



Article

Modelling and Computational Experiment to Obtain Optimized Neural Network for Battery Thermal Management Data

Asif Afzal ^{1,2,*} , Javed Khan Bhutto ³, Abdulrahman Alrobaian ⁴, Abdul Razak Kaladgi ^{1,*} 
and Sher Afghan Khan ⁵

¹ Department of Mechanical Engineering, P. A. College of Engineering, Visvesvaraya Technological University, Mangaluru 574153, India

² Department of Mechanical Engineering, School of Technology, Glocal University, Delhi-Yamunotri Marg, SH-57, Mirzapur Pole, Saharanpur District, Uttar Pradesh 247121, India

³ Department of Electrical Engineering, College of Engineering, King Khalid University, P.O. Box 394, Abha 61421, Saudi Arabia; jbhutto@kku.edu.sa

⁴ Department of Mechanical Engineering, College of Engineering, Qassim University, Buraydah 51452, Saudi Arabia; alrobaian@qu.edu.sa

⁵ Department of Mechanical Engineering, Faculty of Engineering, International Islamic University, Kuala Lumpur 50728, Malaysia; sakhan06@gmail.com

* Correspondence: asif.afzal86@gmail.com (A.A.); abdulkaladgi@gmail.com (A.R.K.)

Abstract: The focus of this work is to computationally obtain an optimized neural network (NN) model to predict battery average Nusselt number (Nu_{avg}) data using four activations functions. The battery Nu_{avg} is highly nonlinear as reported in the literature, which depends mainly on flow velocity, coolant type, heat generation, thermal conductivity, battery length to width ratio, and space between the parallel battery packs. Nu_{avg} is modeled at first using only one hidden layer in the network (NN_1). The neurons in NN_1 are experimented from 1 to 10 with activation functions: Sigmoidal, Gaussian, Tanh, and Linear functions to get the optimized NN_1 . Similarly, deep NN (NN_D) was also analyzed with neurons and activations functions to find an optimized number of hidden layers to predict the Nu_{avg} . RSME (root mean square error) and R-Squared (R^2) is accessed to conclude the optimized NN model. From this computational experiment, it is found that NN_1 and NN_D both accurately predict the battery data. Six neurons in the hidden layer for NN_1 give the best predictions. Sigmoidal and Gaussian functions have provided the best results for the NN_1 model. In NN_D , the optimized model is obtained at different hidden layers and neurons for each activation function. The Sigmoidal and Gaussian functions outperformed the Tanh and Linear functions in an NN_1 model. The linear function, on the other hand, was unable to forecast the battery data adequately. The Gaussian and Linear functions outperformed the other two NN-operated functions in the NN_D model. Overall, the deep NN (NN_D) model predicted better than the single-layered NN (NN_1) model for each activation function.

Keywords: neural network; battery; heat transfer; activation functions; hidden layers



Citation: Afzal, A.; Bhutto, J.K.; Alrobaian, A.; Razak Kaladgi, A.; Khan, S.A. Modelling and Computational Experiment to Obtain Optimized Neural Network for Battery Thermal Management Data. *Energies* **2021**, *14*, 7370. <https://doi.org/10.3390/en14217370>

Academic Editor: Andrea De Pascale

Received: 11 October 2021

Accepted: 2 November 2021

Published: 5 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In modern electric cars, the battery thermal management system (BTMS) is a prominent aspect in the efficient design and long life of a lithium-ion module. To carry out a comprehensive analysis of this system, a large number of experiments are necessary. Owing to the complex nature of BTMS and a wide range of operating conditions, experimental investigations are both expensive and time-consuming. Artificial neural network (ANN) techniques, on the other hand, can be used to forecast performance based on a set of operational conditions. In the recent past, ANN have been frequently employed by a number of researchers to represent complex and nonlinear BTM (battery thermal management) systems because of their general approximation abilities [1–6].

Using ANN with feed-forward (FF) architecture, Arora et al. [7] estimated the heat production for several battery modules under various operating situations. To predict the of the battery module, Afzal et al. [8] employed ANN where FF-network was used to increase the accuracy of the temperature prediction. Xie et al. [9] employed ANN with an FF architecture to forecast the battery module ohmic and polarization resistances as a function of SOC. Fang et al. [6] used an FF network to create an ANN to predict the surface temperature of a Ni–MH battery. To reduce the number of battery temperature sensors in the system, Kim et al. [10] employed an FF-net to anticipate the battery pack temperature. Farzad et al. [5] built an ANN model for battery pack temperature prediction using a BTMS. The model was trained, verified, and checked using a total of two thousand data points. The findings demonstrated that ANN can forecast battery temperature in a variety of BTMS operation circumstances. The battery surface temperature was predicted using an ANN model by Chen et al. [11]. The backpropagation model was used in their investigation with Levenberg–Marquardt as the learning algorithm. They found the network's R^2 nearly equal to 1 so suggested that the ANN can correctly anticipate the temperature of the battery surface. YuHeng et al. [12] used a recurrent ANN to estimate the temperature of Li-ion battery packs. Various operating parameters, comprising charge, discharge, and rest operations, were fed into their network at various operating conditions. They obtained R value close to one for their model and concluded that the RNN (recurrent NN) may be used to forecast Li-ion battery performance. The ANFIS (adaptive neuro-fuzzy inference system) based model, which is a combination of fuzzy inference system and ANN is another AI (artificial intelligence)-based model that has been used to predict Li-ion battery temperature [13]. The ambient temperature, current rate, and state of charge were used as inputs to this model. They found a good prediction of temperature using this model. Kleiner et al. [2], used a novel Nonlinear AutoRegressive with eXogenous (NARX)-network for the prediction of large battery packs. In terms of training and validation behaviour, the NARX is compared to a feedforward using the same general structure and input data. During a 10-h long-term forecast, both ANN techniques prove to be adequate for temperature prediction with an accuracy of within 1K. Park et al. [14] developed a supervised learning technique for optimising the TM (thermal management) system's performance in an EV (electric vehicle). Several ANNs are implemented and tested using features extracted during typical EV driving conditions for TM systems. The case study findings show that the ANN-based model accurately represents the TM system's working features, allowing for accurate battery temperature prediction.

Using ML (machine learning) algorithms, Warey et al. [15] predicted and analysed the impact of the AC (air conditioning) system on passenger thermal comfort. Tang et al. [16] developed an automated calibration model for the liquid-cooled BTMS to forecast cooling capacity and power utilization using support vector regression (SVR). The PSO (particle swarm optimization) method was used to optimize the hyperparameters of the SVR model in order to generate better results. The correlation coefficient of cooling load and power utilization for the envisaged PSO-SVR model is found to be improved when compared to the SVR model, implying that the PSO-SVR model can be utilised as a new technique to fit the complicated non-linear correlation between the power utilization, cooling capacity, and other influential variables of the BTMS system. Ruixin et al. [17] developed an extreme learning machine based thermal model (ELMT) to portray battery pack temperature behaviour under ESC (External short circuit) using the experimental results. They also compared this model to a multi-lumped-state thermal (MLT) model optimized by using the evolutionary algorithm to illustrate the efficiency of the proposed model. They found that the ELMT model has higher computational efficiency and better fitting and prediction accuracy than the MLT model. A spatiotemporal model entitled KL (Karhunen–Loève)-MLP (multilayer perceptron)-LSTM (long short-term memory) was created by Yajun et al. [18] for predicting the temperature distributions in a nonlinear distributed system of battery operation using a 3 step procedure. They found that the developed model was reliable for predicting temperature distributions during the thermal

process, as evidenced by the reasonable agreement between the current model results and those of another standard model. Wen et al. [19] developed a physics-driven machine learning (PD-ML) algorithm to predict temperature, stress, and deformation behaviour of Li-metal. They found that the PD-ML model, when provided with minimal test data, can accurately predict the mechanical characteristics of Li-metal over a wide range of temperature and distortion rate, and can aid in the creation of a Li-metal deformation map. The PD-ML modelling approach was also integrated with the FEM (finite element method) model. In analyzing temperature, stress, and mechanical behaviour of Li-metal, the coupling of PD-ML with FEM gives the advantages of FE (finite element) analysis and the precision of PD-ML. An electrochemical model integrated with a neural network (ETNN) was developed by Feng et al. [20] to predict the SOC (state of charge) and state of temperature (SOT). Based on rigorous testing, they found that this model can reliably predict battery voltage and core temperature at room temperatures with a 10-C rate. The multi-physics model can effectively predict temperature; however, it requires higher computational time and is hard to implement to the optimization processes. To address this problem, a DNN (deep NN) model trained using a small set of simulation data was applied to the BTMS by Park et al. [21]. They found that the DNN model produced an accurate estimate with a total error of less than 1%. By decreasing the computation time for the response of the multi-physics model, the DNN model provided a sufficient increase in system efficiency.

The discussed modelling of BTMS data using different algorithms indicates the importance of data prediction as the battery data depends upon a massive number of parameters. Thermal modelling is quite challenging as they are sensitive to change in operating factors. The variety of battery characteristics reported in the literature are highly sensitive and nonlinear which makes the predictions slightly complicated. Various algorithms such as SVR, ANN, LSTM, ANFIS, etc. are commonly reported in literature. These algorithms are sometimes combined to make a hybrid algorithm to model the battery characteristics. However, no detailed investigation on NN with single and deep layered networks to find an optimized model for a particular set of data is missing. Hence, the prime objective is to computationally analyze NN models which are optimal in predictions using different activation functions available. This study reports the use of single and deep layered NN models exploring four activation functions' potential. The NN model has experimented with different numbers of neurons and hidden layers to obtain an optimized model with fewer prediction errors. The rest of the work starts with Section 2 where the modelling of the battery system is explained briefly. Section 3 is provided with in detail discussion of use of ANN modelling of the BTMS data. The results of single-layered NN and deep NN modelling using different activation functions is provided in detail in Section 4. The work is concluded with insights of the work in Section 5.

2. Battery Modelling

An effective numerical model is created to get insights into the thermal features of Li-ion battery packs and cooling systems by incorporating the continuity of heat flux at the junction. The physical model employed for the coupled analysis of battery packs is depicted in Figure 1. A battery unit, in general, is made up of a layer of battery module (Figure 1a) whose temperature goes up as heat is produced during charging and discharging. As seen in Figure 1b, the prismatic battery module features a rectangular shape that transfers heat into the nearby coolant. It is worth noting here that the flow and thermal fields in the computational domain are also symmetric about the y-axis of the battery plate, in addition to geometric symmetry. As a result, just the left or right half of the physical domain can be used for computing in order to save the computation cost and time. So only 50% of the physical system is included for investigation (Figure 1b), with very few assumptions. According to the cell zone [22], heat produced is thought to be uniform. In the literature, steady heat analysis has been widely published, in which the heat produced while continuous charging and discharging for a longer duration is unchanged. The fluid

flow varies from 0.01 m/s to as high as 12 ms⁻¹ in a laminar flow coolant region. An air pump/fan is used to circulate the air flow, which requires more energy constantly. Thermal impacts are limited to a 2D model to reduce processing time and prevent needless complexity in modelling a practical 3D model. Further details of CPU-time, or memory requirements for this simulation required, can be referred to in [23,24].

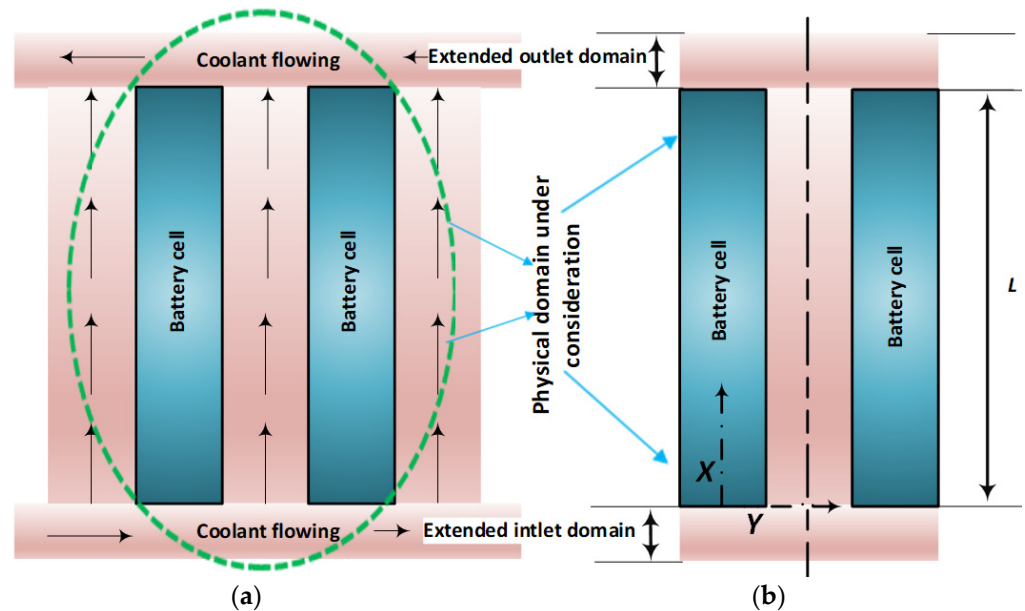


Figure 1. Proposed BTMS: (a) Battery cooling system; (b) Simplified model for numerical analysis.

The Finite volume method is used to evaluate the cooling ability of the proposed fluid-based battery cooling system. The governing equations describing the heat transport mechanism during discharging/charging a Li-ion battery pack.

The following extra approximations and assumptions are made in order to convert the physical model of the battery element provided above into a mathematical model:

- (i) The battery element's substance is homogeneous and isotropic.
- (ii) The battery element's conductivity is temperature independent.
- (iii) The temperature gradient in the x - y plane is insignificant.
- (iv) The flow is 2D, laminar, and incompressible.
- (v) The coolant is viscous and Newtonian.
- (vi) The coolant's thermophysical characteristics are constant.

Using these assumptions, the equations governing the steady, two-dimensional heat conduction in the battery element with non-uniform heat generation can be obtained as [22]

$$(k_r \nabla^2 T_r) + q_r^m = \rho c \frac{\partial T_r}{\partial t} \quad (1)$$

To solve the energy conservation of the fluid domain using a SIMPLE algorithm, the continuity and momentum equations in 2-D considered are given below by Equations (2)–(4) [25],

$$\nabla u_m = 0 \quad (2)$$

$$\frac{\partial u_m^*}{\partial t} (u_m^* \nabla u_m^*) = -\frac{1}{\rho} \nabla p + \mu \nabla^2 u_m^* \quad (3)$$

$$\rho c \frac{\partial T_m^*}{\partial t} + u_m \nabla T_m^* = \alpha \nabla^2 T_m^* \quad (4)$$

In the above Equations (1)–(4) in the vector form, the term r represents a battery, and m represents the coolant domain. T is for the temperature of fluid and battery, u is the

velocity in the x direction, ρ, c represents density and heat capacity of the fluid. α is the thermal diffusivity, and t is the time. The detailed non-dimensionalization, application of boundary conditions, grid independence study, validation, and in detail procedure have been reported in literature recently [22,25–29]. The input parameters that affect the conjugated thermal behaviour of the battery are shown in Figure 2, considered in this study for modelling the average Nusselt number.

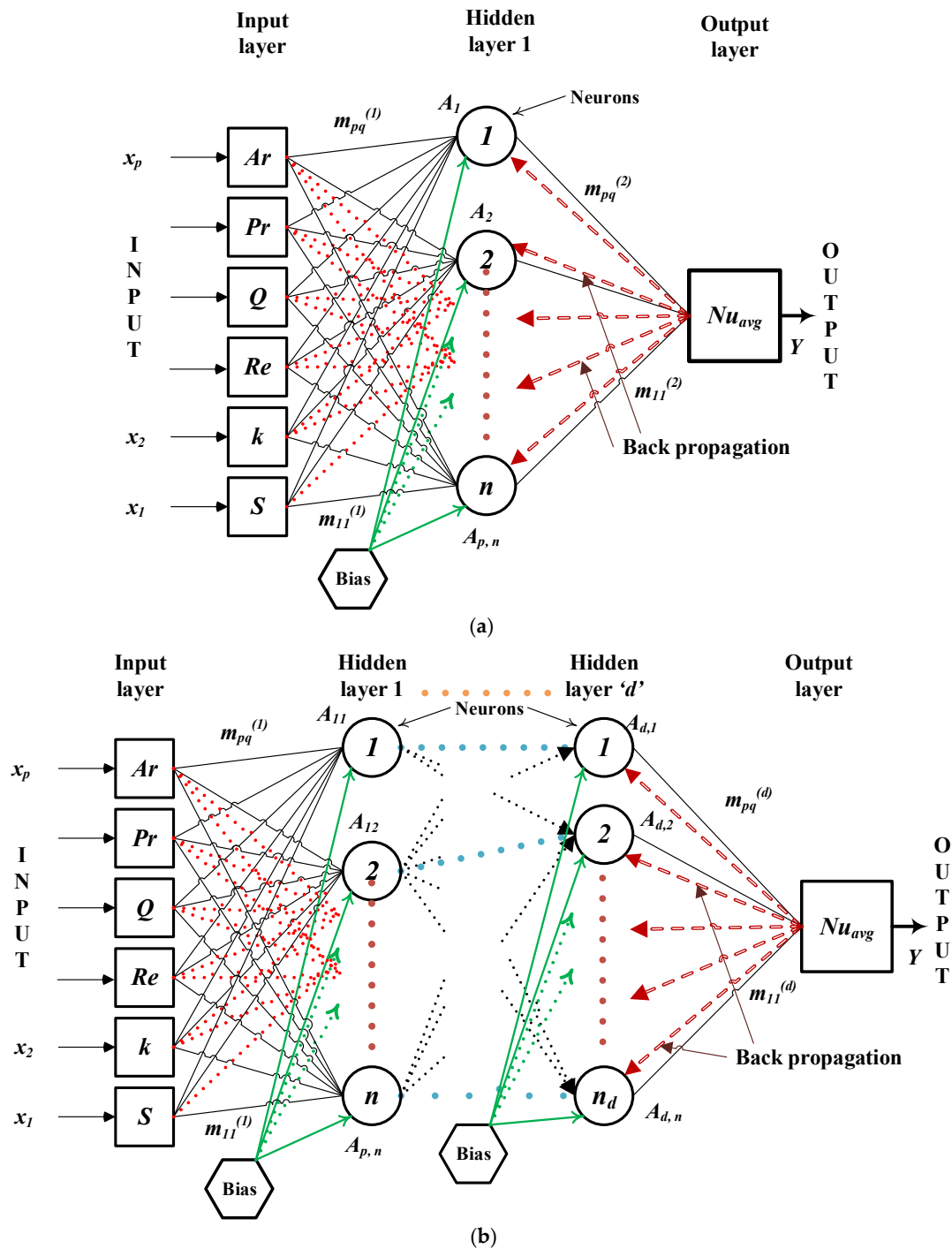


Figure 2. (a) Single hidden layer NN (NN₁) tested for the different number of neurons (b) Deep hidden layer NN (NN_D) tested for the different number of neurons and deep hidden layers ‘d’.

3. Artificial Neural Network (NN) Modelling

Figure 2a depicts the proposed NN model with a single hidden layer used in this study to predict the battery pack's temperature with the number of neurons experimented with from 1 to n . The six inputs to the NN and one output from it are also clearly shown. At first, a single layer model (NN_1) is used, as shown in Figure 2a. In this NN_1 model, the number of neurons is tested from 1 to 10. Then, as shown in Figure 2b the NN is analyzed for multiple deep hidden layers (NN_D) where the neurons in each layer are also varied simultaneously, and hidden layers are also varied from 3 to 10.

A hidden layer has a particular number of nodes in addition to the input and output layers. Each neuron's inputs are weighted according to its edge weight and the weighted inputs are added together. Each neuron also has a bias added to it. An activation function ' f ' maps the whole sum of weighted inputs and the bias non-linearly to the neuron output. In this study, the activation function employed in the hidden layer is the sigmoid function: tanh, Gaussian, and linear function [30]; however, the sigmoid function produced the optimum results for this application. A simple linear function is used in the output layer to modify the weights: direction close to the error minimum. The Levenberg–Marquardt (LM) Algorithm [31], which is a mixture of gradient descent and the Newton technique, is used in the present distribution as this method is the one that converges the fastest.

An artificial neural network model, as depicted in Figure 2, is made up of nodes/neurons that are organised into many layers: an input layer, one or more hidden layers, and an output layer. Each neuron has an activation function that calculates how much stimulation is applied to each neuron. The input variables are transformed at each layer by collecting neurons, which are then disseminated to its next layer, defined by Equations (5)–(7).

$$A_j^n = \sum (m_{pq}^{(1)} x^{n-1} p + m_{q0}^{(1)}) \quad (5)$$

$$b_1^n = \sum (m_{1q}^{(2)} A^{n-1} p + m_{10}^{(2)}) \quad (6)$$

$$y_1^n = F(b_1^n) \quad (7)$$

where x is the 1st layer's input; A is the 1st layer's output; p and q are the NN node indexes; n is the layer's index; $m_{pq}^{(l)}$ (l) is the weight between the q th node in the 1st layer and the p th node in the input layer; b is the output layer's input; $F(b_p^n)$ is the output value of p th node in $(n + 1)$ th layer after activation.

The weight and bias in between the synapses, which evaluate the relevance of the data transmitted across the link, are represented by m and m_0 , respectively. $F(b)$ is the activation function, which uses the combined output of the hidden layer to produce the output y .

The starting weights and biases are allocated at random, and the training cycle repeats till the required result is achieved, as determined by Equation (8) of the cost function.

$$E(m) = \frac{1}{2} \sum_{p=1}^p y(x_p, m_i) - o_p^2 \quad (8)$$

where o is the intended output, $E(m)$ denotes the cost function utilized to assess the training operation, m denotes the weight, and i denotes the cost function calculation index.

The weight and bias of the NN model are modified throughout the training procedure, as shown in Equation (9), to reduce the error.

$$m^n = m^{n-1} - (J^T J + \mu I)^{-1} J e^{n-1} \quad (9)$$

where $J = \partial E / \partial m$ is the Jacobian matrix connected to m , I is the identity matrix, μ is the combining coefficient, and e is the prediction error.

The forward calculation via Equation (10) is the 1st step in the LM method. Equation (10) is used to determine the output and hidden layer forecast errors.

$$\begin{aligned} e_1^{(3)} &= y_1 - 0 \\ \delta_1^{(3)} &= e_1^{(3)} \\ \delta_q^{(2)} &= m_{1q} \delta_1^{(3)} \end{aligned} \quad (10)$$

The Jacobian is determined using a backpropagation technique, as indicated by the following Equation (11).

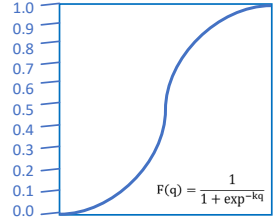
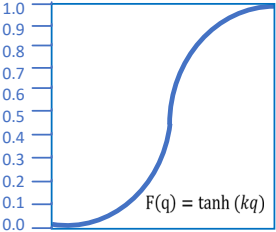
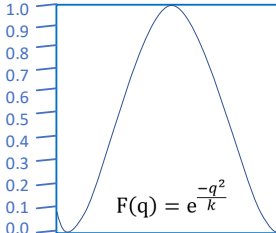
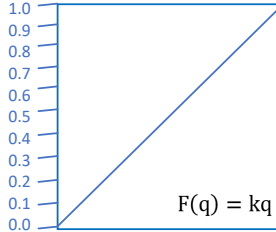
$$\begin{aligned} \frac{\partial E}{\partial m_{pq}} &= \delta_q^{(2)} x_p \\ \frac{\partial E}{\partial w_{m_{1j}}} &= \delta_1^{(3)} A_{1j} \end{aligned} \quad (11)$$

The deep learning model is a NN model having deep hidden layers, as shown in Figure 2b. As explained earlier, the fundamental foundation is the same as the NN model, which is based on the brain's operation and biological architecture, allowing machines to reach human-like intelligence. The most basic kind of NN_D is hierarchical synapses that send information to adjacent neurons based on input, building a complicated system that learns from feedback. The input data is supplied into the 1st layer's non-hidden synapses, and the outcome of this layer is fed as an input for the next phase, and so on, till the outcome is attained. The outcome is expressed as a probability, resulting in a forecast of either "Yes" or "No". Each layer's synapses calculate a small function called an "activation function" that aids in transmitting the relevant neurons. Weights link the synapses of 2 sequential layers together. These weights govern the feature significance in forecasting the targeted value. The NN_D with multiple hidden layers is shown in Figure 2b. The weights are initially random, but adjusted iteratively as the model is trained to learn and anticipate the outcome.

As shown in Figure 2, six operating parameters: flow velocity, coolant type, heat generation, thermal conductivity, battery length and width ratio, and space between the parallel battery packs were set as input parameters of the neural network models. The output parameter of the two neural network models (NN₁ and NN_D) was set to the average Nusselt number. The training data is derived entirely from numerical experiments. For the training procedure, a total of 1750 sets of numerical data were gathered. Each model has at least 3 layers and is designed to have some activation function during the training phase. A hidden layer follows each input layer, which is comprised of input variables. Each of the input layers has an effect on the output parameter prediction. The number of nodes in the input layer indicates how many input variables were used in the model. The hidden levels feed into another aggregation layer, which mixes the findings from the preceding layers and feeds them to the output layer, resulting in a continuous result. Several nodes/neurons in the hidden layer were tested to attain the desired regression efficiency to reduce overfitting and underfitting.

A total of 85% of sets of numerical data were used for training and the remaining 15% of sets of numerical data were used for testing. The neural network regression training process was given 1000 epochs. For a more accurate prediction of the output, the root mean square error (RMSE) must be as low as possible. In Table 1, the four activation functions used for the NN₁ and NN_D models are provided.

Table 1. Activation functions used in this work.

Activation Function	Equation	Graph
Sigmoid	$F(q) = \frac{1}{1 + \exp^{-kq}}$	
Tan hyperbolic (Tanh)	$F(q) = \tanh(kq)$	
Gaussian	$F(q) = e^{-\frac{q^2}{k}}$	
Linear	$F(q) = kq$	

4. Results and Discussion

The modelling of Nu_{avg} , which indicates the heat transfer from the conjugated battery (generating heat) surface, is shown and discussed in detail. A neural network (NN) with a single hidden layer (NN_1) is analyzed thoroughly initially. The RMSE and R-squared are investigated for this NN_1 to get an optimized model that predicts the battery Nu_{avg} by experimenting with several neurons in the hidden layer having different activation functions. Later NN_D (deep NN) model is also analyzed with different hidden layers and neurons in each layer to obtain the optimized network for different activation functions. As reported in a few works, the Nu_{avg} data pertinent to battery thermal management is highly nonlinear [8,22,25,28,32,33]. The entire data is sorted carefully and compiled to enable the intelligent algorithms to predict an essential aspect of battery systems. The optimized model is obtained in this work when the RMSE and R-squared values are at their best. These values are taken from the average of 5 computational cycles for each output, which counts for massive calculations and data generations. However, selected data is presented in this section to avoid too much length in this article.

4.1. NN_1 Modelling of Nu_{avg}

At first, the convergence of RMSE with increasing iterations is demonstrated for the NN_1 model during training and testing for different activation functions. In Figure 3 the

epochs which represent each iteration of data passed for computation by the algorithm is set equal to 1000 that leads to converged RMSE for all cases, with the NN₁ model having six neurons in the hidden layer are shown. Compared to training sessions, the testing RMSE is converged at higher values. Except for the linear function, the remaining three functions converge more or less at the same RMSE while the linear function gives a higher RMSE value. The Linear testing functions have converged at higher values compared to the rest of all scenarios. Between Sigmoidal and Gaussian, the Sigmoidal function did not provide much better testing convergence than the Gaussian function. However, the difference between RMSE of the testing of these two is not very high.

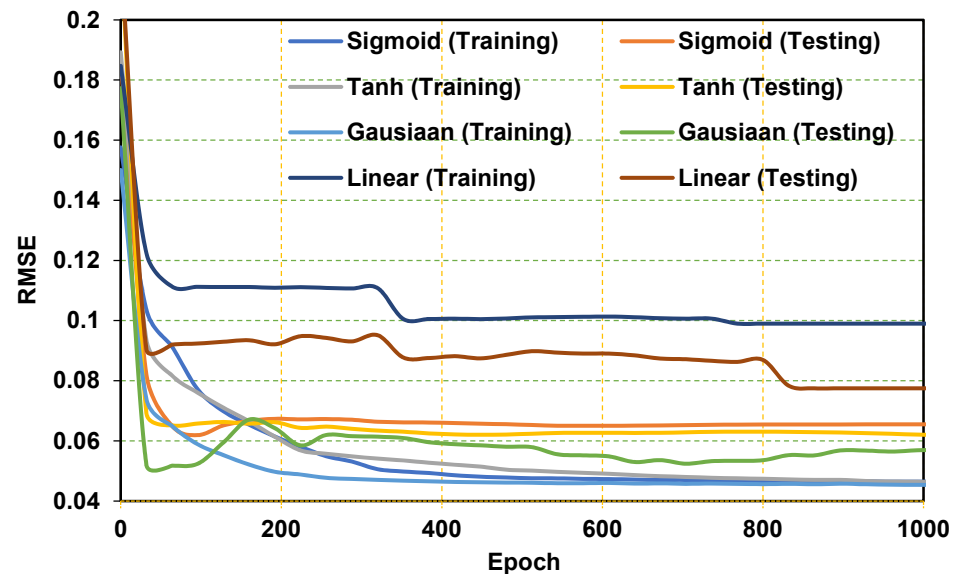


Figure 3. Convergence of RMSE after 1000 epochs having six neurons in the hidden layer.

In Figure 4, the RMSE of the trained and tested NN₁ model after completing more than 1000 epochs with increasing neurons in the hidden layer is shown. All the activations are employed to get the optimized model. From Figure 4a it is seen that the RMSE for Sigmoidal function reduces when neurons are increased from 1 to 6 in the hidden layer. From 6 neurons onwards, the RMSE for Sigmoidal function shows slight increments. For Tanh function, also similar trend can be noted. The Gaussian function has also given a comparatively less RMSE at six neurons in the hidden layer; The linear function has shown a higher and consistent error at all neurons. Hence, with six neurons in the hidden layer, the model training is optimized. Very similar behaviour is also found for the tested model (Figure 4b).

The rest of the computations are provided using the above optimized NN₁ model with 6 neurons in the hidden layer. The Sigmoidal activation function in the hidden layer is trained for the battery Nu_{avg} data, whose training and predictions are depicted in Figure 5. The network predictions during both sessions show that the values are close to the actual (true) values as the data is denser towards the centre trendline. A slight outlier in the testing region can be noted because the R^2 and RMSE are slightly more significant than the training provided values. As the R^2 value is above 0.9, the modelling can be successful and capable of predicting well. The RSME value for training is 0.029, better than the RMSE of testing (0.04).

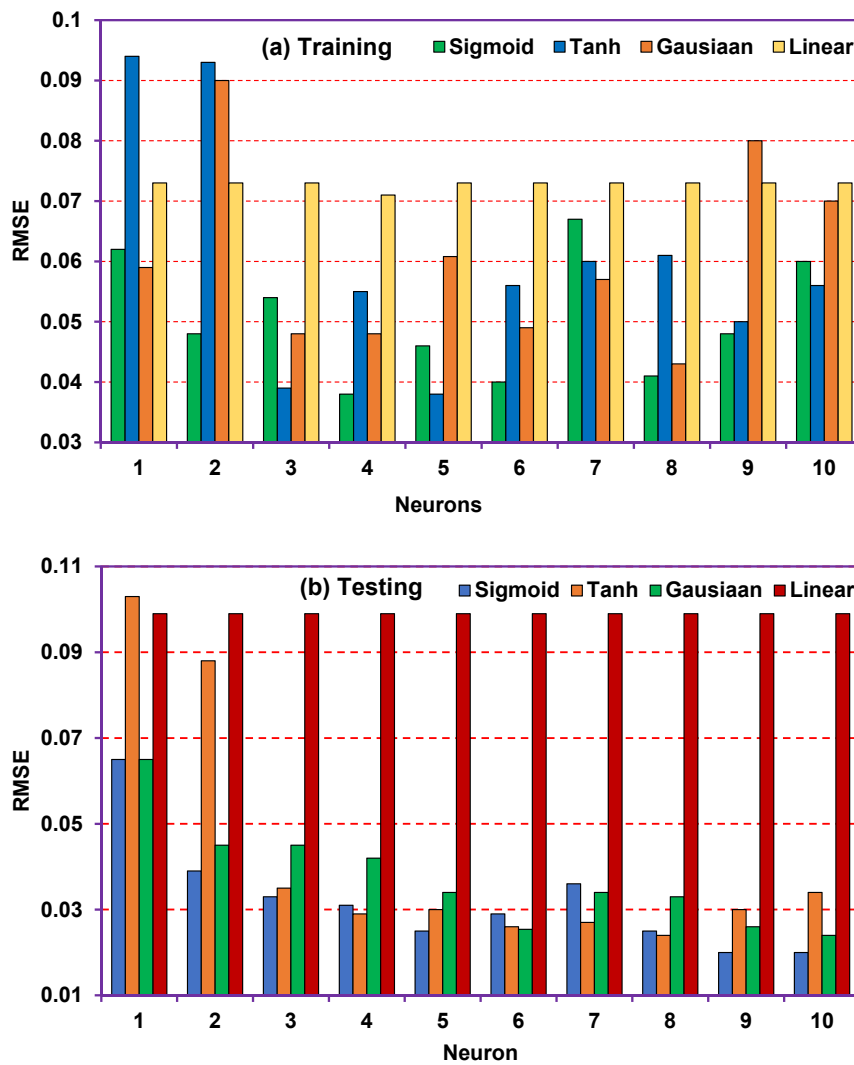


Figure 4. RMSE was obtained for neurons in a single hidden layer varying from 1 to 10 using four activation functions during (a) training (b) testing.

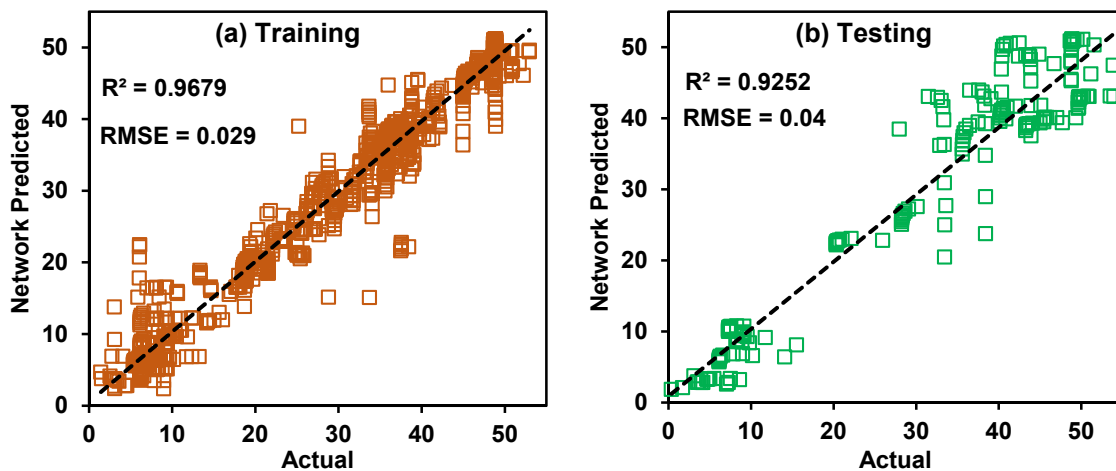


Figure 5. (a) Training (b) testing of NN_1 using Sigmoidal function indicating the R^2 and RMSE for the optimized model.

The optimized NN_1 model with six neurons using tanh as the activation function is used for training the data. It is noted from Figure 6a,b that the network calculated battery output data is in line with the actual battery values. The highly sensitive Nu_{avg}

has significant outliers, as indicated in Figure 6b, slightly underfitting with the NN_1 network. The trained computations are accurate as the R^2 and RMSE are above 0.9 and close to zero, respectively. However, during testing, this model has slightly under-predicted the data because the orientation towards the centerline is not seen. Using the Gaussian function for NN_1 model calculations has shown some improved predictions than the Tanh function calculations. Figure 7a,b provides an accessible analysis of the network predictions compared to the actual battery values. The 45° line drawn for the mapping of both the data indicates that the NN_1 model with Gaussian function is excellent in battery Nu_{avg} data. The training is in line with the Tanh function, but the testing is far better than the Tanh function as the R^2 is above 0.9 during both sessions. In Figure 8, the NN_1 model calculations using the Linear activation function are depicted. Both Figures 8a and 9b clearly show that the Linear function cannot help the neural model to depict the sensitive data of the battery system based on thermal parameters. Figure 8a shows how the Nu_{avg} values are out of bounds in the training session leading to R^2 and RMSE to just 0.68 and 0.099, respectively, which is unacceptable. The predictions in testing also have high errors, as shown in Figure 8b as the R^2 and RMSE values are nearly 0.8 and 0.073 which are very high compared to the other three activation functions explained earlier.

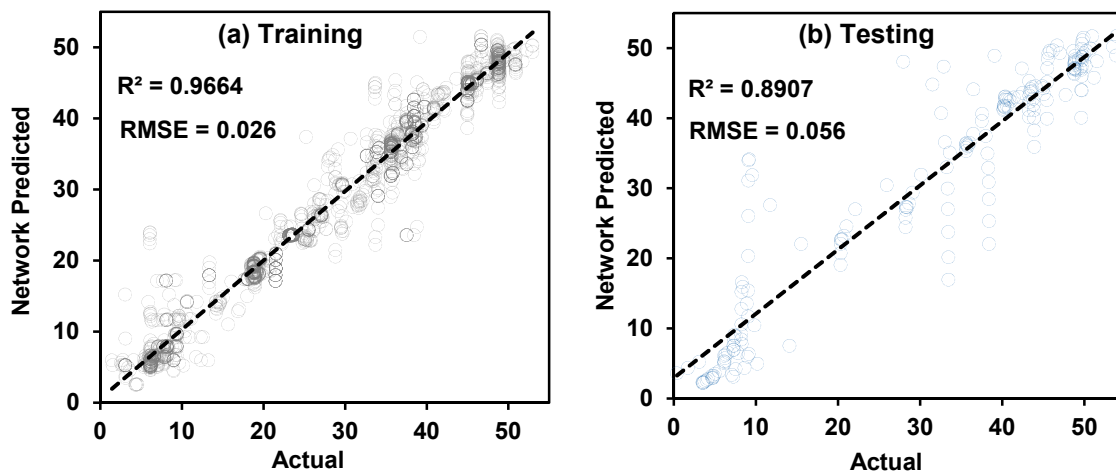


Figure 6. Network calculated battery data using Tanh as the activation function during (a) training and (b) testing of the NN_1 model.

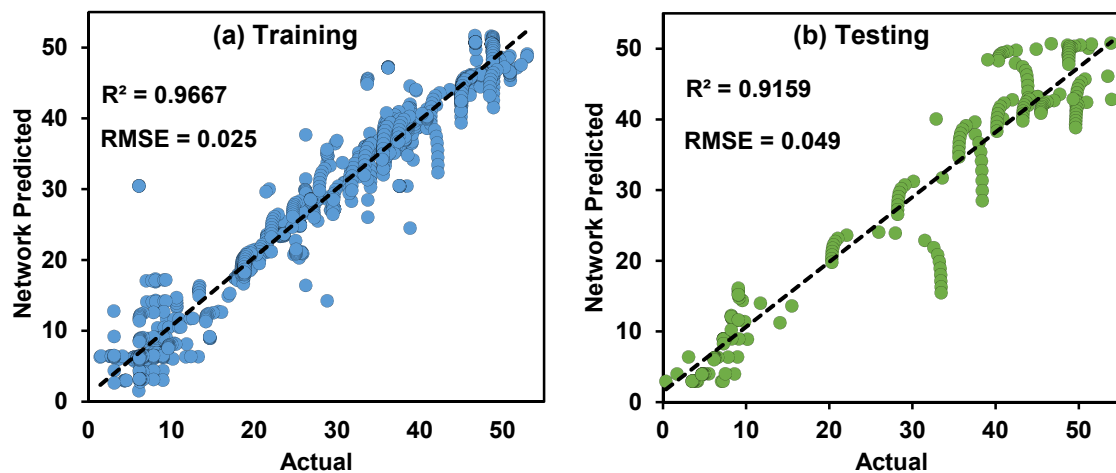


Figure 7. Gaussian function operated NN_1 model in (a) training and (b) testing.

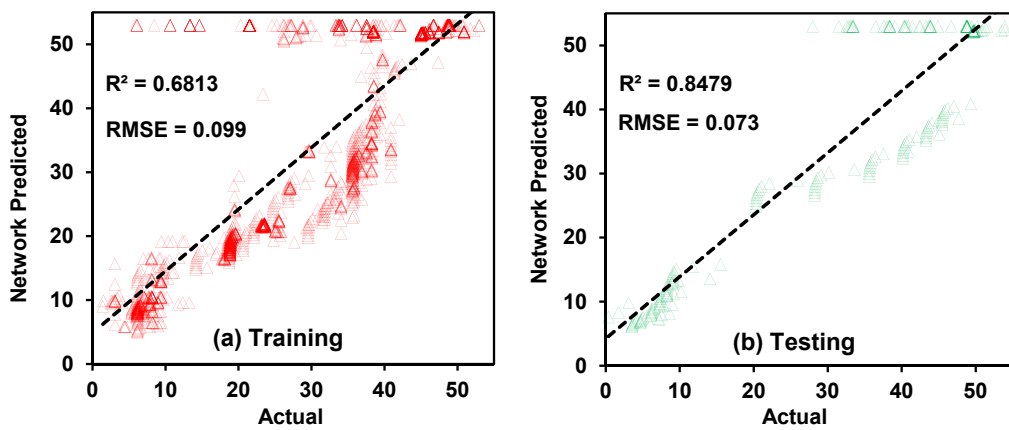


Figure 8. Linear activation function used to calculate the battery output data for the NN₁ model in (a) training and (b) testing.

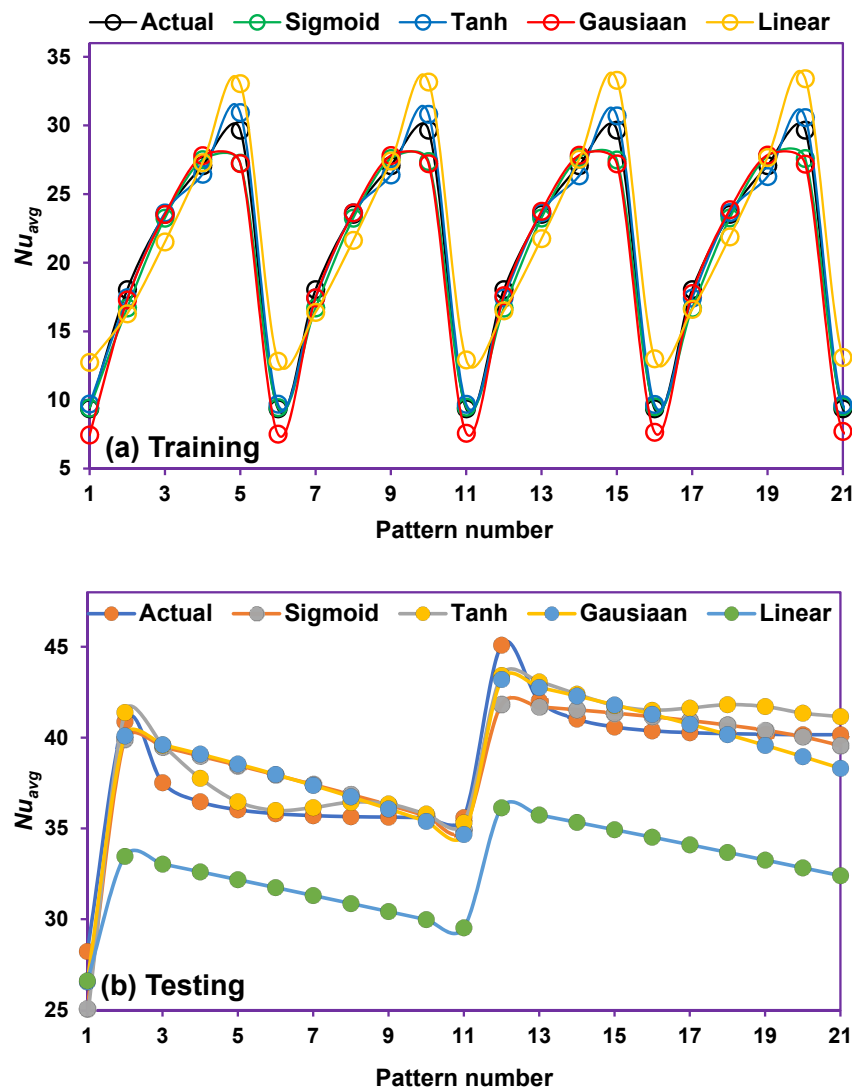


Figure 9. The trend of battery Nu_{avg} data for the selected pattern (data points) was obtained from (a) training and (b) testing using four activation functions.

The overall trend of actual battery Nu_{avg} data from four activation functions applied to the optimized NN₁ model is given in Figure 9a,b for training and predictions, respec-

tively. The actual trend in the black line is almost overlapped with the Sigmoidal and Gaussian function NN_1 model. The Tanh function has some difference from the other two, followed by the Linear function, which has shown a more significant difference than others. Figure 9b illustrates the predictions made by the trained NN_1 model where the actual values and Sigmoidal and Gaussian are very close to the Tanh and Linear function data points.

4.2. NN_D Modelling of Nu_{avg}

The RMSE variations for different deep layers varying from 3 to 10 in the back-propagating NN model are shown in Figure 10a,b. The neurons in each deep layer are simultaneously varied from 4 to 10 for each activation function. In brief, analyses of RMSE from NN_D applying the Sigmoidal activation function is illustrated for the shortness of the article. R^2 values are also avoided for the same. However, optimized values of R^2 are directly shown in the optimized model (in the graphs) achieved after experimentations. From Figure 10a,b, the RMSE from the trained model shows that the six neurons in each six-layered deep NN model are the best from the entire set of computations. This is also confirmed with the R^2 values which are added in the forthcoming graphs as they are the optimal value. Though nine neurons have shown some promising results, it is not considered the best because its value is too fluctuating for different hidden layers, and the computational cost is also involved. From the similar set of experiments for Tanh, Gaussian, and Linear functions, it is concluded that 8-neurons in 4-deep layers each, 6-neurons in 5-deep layers each, and 7-neurons in 3-deep layers each respectively provided the best RMSE and R^2 .

The application of the Sigmoidal function for deep NN (NN_D) to train and predict the Nu_{avg} data is shown in Figure 11. The least (optimized) R^2 and RMSE value was obtained for NN with six deep hidden layers having six neurons each. The respective R^2 and RMSE are 0.97 and 0.018 obtained from the trained model and 0.93 and 0.045 for the tested model. This deep model has proved to be much more efficient than the earlier model accuracy as the error is less and the mapping with actual value is much more closer. The training is more accurate while the tested model gives slightly more error, which is highly acceptable but may concern the difference in terms of percentage. This is again tested for other functions, which shows much less difference in the error between the training and testing model. Figure 12a,b shows the deep network predictions during the training/testing sessions using Tanh as the activation functions. The optimized error was obtained for this function applied to NN model having four hidden layers with eight neurons each. The R^2 and RMSE are 0.98 and 0.013 from this optimized model, which matches the actual values excellently. The training R^2 and RMSE are 0.91 and 0.048 during the testing, which is equally good as Sigmoidal function operated NN_D .

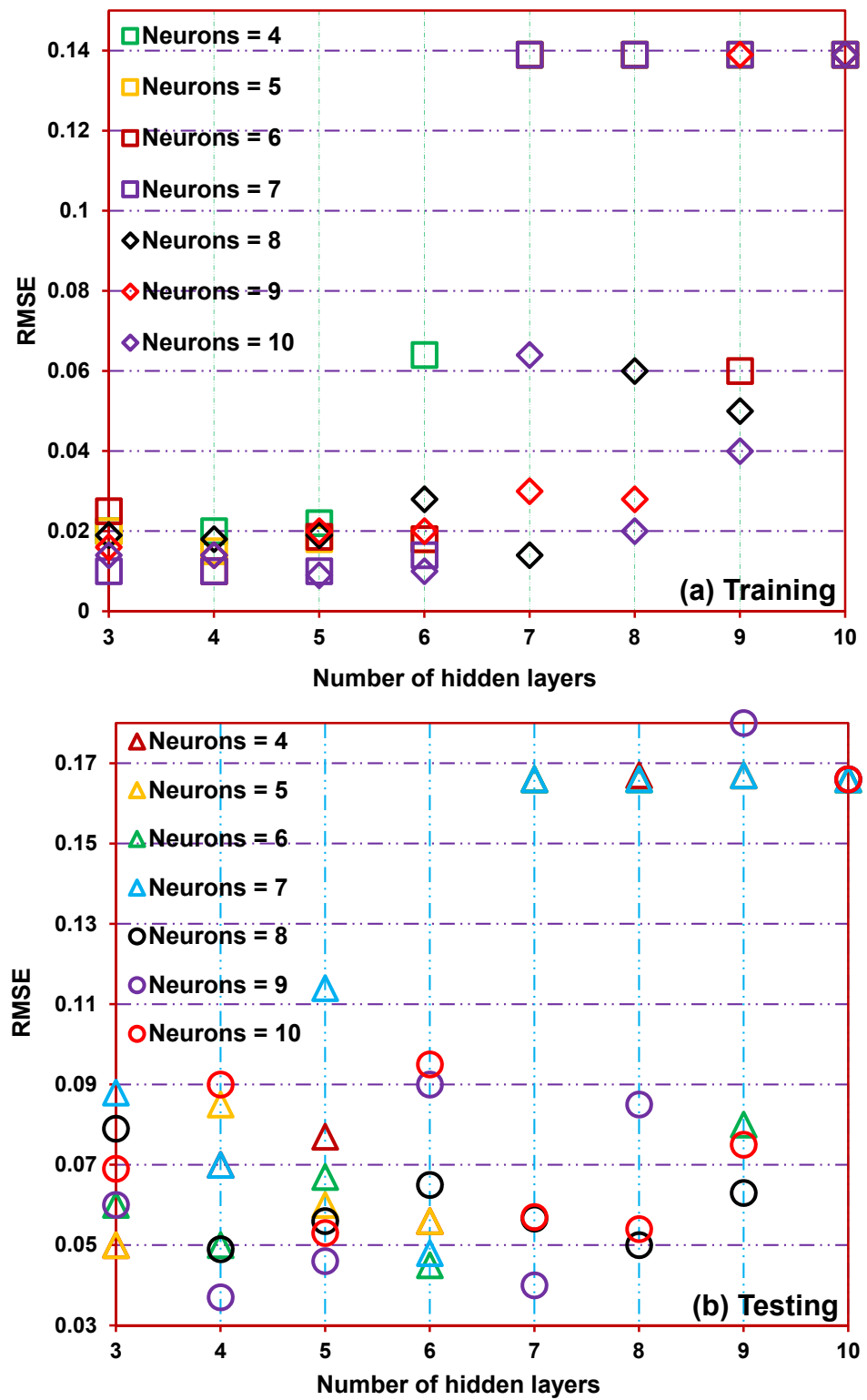


Figure 10. RMSE for the different number of neurons in each hidden layer. The number of hidden layers in deep NN are analyzed from three to 10 to get the optimized model during (a) training and (b) testing.

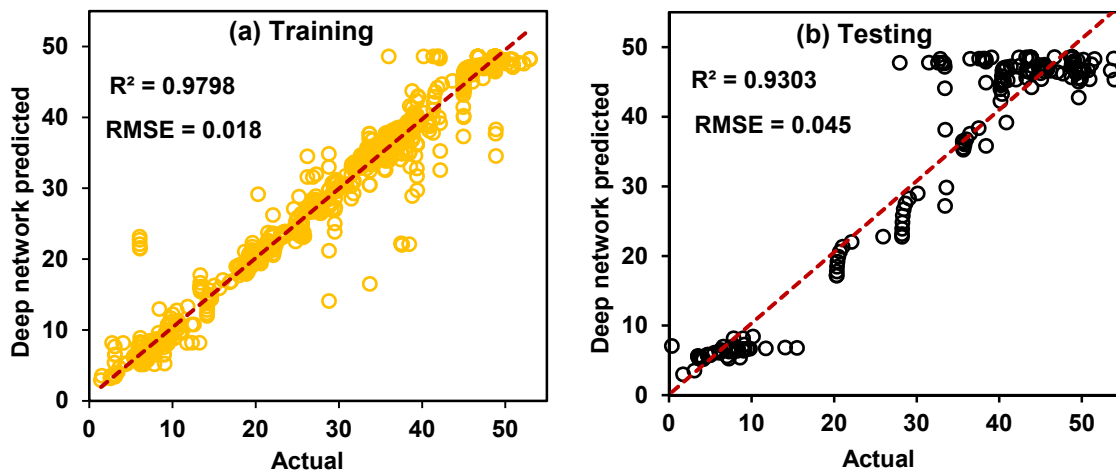


Figure 11. Actual vs. NN_D (deep NN) was obtained using the Sigmoidal function, having six hidden layers with six neurons each during (a) training and (b) testing.

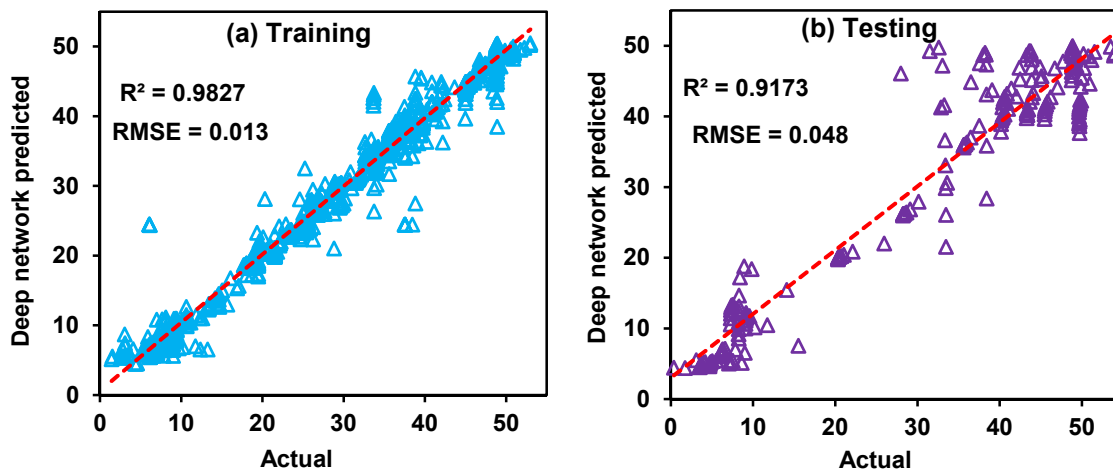


Figure 12. Tanh function applied deep NN having four hidden layers with eight neurons each as the optimized model for Nu_{avg} prediction during (a) training and (b) testing.

The Gaussian function operated deep NN model is optimal with six neurons each in 5 hidden layers. The R^2 and RMSE are 0.97 and 0.023 from the output of trained model numerals, while in the testing R^2 and RMSE are 0.94 and 0.036, as shown in Figure 13. This has indicated that the Gaussian function is far better than the previous two models. Another interesting point to note is the less difference obtained in the trained and tested R^2 and RMSE value which was not obtained from many previous regressions. Next, unexpectedly, the Linear function, which was not accepted in the single NN model earlier, has provided better results than the Sigmoidal and Tanh function. Figure 14 depicts the same for the Linear function having near R^2 and RMSE values as Gaussian function.

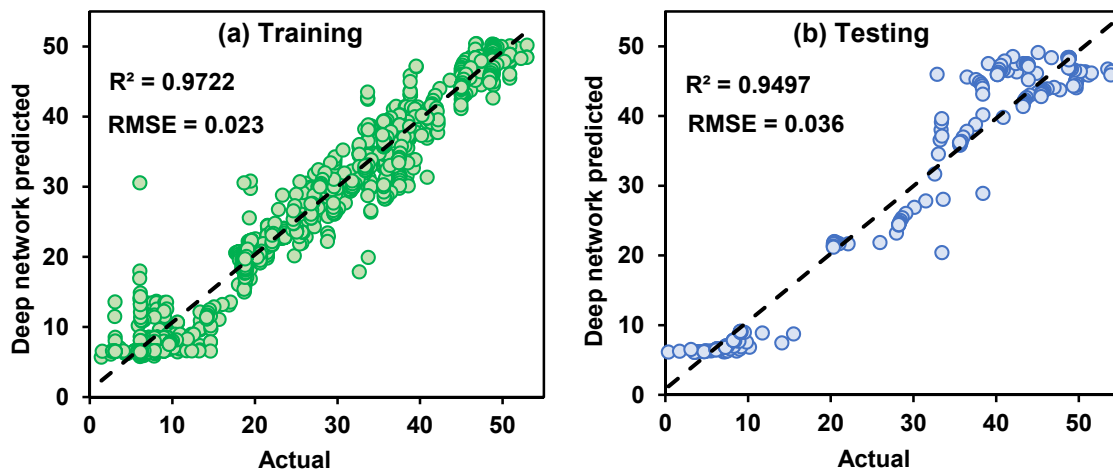


Figure 13. Deep NN optimized model with six neurons each in 5 hidden layers using Gaussian function in each neuron during (a) training and (b) testing.

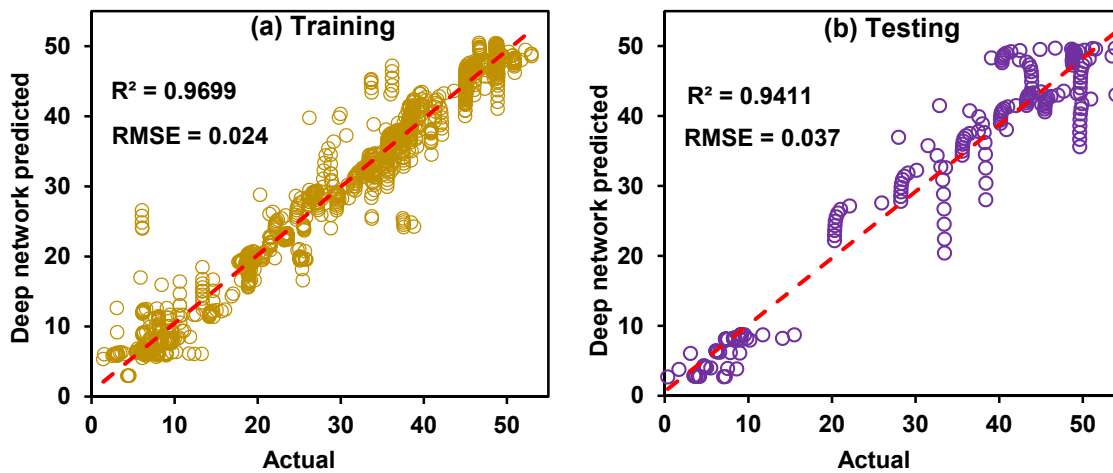


Figure 14. R^2 and RMSE value of optimized deep NN model using Linear activation function at seven neurons each in 3 hidden layers during (a) training and (b) testing.

In Figure 15, a combined trend of a deep NN model optimized with their neurons in hidden layers as mentioned earlier and the actual values are shown. The closeness between actual values and different activation functions is indicated for training and testing. The empty circles represent the trained values, and empty squares are for tested values. The scattered data points indicate that the actual Nu_{avg} and predictions from the optimized models which are close enough for the forecasting. Though all are acceptable, except Tanh function, the rest have provided very close predictions and are highly useful in this enormously sensitive data of battery thermal system.

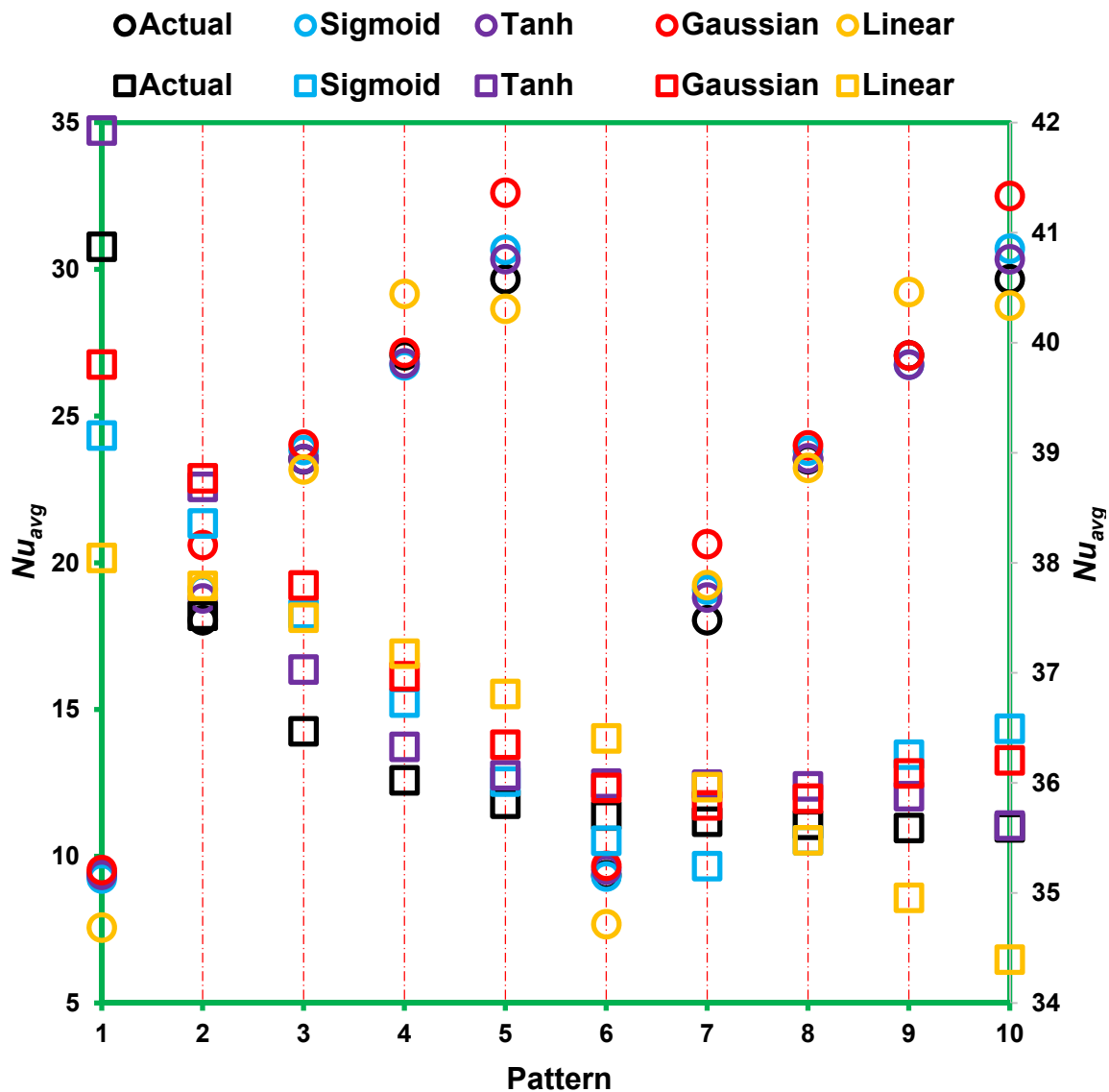


Figure 15. Training (circles) and testing (squares) made by the deep NN models optimized from four activations functions compared to actual battery data.

5. Conclusions

Battery thermal management is crucial in the high capacity and longer life of Li-ion modules; careful modelling of heat transfer phenomena is required. An attempt was made to model the highly sensitive battery data to predict the heat transfer character, namely, average Nusselt number (Nu_{avg}) depending on six operating parameters, including the battery heat generation, coolant properties, and battery spacing. A single layer and deep layer NN model with different activation functions was employed. A massive computational experiment (around 800 cycles of computations) was performed to obtain the optimized NN models for each activation function. The optimized NN model was obtained with 6 neurons in the hidden layer for all activation functions in a single layer. At the same time, the optimized model was in a deep NN model with 6 hidden layers having 6 neurons each for Sigmoidal. For Tanh, 8-neurons in 4-deep layers each, for Gaussian; 6-neurons in 5-deep layers each, and for Linear function; 7-neurons in 3-deep layers each was the optimized model. The RMSE and R^2 were accessed for all the models during the training and testing of the networks. In a single NN model, the Sigmoidal and Gaussian function outperformed the Tanh and Linear functions. Linear function, however, failed to predict the battery data sufficiently. In the deep NN model, the gaussian and Linear

functions outperformed the other two functions operated NN. Overall, deep NN provided a much better prediction than the single-layer NN model.

Author Contributions: Conceptualization, A.A. (Asif Afzal); Data curation, A.A. (Abdulrahman Alrobaian) and S.A.K.; Formal analysis, A.A. (Asif Afzal) and A.R.K.; Funding acquisition, S.A.K.; Investigation, J.K.B., A.A. (Abdulrahman Alrobaian) and S.A.K.; Methodology, A.A. (Asif Afzal) and A.A. (Abdulrahman Alrobaian); Resources, J.K.B. and A.R.K.; Validation, A.R.K.; Writing—original draft, A.A. (Asif Afzal) and A.R.K.; Writing—review and editing, J.K.B. All authors have read and agreed to the published version of the manuscript.

Funding: Deanship of Scientific Research at King Khalid University grant no. R.G.P. 1/299/42.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University, Saudi Arabia for funding this work through the Research Group Program under grant no. R.G.P. 1/299/42.

Conflicts of Interest: The authors declare no conflict of interest.

Nomenclature

Ar	Aspect ratio of a battery cell	Greek symbols	
c	Specific heat in J/kgK	α	Thermal diffusivity of fluid m^2/s
k_r	Thermal conductivity of battery in W/MK	ν	Kinematic viscosity of fluid m^2/s
Nu_{avg}	Average Nusselt number	ρ	Density of fluid kg/m^3
q_r^m	Volumetric heat generation W/m^3	Acronyms	
Q	Dimensionless volumetric heat generation	BTMS	battery thermal management system
Pr	Prandtl number	NN	Neural network
Re	Reynolds number	NN ₁	Single layered neural network
T_r	Temperature of battery in K	NN _D	Deep neural network
T_m	Temperature of fluid in K	SIMPLE	Semi-implicit pressure linked equation
t	Time in seconds	LM	Levenberg–Marquardt
u_m	Velocity along the axial direction m/s	AI	Artificial intelligence
p	Pressure in N/m^2	ML	Machine learning
x	Axial direction	ANN	Artificial Neural network
y	Transverse direction	Li-ion	Lithium-ion
		R ²	R-squared
		RMSE	Root mean square error

References

- Kolodziejczyk, F.; Mortazavi, B.; Rabczuk, T.; Zhuang, X. Machine learning assisted multiscale modeling of composite phase change materials for Li-ion batteries' thermal management. *Int. J. Heat Mass Transf.* **2021**, *172*, 121199. [[CrossRef](#)]
- Kleiner, J.; Stuckenberger, M.; Komsijska, L.; Endisch, C. Real-time core temperature prediction of prismatic automotive lithium-ion battery cells based on artificial neural networks. *J. Energy Storage* **2021**, *39*, 102588. [[CrossRef](#)]
- Benmouna, A.; Becherif, M.; Boulon, L.; Dépature, C.; Ramadan, H.S. Efficient experimental energy management operating for FC/battery/SC vehicles via hybrid Artificial Neural Networks-Passivity Based Control. *Renew. Energy* **2021**, *178*, 1291–1302. [[CrossRef](#)]
- Yetik, O.; Karakoc, T.H. Estimation of thermal effect of different busbars materials on prismatic Li-ion batteries based on artificial neural networks. *J. Energy Storage* **2021**, *38*, 102543. [[CrossRef](#)]
- Jaliliantabar, F.; Mamat, R.; Kumarasamy, S. Prediction of lithium-ion battery temperature in different operating conditions equipped with passive battery thermal management system by artificial neural networks. *Mater. Today Proc.* **2021**. [[CrossRef](#)]
- Fang, K.; Mu, D.; Chen, S.; Wu, B.; Wu, F. A prediction model based on artificial neural network for surface temperature simulation of nickel–metal hydride battery during charging. *J. Power Sources* **2012**, *208*, 378–382. [[CrossRef](#)]

7. Arora, S.; Shen, W.; Kapoor, A. Neural network based computational model for estimation of heat generation in LiFePO₄ pouch cells of different nominal capacities. *Comput. Chem. Eng.* **2017**, *101*, 81–94. [[CrossRef](#)]
8. Mokashi, I.; Afzal, A.; Khan, S.A.; Abdullah, N.A.; Bin Azami, M.H.; Jilte, R.D.; Samuel, O.D. Nusselt number analysis from a battery pack cooled by different fluids and multiple back-propagation modelling using feed-forward networks. *Int. J. Therm. Sci.* **2021**, *161*, 106738. [[CrossRef](#)]
9. Xie, Y.; He, X.; Hu, X.; Li, W.; Zhang, Y.; Liu, B.; Sun, Y. An improved resistance-based thermal model for a pouch lithium-ion battery considering heat generation of posts. *Appl. Therm. Eng.* **2020**, *164*, 114455. [[CrossRef](#)]
10. Kim, T.J.; Youn, B.D.; Kim, H.J. Battery Pack Temperature Estimation Model for EVs and Its Semi-transient Case Study. *Chem. Eng. Trans.* **2013**, *33*, 955–960.
11. Chen, S.; Fang, K.-Z.; Mu, D.-B.; Wu, B.-R. Application of neural network model to predicting surface temperature of lithium-ion battery. *Trans. Beijing Inst. Technol.* **2013**, *33*, 421–424.
12. Jiang, Y.; Yu, Y.; Huang, J.; Cai, W.; Marco, J. Li-ion battery temperature estimation based on recurrent neural networks. *Sci. China Technol. Sci.* **2021**, *64*, 1335–1344. [[CrossRef](#)]
13. Fan, B.; Lin, C.; Wang, F.; Liu, S.; Liu, L.; Xu, S. An Adaptive Neuro-Fuzzy Inference System (ANFIS) Based Model for the Temperature Prediction of Lithium-Ion Power Batteries. *SAE Int. J. Passeng. Cars-Electron. Electr. Syst.* **2018**, *12*, 5–11.
14. Park, J.; Kim, Y. Supervised-learning-based optimal thermal management in an electric vehicle. *IEEE Access* **2019**, *8*, 1290–1302.
15. Warey, A.; Kaushik, S.; Khalighi, B.; Cruse, M.; Venkatesan, G. Data-driven prediction of vehicle cabin thermal comfort: Using machine learning and high-fidelity simulation results. *Int. J. Heat Mass Transf.* **2020**, *148*, 119083.
16. Tang, X.; Guo, Q.; Li, M.; Wei, C.; Pan, Z.; Wang, Y. Performance analysis on liquid-cooled battery thermal management for electric vehicles based on machine learning. *J. Power Sources* **2021**, *494*, 229727. [[CrossRef](#)]
17. Yang, R.; Xiong, R.; Shen, W.; Lin, X. Extreme learning machine-based thermal model for lithium-ion batteries of electric vehicles under external short circuit. *Engineering* **2021**, *7*, 395–405.
18. Fan, Y.; Xu, K.; Wu, H.; Zheng, Y.; Tao, B. Spatiotemporal modeling for nonlinear distributed thermal processes based on KL decomposition, MLP and LSTM network. *IEEE Access* **2020**, *8*, 25111–25121.
19. Wen, J.; Zou, Q.; Wei, Y. Physics-driven machine learning model on temperature and time-dependent deformation in lithium metal and its finite element implementation. *J. Mech. Phys. Solids* **2021**, *153*, 104481. [[CrossRef](#)]
20. Feng, F.; Teng, S.; Liu, K.; Xie, J.; Xie, Y.; Liu, B.; Li, K. Co-estimation of lithium-ion battery state of charge and state of temperature based on a hybrid electrochemical-thermal-neural-network model. *J. Power Sources* **2020**, *455*, 227935. [[CrossRef](#)]
21. Park, T.-R.; Park, H.; Kim, K.; Im, C.-N.; Cho, J.-H. Heat and weight optimization methodology of thermal batteries by using deep learning method with multi-physics simulation. *Energy Convers. Manag.* **2021**, *236*, 114033.
22. Afzal, A.; Mohammed Samee, A.D.; Abdul Razak, R.K.; Ramis, M.K. Effect of spacing on thermal performance characteristics of Li-ion battery cells. *J. Therm. Anal. Calorim.* **2019**, *135*, 1797–1811. [[CrossRef](#)]
23. Afzal, A.; Saleel, C.A.; Suvanjan, K.P.; Mohammed, B. Parallel finite volume method - based fluid flow computations using OpenMP and CUDA applying different schemes. *J. Therm. Anal. Calorim.* **2021**, *145*, 1891–1909. [[CrossRef](#)]
24. Afzal, A.; Ansari, Z.; Ramis, M.K. Parallelization of Numerical Conjugate Heat Transfer Analysis in Parallel Plate Channel Using OpenMP. *Arab. J. Sci. Eng.* **2020**, *45*, 8981–8997. [[CrossRef](#)]
25. Afzal, A.; Samee, A.D.M.; Razak, R.K.A.; Ramis, M.K. Thermal management of modern electric vehicle battery systems (MEVBS). *J. Therm. Anal. Calorim.* **2020**, *144*, 1271–1285. [[CrossRef](#)]
26. Afzal, A.; Mokashi, I.; Khan, S.A.; Abdullah, N.A.; Azami, M.H.B. Optimization and analysis of maximum temperature in a battery pack affected by low to high Prandtl number coolants using response surface methodology and particle swarm optimization algorithm. *Numer. Heat Transf. Part A Appl.* **2020**, *79*, 406–435. [[CrossRef](#)]
27. Afzal, A.; Abidi, A.; Samee, A.; Razak, R.; Soudagar, M.; Saleel, A. Effect of parameters on thermal and fluid flow behavior of battery thermal management system. *Therm. Sci.* **2020**, *25*, 3775–3787. [[CrossRef](#)]
28. Mokashi, I.; Khan, S.A.; Abdullah, N.A.; Azami, M.H.; Afzal, A. Maximum temperature analysis in a Li-ion battery pack cooled by different fluids. *J. Therm. Anal. Calorim.* **2020**, *141*, 2555–2571. [[CrossRef](#)]
29. Afzal, A.; Razak, A.; Jilte, R.D.; Ibrahim, M.; Kumar, R.; Mujtaba, M.A.; Alshahrani, S.; Saleel, C.A. Thermal modelling and characteristic evaluation of electric vehicle battery system. *Case Stud. Therm. Eng.* **2021**, *26*, 101058. [[CrossRef](#)]
30. Haykin, S.S. *Neural Networks and Learning Machines/Simon Haykin*; Prentice Hall: New York, NY, USA, 2009.
31. MI, L. A brief description of the Levenberg-Marquardt algorithm implemented by levmar. *Found. Res. Technol.* **2005**, *77*, 1–6.
32. Afzal, A.; Ramis, M.K. Multi-objective optimization of thermal performance in battery system using genetic and particle swarm algorithm combined with fuzzy logics. *J. Energy Storage* **2020**, *32*, 101815. [[CrossRef](#)]
33. Afzal, A.; Samee, A.D.M.; Jilte, R.D.; Islam, T.; Manokar, A.M.; Abdul, K. Battery thermal management: An optimization study of parallelized conjugate numerical analysis using Cuckoo search and Artificial bee colony algorithm. *Int. J. Heat Mass Transf.* **2021**, *166*, 120798. [[CrossRef](#)]