural network algorithms

This section provides a description of artificial neural networks and the EBaLM-THP algorithm.

2.1. Artificial neural networks

Artificial neural networks are computational intelligence architectures based on biological neural networks and have the capability of "learning" the behavior of input data.

The basic unit of an ANN is a neuron. An artificial neuron acts in the same way as a biological neuron; each has a set of inputs and produces an output based on the inputs.

A biological neuron produces an output by comparing the sum of each input to a threshold value. Based on that comparison it produces an output. In addition, it is able to differently weigh each input according to the priority of the input. The inputs and outputs of a biological neuron are called synapses and these synapses may act as inputs to other neurons or as outputs such as muscles. Thus it creates an interconnected network of neurons which combined produce an output based on a number of weights, sums and comparisons.

An artificial neuron aims at achieving the same by using different input vectors, weights, a threshold value and output vectors. For each input x_q there is a weight w_q assigned. The neuron calculates the weighted sum z as:

$$\text{net} = \sum_{q=1}^{n} w_q x_q \tag{1}$$

The output of the neuron is governed by the activation function, which acts as a threshold. The output is given by:

$$o = f_s\left(\sum_{q=1}^{n} w_q x_q\right) \tag{2}$$

where $f_s(x)$ is the sigmoid activation function:

$$f_s(x) = \frac{1}{1 + e^{\lambda_s x}} \tag{3}$$

A neural network consists of multiple interconnected artificial neurons, arranged in several layers. Fig. 1 shows the typical arrangement of neurons in an artificial neural network. In Fig. 1

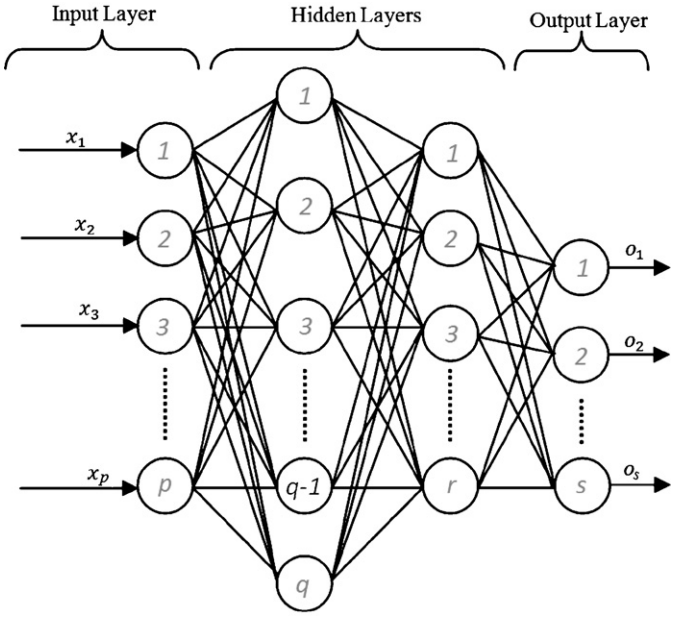

Fig. 1. Artificial neural network architecture.

each circle depicts an artificial neuron. The neurons are arranged in layers, there are one input and one output layer and multiple hidden layers. The neurons in the input layer have the activation function $f_s(x) = x$.

The output of neuron i in layer $l+1$ is calculated as:

$$o_i^{l+1} = \sum_{j=1}^{S_l} w_{ij}^{l+1} o_j^l + b_i^{l+1} \tag{4}$$

where S_l denotes the number of neurons in layer l, w_{ij}^{l+1} is the weight of the connection from neuron j in layer l, b_j^{l+1} is the bias of neuron i and o_j^l is the output of neuron j in layer l.

The output of neuron i in layer $l+1$ is given by:

$$o_i^{l+1} = f_s^{l+1}(x_i^{l+1}) \tag{5}$$

For a given layer L we can calculate the error if the desired output is known using:

$$E = \sum_{p=1}^{P}\sum_{m=1}^{M}(d_{pm} - o_{pm}^{L})^2 \tag{6}$$

where P is the number of patterns, M is the number of outputs and d_{pm} is the desired output pattern p and output m.

2.2. EBaLM-THP algorithm

The EBaLM-THP algorithm presented by Ridluan et al., 2009 is a combination of Error Back Propagation (EBP) and Levenberg–Marquardt (LM) algorithms, and hence benefits from the advantages both these algorithms provide.

The EBP algorithm was first described by Rumelhart and McClelland, 1986 and later elaborated by Werbos, 1994. The main advantage of using EBP is the ability to have multi layer ANN and propagate the errors calculated at the output neurons to the previous layers. The propagated error is then used to calculate the change in weight w_q which is Δw_q using the following equation:

$$\Delta w_q = \alpha \sum_{m=1}^{M}\sum_{p=1}^{P}[(d_{pm} - o_{pm})F'\{a_p\}f_s'(net_p)x_p] \tag{7}$$

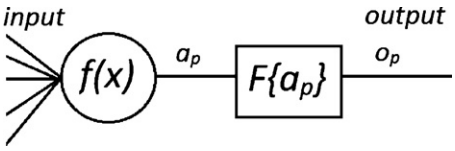

Fig. 2. Transfer function for a single neuron.

where $F'\{a_p\}$ is the first derivative of the transfer function between neurons as depicted in Fig. 2. α is called the learning constant which governs the speed of learning and is set to a number between 0 and 1. $f_s'(\text{net}_p)$ is the first derivative of the activation function $f_s(x)$.

The weight vector for the next iteration is calculated using the weight change, Δw_q as follows:

$$w_{new} = w_q + \Delta w_q \tag{8}$$

where w_{new} is the new weight vector for the next iteration.

The steepest descent method for updating weights, changes weights in such a way that the weight change of a neuron maximizes the change in total error. That is the derivative of the total error with respect to the weight, which is called the gradient vector given by Eq. (9), must be maximized.

$$g = \begin{bmatrix} \partial E/\partial w_1 \\ \partial E/\partial w_2 \\ \partial E/\partial w_3 \\ \cdots \\ \partial E/\partial w_N \end{bmatrix} \tag{9}$$

where g is the gradient vector, and E is the total error calculated using Eq. (6). The new weight vector is calculated in the steepest descent method as:

$$w_{\text{new}} = w_q - \alpha g \tag{10}$$

Thus this method increases the rate of learning for a neural network drastically. The Newton method further increases the rate of learning by including the Hessian matrix in the calculation of the new weight vector:

$$w_{new} = w_q - A_q^{-1} g \tag{11}$$

where A_q is the Hessian matrix for the weight vector q which is defined as:

$$A = \begin{bmatrix} \partial^2 E/\partial w_1^2 & \partial^2 E/\partial w_2 \partial w_1 & \cdots & \partial^2 E/\partial w_N \partial w_1 \\ \partial^2 E/\partial w_1 \partial w_2 & \partial^2 E/\partial w_2^2 & \cdots & \partial^2 E/\partial w_N \partial w_1 \\ \cdots & & & \\ \partial^2 E/\partial w_1 \partial w_N & \partial^2 E/\partial w_2 \partial w_N & \cdots & \partial^2 E/\partial w_N^2 \end{bmatrix} \tag{12}$$

Although the convergence speed of the Newton method is high the calculation of second order derivative is computationally expensive. Thus the Levenberg-Marquardt (LM) algorithm was introduced by Levenberg (1944) and Marquardt (1963) which retains the faster convergence of the Newton method but reduces the computation time drastically. The LM algorithm uses the Jacobian matrix to derive the gradient and Hessian matrix as follows:

$$A \approx 2J^{\mathrm{T}}J \tag{13}$$

$$g = 2J^{\mathrm{T}}e \tag{14}$$

where J is the Jacobian matrix, J^{T} is the transpose of the Jacobian and e is the error vector. Thus in the LM algorithm the new weight vector w_{new} is calculated as follows:

$$w_{\text{new}} = w_q - (J_q^{\mathrm{T}}J_q + \mu I)^{-1} J_q^{\mathrm{T}} e \tag{15}$$

where I is the identity matrix and μ is a constant between 0 and 1.

The EBaLM-THP algorithm combines EBP and LM algorithms to simulate thermohydraulic models. Thus EBaLM-THP combines the

Table 1
Summary of PCHE experimental data by Ishizuka et al., 2006.

	Inlet Pressure (MPa)	Inlet Temperature (°C)	Mass Flow Rate (kg/hr)	Pressure Drop (kPa)	Heat Transfer (kW)
Hot side	2.2–3.5	150–280	35–90	5.99–26.66	1.624–4.324
Cold Side	6.5–10.7	60–120	35–90	20.09–93.07	

ability to propagate errors through layers of an ANN and the high speed convergence and low computation requirements of the LM algorithm.

3. PCHE

Printed Circuit Heat Exchanger (PCHE) was first introduced in 1985 by Heatric company for refrigeration applications and is a compact heat exchanger type with good heat transfer performance (Ridluan, 2009; Sabharwall et al., 2007). PCHE consist of channels of small hydraulic diameter and has high durability under pressure (Kim et al., 2008). The compact size and high performance make it an ideal choice in modern heat exchanger applications such as petro chemical power plants and fuel cells (Ridluan et al., 2009).

Technological advances in recent years have enabled the company to extend the application of PCHE. More recently PCHE has been considered the key heat exchanger in the indirect Brayton cycle featured in many advanced nuclear systems, including the NGNP-VHTR and GFR (Ridluan et al., 2009).

The PCHE core is made up of plates with chemically etched flow paths that are semi-circular grooves called flow channels, ranging from 0.5 to 5.0 mm in depth. Because of the small size of the flow channels, the heat transfer area per volume of the heat exchanger is high (Ridluan, 2009). These plates are then stacked and diffusion bonded into a single element. Diffusion bonding allows the plates to achieve near parent metal strength (Meter, 2006). The PCHE channels could have a zigzag pattern which has been shown to enhance heat transfer coefficients considerably (Kim et al., 2009; Ridluan, 2009).

The effectiveness of PCHE can be 98% or higher. Possible working temperatures for these heat exchangers range from cryogenic to 900 °C. Pressure differences vary from 20 MPa to 40 MPa (Ridluan, 2009).

One of the main problems in modeling PCHE using conventional methods is the size of flow channels: fluid flow and heat transfer characteristics inside small channels are different from conventionally sized channels. Another problem is if the PCHE has a zigzag pattern: validated correlations for prediction of heat transfer performance of wavy channels are not available. Because of these effects numerical prediction of a PCHE is difficult (Ridluan, 2009).

An experimental dataset for a PCHE using Supercritical CO_2 (SCO_2) was reported by Ishizuka et al., 2006 which was also used by Ridluan et al., 2009. Excerpt data from this work is given in Table 1. The duty of the PCHE reported by Ishizuka was 3 kW; PCHE consisted of 12 hot and 11 cold semi-circular, zigzagged flow channel plates, made of SS316L austentic Chromium-Nickel stainless steel with superior corrosion resistance. The zigzag was defined by 115° and 100° for hot and cold plates, respectively. The cross-sectional areas of hot and cold channel are 0.0002 and 0.000092 m^2, respectively (Ridluan, 2009).

The data contains five control variables and three ouput variables. The control varibles (system inputs) are: mass flow rate, inlet hot sided pressure, inlet cold sided pressure, inlet hot sided temperature and inlet cold sided temperature. The output varibles are hot sided pressure drop, cold sided pressure drop and the heat transfer (Fig. 3).

4. EBaLM-OTR–ann architecture selection for pche modeling

As mentioned above, artificial neural networks (ANNs) have been used for thermal hydraulic applications and have demonstrated good results (Diaz et al., 2001; Mandavgane and Pandharipande, 2006; Ridluan et al., 2009).

ANNs have also been used to model and predict the performance of various heat exchangers with high accuracy (Pacheco-Vega et al., 2001; Peng and Ling et al., 2009; Xie et al., 2007).

Ridluan et al. (2009) used ANN to predict the performance of PCHE, and demonstrated superiority of ANNs as universal approximators relative to comparable techniques.

In this paper we build upon EBaLM-THP (Ridluan et al., 2009) and present EBaLM-OTR approach to alleviate the problem of Over-training. EBaLM approach is based on the Error Back Propagation (EBP) and Levenberg-Marquardt (LM) algorithms for their abilities to conform to training data faster, more accurately than others (Mazrou, 2009; Ridluan et al., 2009). LM algorithm also produces lower training MSE compared to other learning algorithms in Mazrou, 2009.

The EBaLM-OTR algorithm also uses Error Back Propagation (EBP) and Levenberg-Marquardt (LM) algorithms for neural network training. However EBaLM-OTR method sub-divides the data set into training, testing and validation sets and produces two measures that are used to compare different architectures. Using these measures EBaLM-OTR extracts the architecture that is most over-training resilient.

4.1. EBaLM-OTR Algorithm

The proposed method uses k-fold cross validation, which divides the dataset into k number of similar sized segments or folds and uses one fold for validating and one fold for training. However in order to generate a more generalized result the proposed method uses one fold for testing, one fold for validating and the rest of the folds for training. Thus the dataset Y is divided into three portions containing P_{tr} number of training data patterns (Y_{tr}), P_{te} number

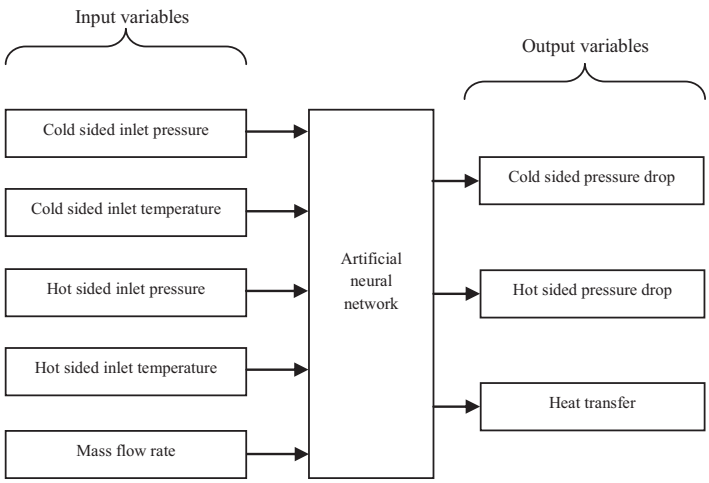

Fig. 3. Input and output variables of the ANN.

```
1. initialize the network with a set of random
weights
2. reorder the data points randomly
3. divide the dataset into k folds
4. FOR all k
5.     WHILE training error < 10^-5 OR 500 iterations
6.       train the network using Y_tr
7.     END WHILE
8.     test the network using Y_te
9.     validate the network using Y_va
10.    calculate training MSE
11.    calculate testing MSE
12.    calculate validating MSE
13. END FOR
```

Fig. 4. Pseudo code of evaluation of one network architecture.

of testing data patterns (Y_{te}) and P_{va} number of validating data patterns (Y_{va}), where:

$$P = P_{tr} + P_{te} + P_{va} \tag{16}$$

$$Y = \{Y_{tr} \cup Y_{te} \cup Y_{va}\} \tag{17}$$

The dataset Y contains P number of data patterns y_i and y_i is an $n+m$ dimensional vector containing n number of inputs and m number of desired outputs, i.e.

$$y_i = \{x_1, x_2, \ldots, x_n, d_1, d_2, \ldots, d_m\} \tag{18}$$

where x_i are inputs and d_i are desired outputs.

As mentioned, for the modeling of PCHE, the number of input dimensions was 5 and number of output dimensions was 1, thus n was 5 and m was 1.

After training was completed the MSE of the training set (MSE_{tr}), MSE of the testing set (MSE_{te}) and MSE of the validation set (MSE_{va}) were calculated using the following equations:

$$MSE_{tr} = \sum_{p=1}^{P_{tr}} (d_p - o_p)^2 \tag{19}$$

$$MSE_{te} = \sum_{p=1}^{P_{te}} (d_p - o_p)^2 \tag{20}$$

$$MSE_{va} = \sum_{p=1}^{P_{va}} (d_p - o_p)^2 \tag{21}$$

where d_p is the desired output for each input pattern and o_p is the actual output produced by the neural network. Thus MSE is a dimensionless value calculated to compare neural network architecture performance.

Fig. 4 shows the pseudo code for evaluation of one ANN architecture. At step1 the network is initialized using a random set of weights. Then at step 2 and 3 the dataset is reordered randomly and divided into training, testing and validating sets. At step 6 the network is trained using the training dataset (Y_{tr}). The training is performed until the network achieves a predefined error or reaches a certain number of iterations (step 5). The number of iterations was set to 500 and the goal error was set to 10^{-5}, these stopping criteria for training were calculated by training randomly selected networks and calculating MSE and the number of iterations it took for the networks to achieve that error.

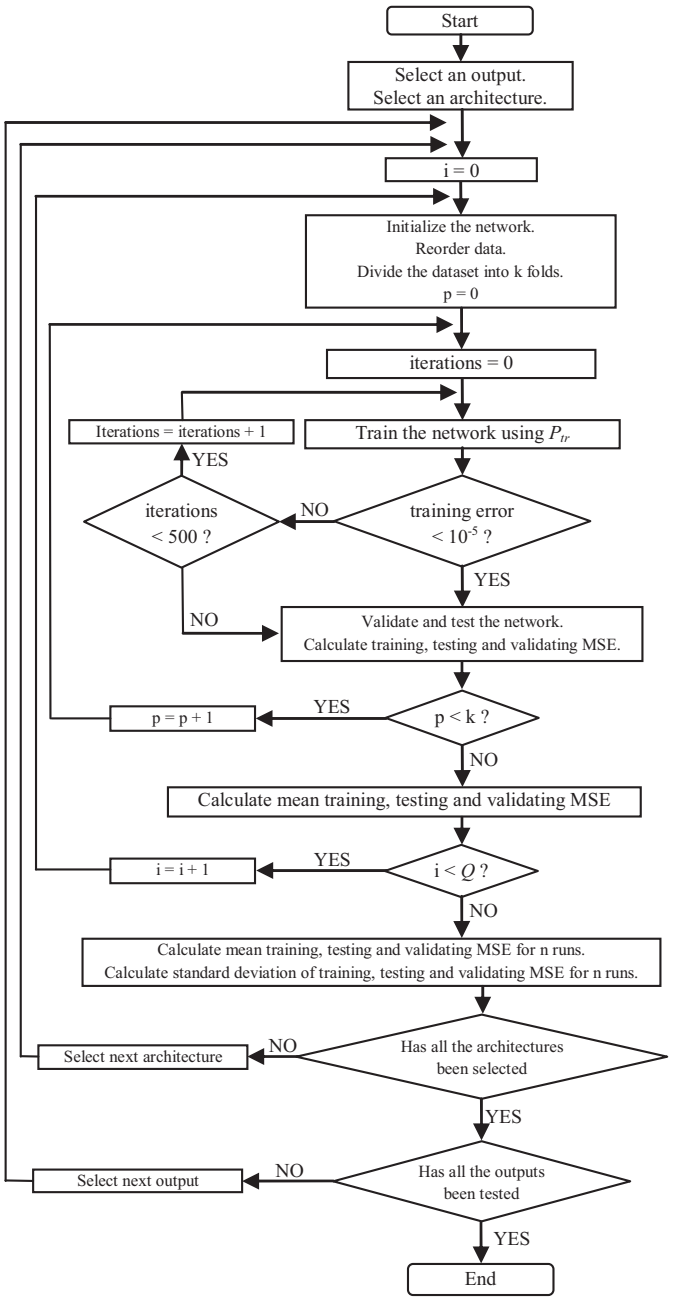

Fig. 5. Flowchart of evaluation of network architecture.

After the training is completed the network is tested using Y_{te} and then validated using Y_{va} (steps 8 and 9). Finally the MSE values are calculated using Eqs. (19)–(21) (steps 10–11). Steps 5 through 12 are repeated for all the folds of the dataset.

For each architecture steps 1 through 13 are repeated Q times and mean and standard deviation of training, testing and validation MSE were calculated. This process was repeated for all the architectures tested. Fig. 5 shows the complete algorithm in a flow chart format.

After repeating the above process Q times the mean MSE (mMSE) and standard deviation of MSE (sdMSE) were calculated for each architecture using Eqs. (22) and (23), respectively.

$$m\text{MSE} = \frac{\sum_{i=1}^{Q} \text{MSE}_i}{Q} \tag{22}$$

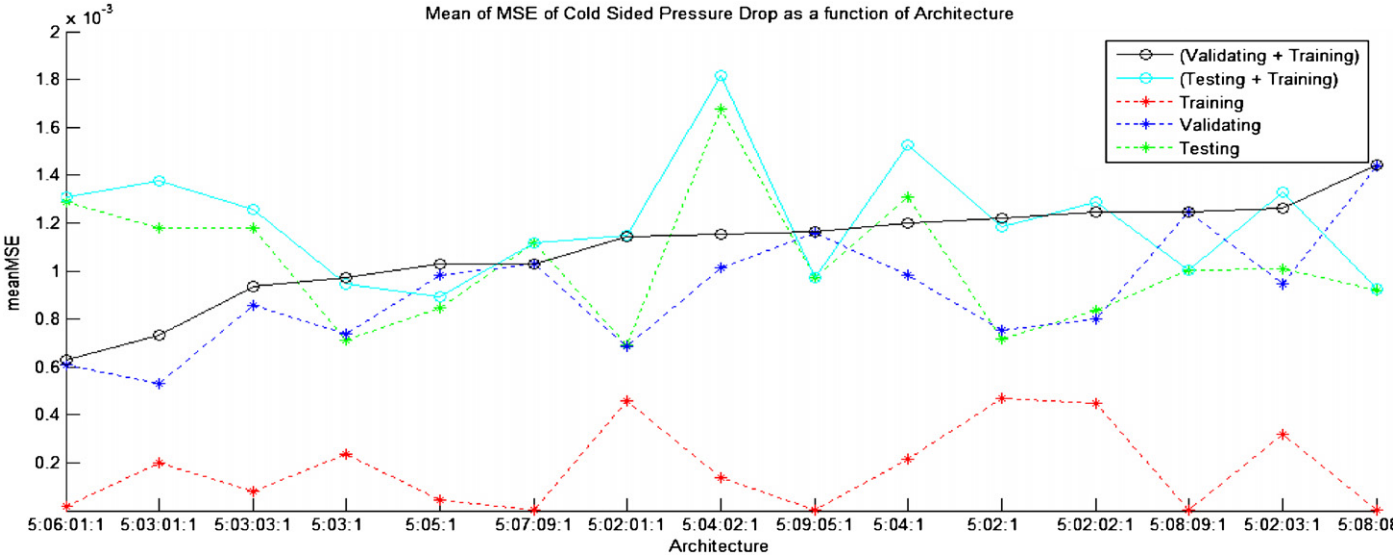

Fig. 6. Conforming capability of network architectures for cold sided pressure drop.

$$sd\text{MSE} = \sqrt{\frac{1}{Q}\sum_{i=1}^{Q}(\text{MSE}_i - m\text{MSE})^2} \quad (23)$$

The goal of the analysis was to evaluate all the network architectures and select the architecture that can model the data with highest accuracy and is most generalized. In order to find the architecture that conforms to the data most, the sum of mean MSE of the training set and mean MSE of the validating set was used. This shows the architecture with the lowest errors for both training and validating, i.e. the architectures that predict the experimental data with lowest errors. The mean MSE of the testing set is used to further evaluate the conforming capability of the network. The most generalized architecture would have close training and validating mean MSE. Therefore to evaluate the generalization capabilities of the architecture, the difference between mean MSE values for training and validating were compared. Therefore the primary evaluation vectors were conforming capability (*Conf*) and generalizing capability (*Gen*), calculated as follows:

$$\text{Conf} = m\text{MSE}_{\text{tr}} + m\text{MSE}_{\text{va}} \quad (24)$$

$$\text{Gen} = m\text{MSE}_{\text{va}} - m\text{MSE}_{\text{tr}} \quad (25)$$

The mean MSE was used to evaluate architectures based on conforming capability and generalizing capability. The standard deviation of MSE was used to evaluate the architecture's stability in producing results, and the architectures with high standard deviations were eliminated.

4.2. EBaLM-OTR Algorithm applied to PCHE modeling

For this PCHE analysis we chose the dataset referenced in Section 3. This dataset has three output dimensions and five input dimensions. Due to different scales and nature of the measurements and values of each output dimension, neural networks were designed so that each network produced one output. Thus each network has five input neurons and one output neuron. Therefore each output was considered separately and three different architectures were selected, one for each output.

Architectures were labeled as *p:q:r:s* or *p:q:s* depending on the number of hidden layers, where *q* and *r* are the number of neurons in the hidden layers and *p* and *s* are the number of neurons in the input and output layers respectively (Fig. 1). For example an architecture labeled 5:3:4:1 has two hidden layers containing 3 and 4 neurons in each. The input layer contains 5 neurons and the output layer contains 1 neuron. Similarly 5:4:1 is an architecture with one hidden layer containing 4 neurons and its input layer contains 5 neurons output layer 1 neuron.

In order to find the optimal architecture that can model each output vector, 100 different neural network architectures were tested for each output dimension. The number of neurons in the hidden layers varied from 1 to 19. The number of neurons in the first hidden layer ranged from 1 to 10 and the number of neurons in the second hidden layer ranged from 0 to 9. The tested architectures ranged from 5:1:1 to 5:9:9:1. This range was selected for testing because pre-testing using fewer neurons produced large errors and architectures having more neurons in more layers would be too computationally intensive for practical use.

Each architecture was tested 10 times. The dataset was divided into five folds which were used for 5-fold cross validation and one fold was used for testing. Five training runs per simulation were done using different folds for training testing and validating.

4.3. Improvements of EBaLM-OTR over EBaLM-THP

The improvements of EBaLM-OTR for architecture selection compared to EBaLM-THP can be elaborated as follows. First since the EBaLM-OTR algorithm uses a validation set to validate the training, it is immune to over-training and thus able to extract the optimal architecture that minimizes over-training and maximizes generalizability. Therefore compared to EBaLM-THP, architectures derived from EBaLM-OTR are more robust and less prone to errors in real world applications and when dealing with noisy data.

Secondly since the algorithm trains and tests each architecture multiple times and uses the standard deviation of the MSE as a selection criterion, the architecture chosen by the EBaLM-OTR algorithm is more consistent in producing low error outputs, compared to EBaLM-THP.

Thirdly the dependency of the LM algorithm on the initial weight set (Wilamowski, 2003) is alleviated by running the algorithm multiple times using different randomly initialized weights each time.

Such improvements allow the EBaLM-OTR algorithm to be used to derive optimal ANN architecture for any problem that requires a robust and consistent prediction algorithm.

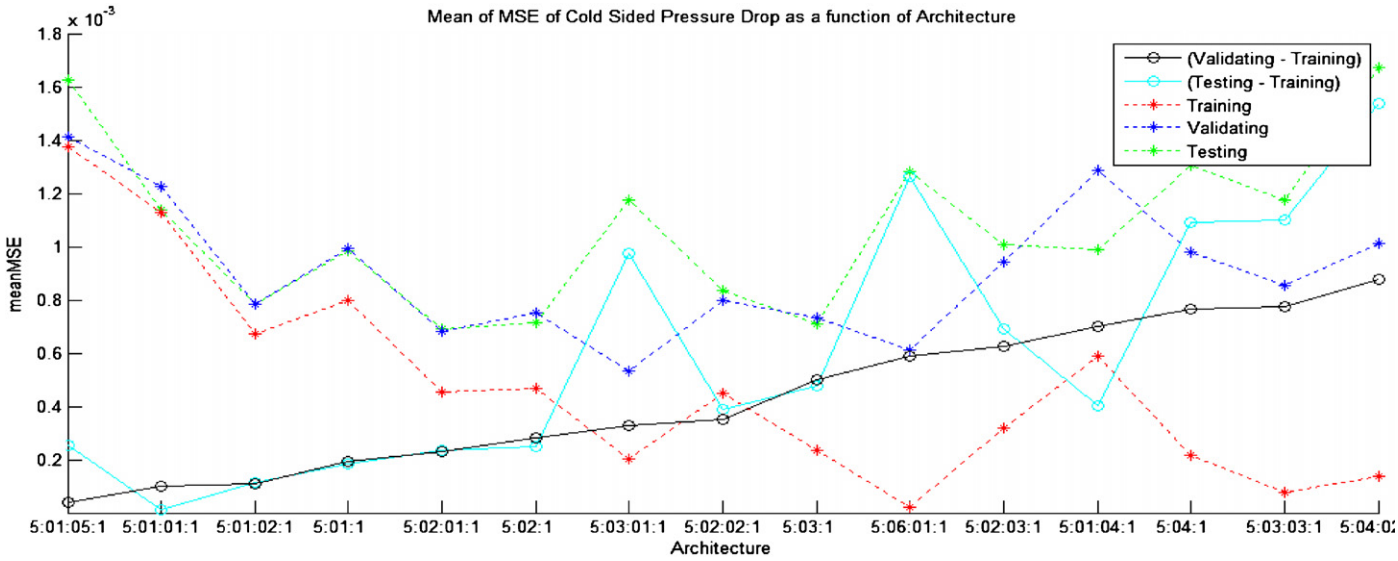

Fig. 7. Generalization capability of network architectures for cold sided pressure drop.

5. Results and discussion

Each architecture was trained and tested 10 times using different data patterns for training and testing, after the runs were completed mean and standard deviation of training, validation and testing MSE were stored for each architecture. The primary selection criteria of architectures were the sum and difference of training and validation mean MSE.

After the optimal architecture for each output was obtained, the selected architectures were tested using the experimental dataset presented Ishizuka et al., 2006. The absolute errors and the mean absolute errors produced by the EBaLM-OTR algorithm and EBaLM-THP algorithm presented by Ridluan et al., 2009 were compared. It was shown that the architectures selected using EBaLM-OTR produced lower errors than EBaLM-THP.

5.1. Architecture selection

Fig. 6 shows 15 architectures that conform to the experimental data with least error for the cold sided pressure drop. The architectures are sorted in the ascending order of $mMSE_{va} + mMSE_{tr}$. Similarly Fig. 7 shows 15 architectures that are most generalized for the prediction of cold sided pressure drop i.e. architectures that have lowest $mMSE_{va} - mMSE_{tr}$. Architectures in Fig. 7 are in the ascending order of $mMSE_{va} - mMSE_{tr}$ value. In order to select the optimal architecture for cold sided pressure drop the architecture with the lowest mean MSE in both Fig. 6 and Fig. 7 must be selected.

Table 2 lists the architectures that are most generalized and highly conform to the experimental data for cold sided pressure drop i.e. it list architectures that are present in both Fig. 6 and Fig. 7. Along with the sum and difference of training and validation mean MSE the table lists the rank of architectures. The rank is the sum of the positions of architectures in each figure. For example architecture 5:3:1:1 appears in Fig. 6 in the second position, and in Fig. 7 in the seventh position, thus architecture 5:3:1:1 has a rank of 2 + 7 (Table 2). The positions of the architectures in each graph give a selection vector that can be used as a measurement of performance. The table also list the training mean MSE and standard deviations MSE which indicating the stability of the architecture.

The architecture to be selected is the one with the lowest mean MSE (for ex. in Fig. 6, that architecture is 5:6:1:1). However the same architecture (5:6:1:1) in Fig. 7 appears in the 10th position meaning its generalizing capability is low. The second architecture in Fig. 6 (5:3:1:1) proves higher generalization capability compared to 5:6:1:1. In fact the architecture with highest generalization capability that appears in both figures is architecture 5:2:1:1 which is in 7th position in Fig. 5 and 5th position in Fig. 7.

Similarly if one selects the architecture in the lowest position in Fig. 7 which is 5:1:5:1, we can see that although 5:1:5:1 has a high generalization capability, the training, testing and validating errors are high, which would put it in a higher position in Fig. 6. In this case the $mMSE_{tr}$ and $mMSE_{va}$ are so high that they are off the chart in Fig. 6.

Therefore in order to select the optimal architecture positions in Fig. 6 and Fig. 7 must be taken into account. In order to quantify the lowest positions in both Fig. 6 and Fig. 7, the sum of the positions of architectures in each figure is chosen as the first selection criterion. Table 2 lists architectures according to this criterion. From Table 2

Table 2
Selection criteria for the optimal architecture for the cold sided pressure drop.

Architecture	Validation + training	Validation – training	Rank	Training mean MSE
5:3:1:1	0.733×10^{-3}	3.31×10^{-4}	9	0.0012 ± 0.0072
5:6:1:1	0.6317×10^{-3}	5.903×10^{-4}	11	0.0013 ± 0.0067
5:2:1:1	1.142×10^{-3}	2.287×10^{-4}	12	0.00069 ± 0.0039
5:3:1	0.9715×10^{-3}	5.014×10^{-4}	13	0.00071 ± 0.0039
5:3:3:1	0.9338×10^{-3}	7.78×10^{-4}	17	0.0012 ± 0.0042
5:2:2:1	1.246×10^{-3}	3.501×10^{-4}	20	0.00084 ± 0.0071
5:4:1	1.201×10^{-3}	7.662×10^{-4}	23	0.0013 ± 0.0227
5:4:2:1	1.153×10^{-3}	8.775×10^{-4}	23	0.0017 ± 0.0074
5:2:3:1	1.261×10^{-3}	6.257×10^{-4}	25	0.01 ± 0.0089
5:2:1	1.1219×10^{-3}	2.838×10^{-4}	27	0.00072 ± 0.0044

Table 3
Performance of the selected architectures.

Architecture	Cold sided pressure drop			Hot sided pressure drop			Heat transfer		
	$m\text{MSE}_{tr}$	$m\text{MSE}_{va}$	$m\text{MSE}_{te}$	$m\text{MSE}_{tr}$	$m\text{MSE}_{va}$	$m\text{MSE}_{te}$	$m\text{MSE}_{tr}$	$m\text{MSE}_{va}$	$m\text{MSE}_{te}$
5:2:1:1	0.4564×10^{-3}	0.6851×10^{-3}	0.6920×10^{-3}	0.1324×10^{-3}	0.2022×10^{-3}	0.2054×10^{-3}	0.4473×10^{-4}	0.7160×10^{-4}	0.6992×10^{-4}
5:1:5:1	0.0014	0.0014	0.0016	0.1251×10^{-3}	0.1935×10^{-3}	0.1917×10^{-3}	0.3321×10^{-4}	0.7575×10^{-4}	0.7441×10^{-4}
5:7:9:1	9.667×10^{-7}	0.001	0.0011	9.6×10^{-7}	8.36×10^{-4}	0.001	0.0085×10^{-4}	0.2395×10^{-4}	0.3245×10^{-4}

Table 4
Mean absolute error of EBaLM-OTR compared to EBaLM-THP.

Variable	EBaLM-THP Architecture1	EBaLM-THP Architecture2	EBaLM-OTR
Cold sided pressure drop (kPa)	4.5372	6.1291	1.2515
Hot sided pressure drop (kPa)	1.1999	1.0795	3.2232×10^{-1}
Heat exchange (kW)	1.0703×10^{-1}	7.8027×10^{-2}	9.9899×10^{-3}

top three architectures for cold sided pressure drop were selected as 5:3:1:1, 5:6:1:1 and 5:2:1:1.

The second selection criterion was the testing error. Out of the three selected architectures 5:3:1:1 and 5:6:1:1 show high testing MSE while 5:2:1:1 produces significantly lower testing MSE, therefore architecture 5:2:1:1 was selected as the optimal architecture to model cold sided pressure drop.

Finally the standard deviations of the architectures were compared to other tested architectures. The standard deviation of MSE is a measure of the stability of architectures.

The standard deviation of all three mean MSE for the cold sided pressure drop ranged from 9.7×10^{-4} to 4.3×10^{-2} for all the architectures tested. The standard deviations for the top 3 architectures were less than 6.7×10^{-3}, therefore the standard deviations of the MSE are low for all three selected architectures meaning that they were stable in producing results.

Using the same method architectures that are optimal for modeling of hot sided pressure drop and heat transfer were extracted. The architectures selected were 5:1:5:1 and 5:7:9:1 for the hot sided pressure drop and heat transfer respectively.

Table 3 lists the three selected architectures for each output, and the performance of each of the architectures relative to each of the outputs.

5.2. Comparison with EBaLM-THP

After the selection of architectures the absolute errors of the selected architectures were compared with the best performing architectures presented in Ridluan et al., 2009. Ridluan et al., 2009 presented two architectures that produced the best results for PCHE modeling, which were 5:7:5:3 (EBaLM-THP Architecture1) and 5:9:7:3 (EBaLM-THP Architecture2). Table 4 shows the mean absolute error achieved by each algorithm for each output dimension, it can be clearly seen that the EBaLM-OTR architectures produce lower absolute errors for all three outputs.

6. Conclusions

This manuscript presents EBaLM-OTR technique for the analysis and selection of the most optimal ANN architecture for PCHE modeling that alleviates the problem of over training. It successfully identified over-trained networks as well as networks that cannot conform to the data satisfactorily. The technique also demonstrated the ability to overcome problems such as small training set and the effect of initial weights on the network training.

Using the EBaLM-OTR technique it was possible to identify the architectures that are most generalized and conform to each output variable in most optimal way. The selected architectures were able to predict each output with an error within 10^{-5}–10^{-3} order of magnitude.

The selected architectures also reflected the complexity of each output. Hot and cold sided pressure drop were successfully modeled using architectures with low number of neurons but in order to model the heat transfer the number of neurons in the architecture needed to be higher i.e. heat transfer was more complex and had more variation than hot and cold sided pressure drops.

The results also show that it is impossible to predict the optimal architecture for the given problem prior to modeling such as presented by EBaLM-OTR. For some problems architectures with higher number of neurons produce better results. But for some problems not only architectures with lower number of neurons suffice but they are able to produce better results than architecture with higher number of neurons.

References

Binos, T., 2003. Evolving neural network architecture and weights using an evolutionary algorithm. M.S. Thesis, Department of Computer Science, Royal Melbourne Institute of Technology, Melbourne, Australia.

Diaz, G., Sen, M., Yang, K.T., McClain, R.L., 2001. Dynamic prediction and control of heat exchangers using ANN. International Journal of Heat and Mass Transfer 44, 1671–1679.

Ermis, K., 2008. ANN modeling of compact heat exchangers. International Journal of Energy Research 32, 581–594.

Gezelius, K., Driscoll, M.J., Hejzlar, P., 2004. Design of compact heat exchangers for compact gas cooled fast reactors. MIT-ANP-TR 103.

Gongnan, X., Sunden, B., Wang, Q., Tang, L., 2009. Performance prediction of laminar and turbulent heat transfer and fluid flow of heat exchangers having large tube-row by artificial neural networks. International Journal of Heat and Mass Transfer 52, 2484–2497.

Hopp, H., Prechelt, L., 1998. Portable and efficient high-level parallel programming of neural networks. Systems Analysis Modeling Simulation (SAMS) journal.

Ishizuka, T., Kato, Y., Muto, Y., Nikitin, K., Ngo, T.L., 2006. Thermal–hydraulic characteristics of a printed circuit heat exchanger in a supercritical CO_2 loop Bull Res Lab Nucl. Reactor 30, 109–116.

Kim, D.E., Kim, M.H., Cha, J.E., Kim, S.O., 2008. Numerical investigation on thermal–hydraulic performance of new printed circuit heat exchanger model. Nuclear Engineering and Design 238, 3269–3276.

Kim, I.H., No, H.C., Lee, J.I., Guk Jeon, B., 2009. Thermal hydraulic performance analysis of the printed circuit heat exchanger using a helium test facility and CFD simulations. Nuclear Engineering and Design 239, 2399–2408.

Levenberg, K., 1944. A method for the solution of certain non-linear problems in least squares. The Quarterly of Applied Mathematics 2, 164–168.

Mandavgane, S.A., Pandharipande, S.L., 2006. Application of ANN for modeling of heat exchanger with concentration as variable. Indian Journal of Chemical Technology 13, 173–176.

Marquardt, D., 1963. An algorithm for least-squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics 11, 431–441.

Mazrou, H., 2009. Performance improvement of artificial neural networks designed for safety key parameters prediction in nuclear research reactors. Nuclear Engineering and Design 239, 1901–1910.

Meter, J.V., 2006. Experimental investigation of a printed circuit heat exchanger using supercritical carbon dioxide and water as heat transfer media. M.S. Thesis, Kansas State University, Manhattan, Kansas.

Pacheco-Vega, A., Sen, M., Yang, K.T., McClain, R.L., 2001. Neural network analysis of fin-tube refrigerating heat exchanger with limited experimental data. International Journal of Heat and Mass Transfer 44, 763–770.

Patra, S.R., Jahadeesan, R., Rajeswari, S., Satyamurthy, S.A.V., 2010. ANN model for intermediate heat exchanger for nuclear reactor. International Journal of Computer Applications 1, 65–72.

Peng, H., Ling, X., 2009. Neural network analysis of thermal characteristics on plate-fin heat exchangers with limited experimental data. Applied Thermal Engineering 29, 2251–2256.

Ridluan, A., 2009. Design optimization of a thermal component using cfd as the design tool. Doctoral Dissertation, University of Idaho, Idaho Falls, Idaho.

Ridluan, A., Manic, M., Tokuhiro, A., 2009. EBaLM-THP – A neural network thermohydraulic prediction of advanced nuclear system components. Nuclear Engineering and Design 239, 308–319.

Rumelhart, D.E., McClelland, J.L., 1986. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press. Cambridge, MA. vol. 1.

Sabharwall, P., Gunnerson, F., Tokuhiro, A., Utgiker, V., Weaver, K., Sherman, S., 2007. Theoretical Design of a Thermosyphon for Efficient Process Heat Removal from Next Generation Nuclear Plant (NGNP) for Production of Hydrogen. Idaho National Laboratory and University of Idaho, INL/EXT-07-13433.

Sabharwall, P., 2009. Engineering Design Elements of a Two-Phase Thermosyphon to Transfer NGNP Thermal Energy to a Hydrogen Plant. INL/EXT-09-15383. Idaho National Laboratory Next Generation Nuclear Plant Project.

Sabharwall, P., Utgikar, V., Tokuhiro, A., Gunnerson, F., 2009. Design of liquid metal phase change heat exchanger for next-generation nuclear plant process heat application. Journal of Nuclear Science and Technology 46, 534–544.

Sabharwall, P., Patterson, M., Utgikar, V., Gunnerson, F., 2010. Phase change heat transfer device for process heat applications. Nuclear Engineering and Design 240, 2409–2414.

Setiono, R., 2001. Feedforward neural network construction using cross validation. Neural Computation 13, 2865–2877.

Su, G., Fukuda, K., Jia, D., Morita, K., 2002. Application of an artificial neural network in reactor thermohydraulic problem: prediction of critical heat flux. Journal of Nuclear Science and Technology 39, 564–571.

Tan, C.K., Ward, J., Wilcox, S.J., Payne, R., 2009. Artificial neural network modeling of the thermal performance of a compact heat exchanger. Applied Thermal Engineering 29, 3609–3617.

Werbos, P.J., 1994. The Roots of Backpropagation. From Ordered Derivatives to Neural Networks and Political Forecasting. John Wiley & Sons, New York.

Wilamowski, B. M., 2003. Neural Network Architectures and Training. International Conference on Industrial Technology. TU1-T12.

Wright, J. L., Manic, M., 2010. Neural network architecture selection analysis with application to cryptography location. IEEE World Congress on Computational Intelligence, Barcelona, Spain. 2941–2946.

Xie, G.N., Wang, Q.W., Zeng, M., Luo, L.Q., 2007. Heat transfer analysis for shell-and-tube heat exchangers with experimental data by artificial neural network approach. Applied Thermal Engineering. 27, 1096–1104.

Yang, K.T., McClain, K.T., 1999. Simulation of heat exchanger performance by ANN. International Journal HVAC & R Research 5, 195–208.