Volume 2022, 13 pages
Article ID: ASTE-2202242112437

Applied Science and Technology Express

https://www.htpub.org/Applied-Science-And-Technology-Express/

Estimation of Dynamic Viscosity of Nanofluids via Graph Neural Network for Enhanced Oil Recovery

Mohammad Redzuan Firdaus bin Fozi

Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia, Johor Bahru, Malaysia.
Email: Redzuanfirdaus98@gmail.com

Article

Received:
28th January2022
Received in revised form:
19th February 2022
Accepted:
27th February 2022

Keywords:
Graph Neural Network,
Levenberg-Marquardt,
Dynamic Viscosity,
Nanofluid.

Abstract

The Graph Neural Network (GNN) is a sort of Artificial Neural Network used to anticipate a system's performance. In order to calculate the dynamic viscosity of nanofluid, a neural network is usually utilised for simulation. GNN has been proven to be quite beneficial in numerous studies. GraphSAGE and Artificial Neural Network-Multilayer Perceptron are two different forms of GNNs employed in this study (ANN-MLP). To compute the flow of nanofluids in porous media, the ANN-MLP with Levenberg-Marquardt training method is utilised. The model's data was gathered from the literature. By examining the value of R2, the hidden neurons for the neural network are carefully picked. The GNN is capable of simulating nanofluid dynamic viscosity. The simulation provides beneficial information.

1. Introduction

A graph, according to Zhou et al [1,] is a type of data structure that can model a set of data as vertices (nodes) and their relationships as edges. Due to the rise of the Industrial Revolution 4, the graph has played a significant part in machine learning and artificial intelligence (IR4.0). Researchers that combine graphs with machine learning receive a lot of attention because the graph has a lot of promise since it can deal with a lot of data or systems, such as data from social sciences (social networks) and natural sciences (natural networks) (physics, chemistry, and biology).

1.1. Graph Neural Network and Its Application

The Graph Neural Network is a computational technology that is utilised in a variety of fields. According to Irfan et al. [2,] dynamic viscosity of nanofluids is essential in oil and gas, especially in oil recovery. Artificial intelligence, according to Ahmadi et al [3] is one of the strategies utilised in oil recovery, and it has been introduced to be useful in modelling the dynamic viscosity of nanofluids. It can be used with multilayer-perceptron, regression forest, and other techniques, according to Ramezanizadeh et al [4] and Ahmadi et al [5]. The Neural Network, on the other hand, is primarily utilised for forecasting. Its input and output data are coupled for predicting purposes, similar to neurons in the human brain [6].

Directed and undirected graphs are the two types of graphs. The former has symmetrical edges between two vertices [7], while the later has symmetrical edges between two vertices [8]. A graph G is defined as a pair (N, E), where N denotes the number of nodes and E denotes the number of edges. At least one edge exists for each node [9]. The nodes are dendrites, and the output is the axon terminal, with the signal being transmitted by myelinated axons [6]. According to Zell [10], the inputs or signals in artificial neural networks are real numbers, and the connections are transfer functions. There is a weight that exists between two nodes that modifies as needed during the training process. In an artificial neural network, the weight is crucial since it determines the most likely value or weight during simulation. There are other techniques for calculating the weight, including calculating the mean squared error, calculating correlation coefficients, and so on [4].

GNN (Graph Neural Network) is a hybrid of a graph and a neural network. GNN has been used to a variety of fields, including social and natural sciences [1]. It is mostly used for forecasting purposes. Its major features are nodes as inputs and outputs, with their weight and transfer functions linking the nodes together.

1.2. Model of GNN: GraphSAGE and Multi-Layer Perceptron

The frameworks of the neural network must be understood in order to simulate the process. The framework of GraphSAGE was introduced by Hamilton et al [12]. It is highly handy to construct a low-dimensional vector of nodes from a graph with numerous properties that represent nodes. GraphSAGE produces characteristics based on the node's immediate surroundings [1]. This is especially beneficial for unseen nodes, as their nearby neighbourhoods can be used to represent them.

Multi-layer perceptron is another variation of GNN framework. The multilayer perceptron is the preferred framework for this study since it is simple to use and requires few resources. In comparison to GraphSAGE, the programme required a high-end processor to run. MLP can be executed on low-end hardware with a simple MATLAB programme. MLPs are a type of feedforward neural network that has multiple layers of input, hidden layer, and output. The concealed layer's size is usually decided through trial and error. It works in the same way as GraphSAGE, where aggregation determines a feature's prior feature.

1.3. Nanofluids Dynamic Viscosity Prediction

Primary oil recovery, secondary oil recovery, and enhanced oil recovery (EOR) are the three major stages in oil recovery [2]. Many studies have suggested that nanofluids or nanoparticles could significantly aid EOR [2]. Thermal conductivity, dynamic viscosity, and other physicochemical parameters are available in nanofluids [13].

The dynamic viscosity of nanofluids, on the other hand, is addressed in this study because it impacts nanofluid flow across porous media [13]. The temperature of the nanofluids, the size of the nanoparticles, and the volume fraction of the nanofluids are all factors that influence dynamic viscosity. These three variables have been the subject of numerous studies [3,4,5]. The researchers mostly employed multilayer-perceptron as their Neural Network of choice. According to Ahmadi et al [3,4] and Ramezanizadeh at al [5,] this Neural Network can mimic dynamic viscosity with reasonable accuracy.

The use of multilayer-perceptron, random forest have been reported in the literature. However, almost no literature available on the application of GNN on the dynamic viscosity of nanofluids. This research will investigate the possibility of using GNN for nanofluid-related problems. Here, we develop a GNN algorithm to model dynamic viscosity for a given nanofluid. We first introduce the

proposed GraphSAGE method, then we present the designed algorithm. Lastly results and discussions are presented.

2. Method

A multilayer perceptron is used as the main foundation for simulation of nanofluids in this study.

2.1. Levenberg-Marquardt algorithm

The steepest descent method and Gauss-algorithm Newton's are combined in the Levenberg-Marquardt training algorithm. Fortunately, it inherits the steepest descent method's stability as well as Gauss-fast Newton's method's convergence [10]. The basic process of the Levenberg-Marquardt method is represented in Figure 1. It begins by training the complex curvature, then switches to the steepest descent approach and calculates until a significant quadratic approximation is obtained. It then used Gauss-approach Newton's to approximate its convergences after obtaining the quadratic curvature.

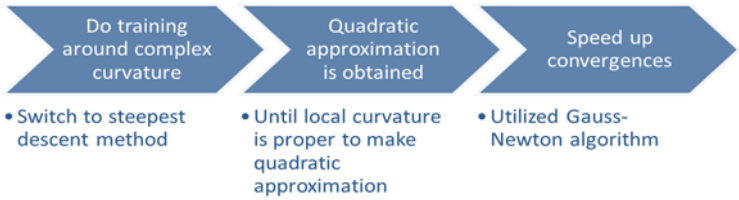

Figure 1. Basic flow of Levenberg-Marquardt method

To use the Levenberg-Marquardt training algorithm in neural network training, you'll need to know how to generate the Jacobian matrix and iteratively rearrange the training phase to update the weight. Node y is depicted as a variable element in Figure 2.

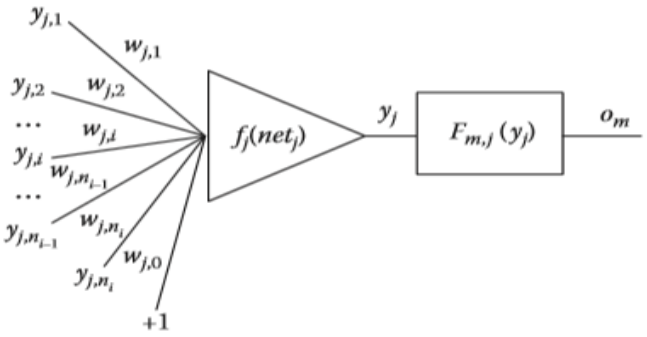

Figure 2. Connection of neuron j with the rest of the network

Because it might be perceived as either an input or an output from another neuron. y_{ij} is the output node in Figure 2, and it is calculated using

$$y_j = f_j(net_j). \tag{1}$$

Where f_j is the activation function of neuron j, and net_j is defined as

$$net_j = \sum_{i=1}^{ni} w_{j,i} y_{j,i} + w_{j,0}, \tag{2}$$

and y_{ij} is the i'th input node of neuron j weighted by w_{ij}, and $w_{j,0}$ is bias weight of neuron j. From Eq. (1), the derivative of net_j is

$$\frac{\delta net_j}{\delta w_{i,j}} = y_{i,j}, \tag{3}$$

and the slope f_j is

$$s_j = \frac{\delta y_j}{\delta net_j} = \frac{\delta f_j(net_j)}{\delta net_j}. \tag{4}$$

Between the output node, y_j, and network output, om, there is a complex nonlinear relationship. The relationship is

$$o_m = F_{m,j}(y_j), \tag{5}$$

which obtains

$$\frac{\delta o_m}{\delta y_j} = F'_{m,j}(y_j). \tag{6}$$

The complexity of this nonlinear function $F_{m,j}(y_j)$ depends on how many other neurons between neuron j and output m. If neuron j is at network output m, then $o_m=y_j$ and $F'_{m,j}(y_j)=1$. Elements of the Jacobian matrix are

$$\frac{\delta e_{p,m}}{\delta w_{j,i}} = \frac{\delta(d_{p,m} - o_{p,m})}{\delta w_{j,i}} = -\frac{\delta o_{p,m}}{\delta w_{j,i}} = -\frac{\delta o_{p,m}}{\delta y_j}\frac{\delta y_j}{\delta net_j}\frac{\delta net_j}{\delta w_{j,i}} \tag{7}$$

Combining Eq (4) to Eq (7) yeilds

$$\frac{\delta e_{p,m}}{\delta w_{j,i}} = -F'_{m,j} s_j y_{j,i}. \tag{8}$$

Let

$$E\varphi_{m,j} = s_j F'_{m,j}. \tag{9}$$

the element of the Jacobian matrix can be found by substituting Eq (9) in Eq (8). We obtain

$$\frac{\delta e_{p,m}}{\delta w_{j,i}} = -\varphi_{m,j} y_{j,i}, \tag{10}$$

whereby y_{ij} is determined by forward propagation (inputs to output), and $\varphi_{m,j}$ is calculated from backward propagation (output to inputs). At output neuron, m, when, j=m, we found out that $\varphi_{m,j}=s_m$.

2.2. Training process for the neural network using Levenberg-Marquardt algorithm

The training process is represented in and described as follows.

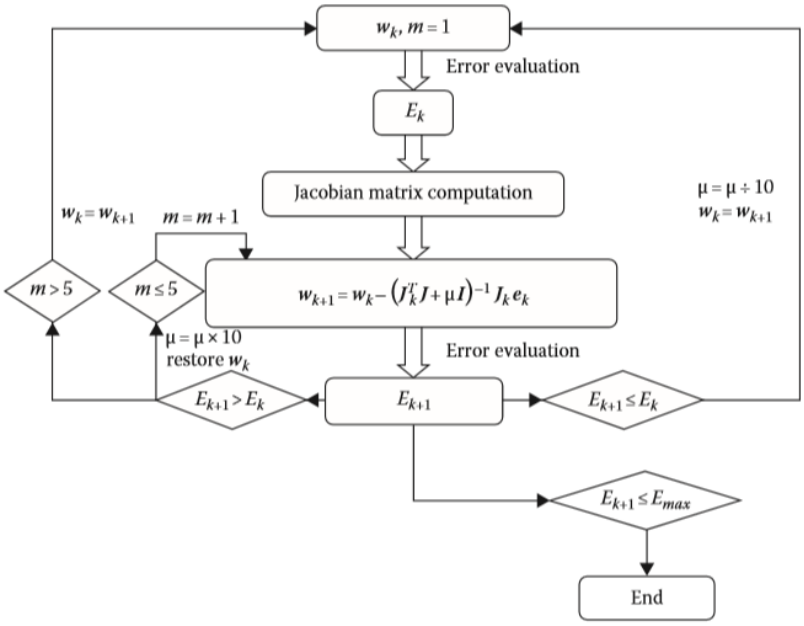

Figure 3. Training process with Levenberg-Marquardt algorithm (update rule)

i. The initial weight is randomly generated, and the total error (SSE) is evaluated.

ii. Then, weight updates as directed by Eq. 3.24.

iii. With the new weight, do another evaluation of total error.

iv. If the current error is increased as a result of the update, then retract the step (such as reset the weight vector to the previous value) and increase the combination coefficient μ by a factor of 10 or by some other factor. Then go to step (ii) and try to update again.

v. If the current error is decreased as a result of the update, then accept the step (such as keeping the new weight) and decrease the combination coefficient by some factor or by the same step as step (iv).

vi. Go to step (ii) with the new weight until the total error is smaller than the required value.

2.3. Artificial Neural Network modeling structure

Neural network (NN) is a model that is driven by data. [17,18] This means that in order to model the dynamic viscosity of nanofluids, experimental data is required. This study relies on publicly available data. There are approximately 520 data points gathered from the literature.

The data is separated into three sections at random: training, validation, and testing. They are split into three groups: 70% for training, 15% for validation, and 15% for testing. The concentration and temperature of the nanofluids, and the size of nanoparticles are the model's inputs and the dynamic viscosity of the nanofluids is the model's output.

For each hidden layer, there are 17 hidden neurons. The sigmoid function, or tanh(x), is employed as a transfer function. The transfer function [17,18] is a function that connects input and output. The created neural network is shown in Figure 4.

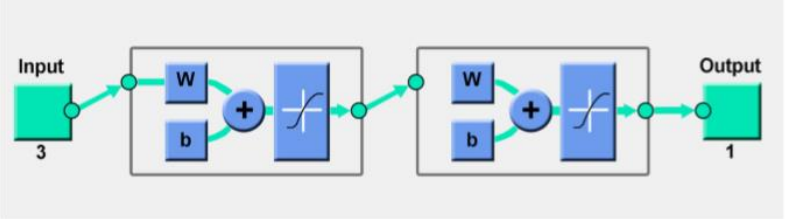

Figure 4. A three-layer network for modelling with 3-17-1 arrangements

quation depicts the relationship between weight, bias input, and output values in the neuron.

$$y = [LW]_{1\times17} \times \tanh([IW]_{17\times3} \times [I]_{3\times1} + b_1) + b_2. \quad (11)$$

The Jacobian matrix is the most significant part of the Levenberg-Marquardt training algorithm. The unknowns y_i and j can be determined using forward and backward propagation techniques, respectively.

3. Results and Discussion

This section discusses the model's effectiveness. The correlation coefficient and mean square error are calculated using ANN. For the simulation, 17 hidden neurons were used. This is due to the fact that mean square error, or error evaluation, is the smallest feasible when compared to another number of neurons. The ANN algorithm then is applied to determine the dynamic viscosity of nanofluids.

3.1. ANN Analysis

The R-squared and MSE values achieved during the model development are listed in Table 1. The R-squared is used as a criterion for finding the best hidden neurons. The model has a maximum R^2 value of 0.9983 for training, 0.9847 for validation, and 0.9821 for testing at 17 neurons. When compared to another number of hidden neurons, all the values approach to 1. The lowest mean square error values for 17 hidden neurons was 0.0078 for training, 0.1124 for validation, and 0.0597 for testing. The minimal errors are indications of an efficient performance of the network.

The correlation between the variables in the model is shown in Table 2. Viscosity is the desired result. Temperature and viscosity have a negative correlation of -0.2061. Size and viscosity have a correlation of 0.0570. The volume fraction and viscosity have a correlation of 0.5131. There is a low correlation (less than 0.5) between all of the variables. Cook [19] said in his literature review that minimal correlations between data ensure that the L-M algorithm operates well, while highly correlated data degrades network performance.

Table 1. R-squared value and MSE value for hidden neuron

	R-squared			MSE		
	Training	Validation	Testing	Training	Validation	Testing
12	0.4967	0.6991	0.5767	2.2558	0.3886	1.3101
13	0.6934	0.7001	0.5608	1.1483	0.7993	2.4749
14	0.7034	0.7256	0.6019	1.1174	1.2448	1.4527
15	0.8523	0.8704	0.4178	0.3724	0.3669	6.1665
16	0.7315	0.4837	0.8081	0.9214	4.065	0.2405
17	0.9983	0.9847	0.9821	0.0078	0.1124	0.0597
18	0.6111	0.5411	0.6594	1.6994	0.5324	1.1495
19	0.672	0.9141	0.7869	1.4329	0.6144	0.4194
20	0.7133	0.5672	0.6876	0.9974	1.07	2.3688

Table 2. Correlation table between the variables used in the model

	Temperature	Size	Volume fraction	Viscosity
Temperature	1.0000	0.1774	0.0121	-0.2061
Size	0.1774	1.0000	-0.0307	0.0570
Volume fraction	0.0121	-0.0307	1.0000	0.5131
Viscosity	-0.2061	0.0570	0.5131	1.0000

During the training procedure, the mean square error diagram is shown in Figure 5. The MSE is significant since it tells the network when it's time to cease training (stopping point). The curve is formed by taking into account the amount of training stage iterations. The training phase's error evaluation is identical to the testing and validation stages'. When the validation error starts to climb, the training period will come to an end. At the 86th iteration, the most generalisations occur. The training phase is configured to end when the validation error hits a minimum value, as this will increase generalisations and reduce overfitting. Overfitting, on the other hand, isn't a major worry because the MSE is roughly equal.

The regression lines in Figure 6 depicts the correlation between the expected and actual data. When the coefficient is close to 1, the correlation between the predicted and actual value is higher. The R-squared value for the training phase is 0.9983, 0.9847 for the validation phase, and 0.9821 for the testing phase. Because these numbers are all near to one, the model can be used to estimate the dynamic viscosity of nanofluids.

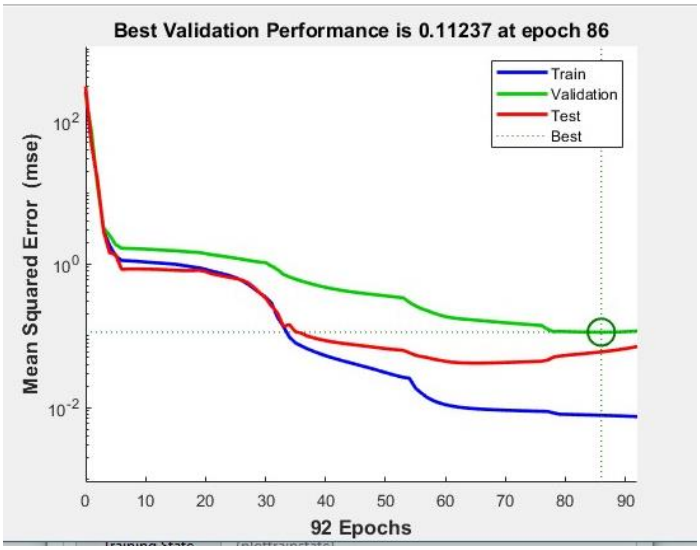

Figure 5. Mean square error diagram on which training process occurs

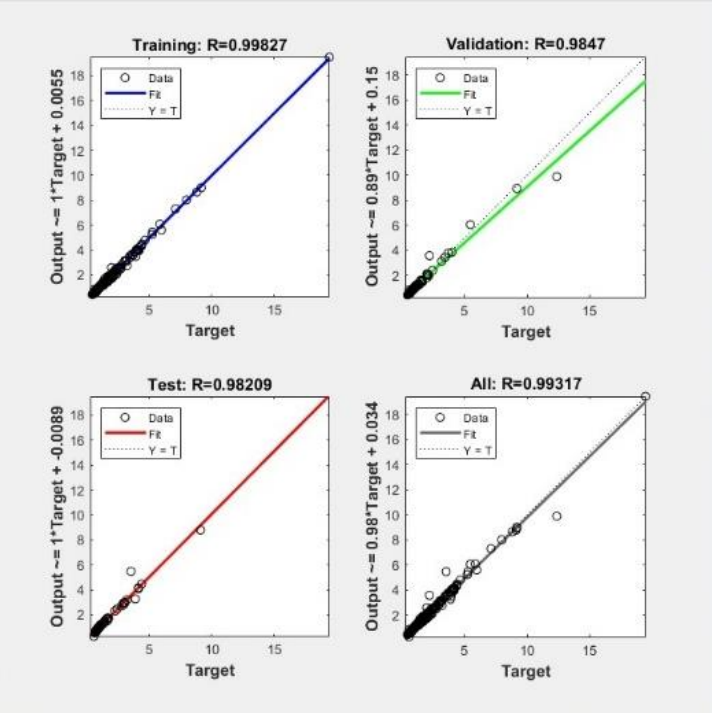

Figure 6. Regression line for the training set, validation set, and testing set

3.2. Dynamic Viscosity of Nanofluids

The dynamic viscosity of nanofluids is determined after the ANN model has been trained. It is made to prove whether or not the model has been properly trained. The result can provide insight into the dynamicity of nanofluid viscosity in relation to other parameters such as the influence of nanofluid temperature, volume, and nanoparticle size. The properties of nanofluids as they flow through porous media are also collected.

The graph of volume fraction versus viscosity for 0.01% -13% is shown in Figure 7. From 0.01 percent of volume fraction until about 5% of volume fraction, the curve remains pretty steady. This indicates that the volume percentage in that range has a minor impact on viscosity. From 5% to 10% volume fraction, the viscosity of the curve continuously increases. This suggests that the nanofluids' viscosity is influenced by the volume fraction in that region. From 10% to 13% volume fraction, the curve significantly increases with respect to viscosity. As a result, viscosity is affected by the volume fraction in that range. As a result, the hypothesis that an increase in volume fraction leads to an increase in nanofluid viscosity is confirmed.

The curve of viscosity vs. temperature is shown in Figure 8 . When the temperature is between 25 and 70 degrees Celsius, the viscosity reduces as the temperature rises, supporting Lee J.H et al.'s[20] assertion. When the temperature is between 70 and 150 degrees Celsius, a significant increase in viscosity is observed as the temperature rises. There is a point at which the viscosity increases as the temperature rises. The hysteresis phenomenon is the name for this effect [23].

The relationship between viscosity and size is depicted in Figure 9. Nanoparticles with sizes of 12,30,31,37,48, and 99nm were utilised. The size of nanoparticles and the viscosity of the nanofluid have almost no relationship. This means that particle size has no effect on the viscosity of nanofluids, as it neither raises nor decreases it. These findings support the findings of Pastoriza-Gallego et al. [22], who found that nanoparticle size is insignificant.

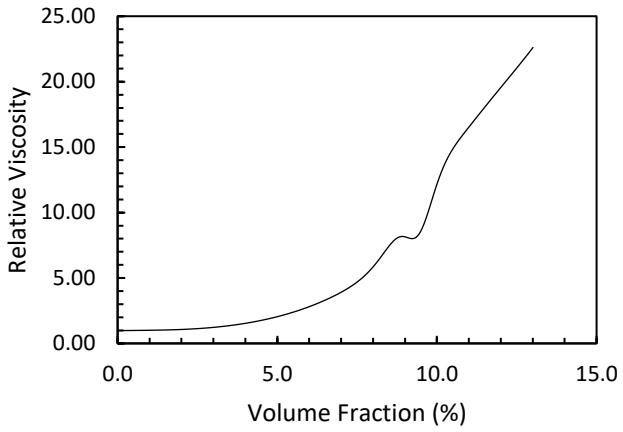

Figure 7. Viscosity vs. volume fraction

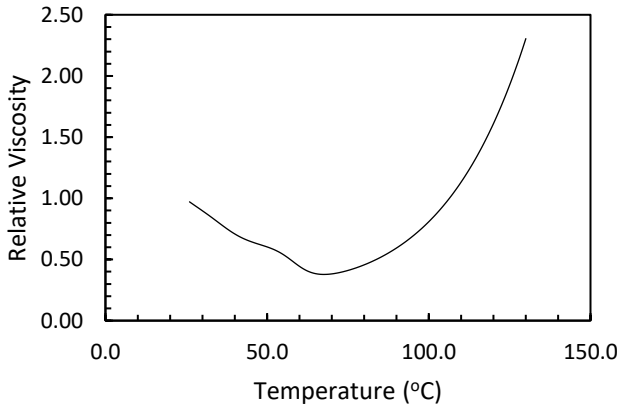

Figure 8. Viscosity vs temperature

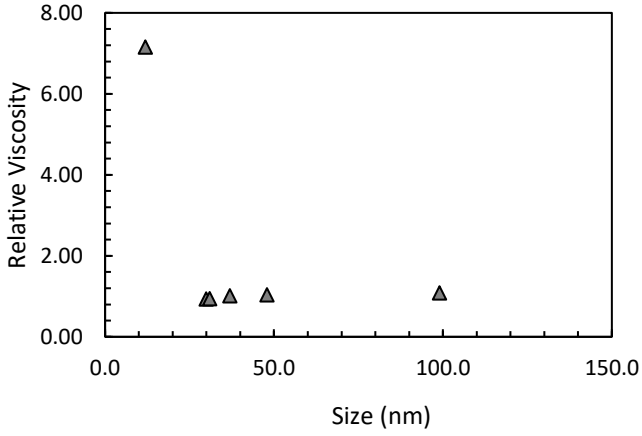

Figure 9. Viscosity vs size

3.3. Comparison between Actual and Simulated Data and Error Analysis

A comparison of calculated and real values is presented in this section. The aim is to determine how close they are in terms of inaccuracy. The relative error between the calculated and real numbers is determined during the error analysis. This is how to determine the mean relative error.

$$Relative\ Error\ \% = \frac{\left|data_{predicted} - data_{actual}\right|}{data_{actual}} \times 100 \quad (12)$$

$$Mean\ Relative\ Error\ \% = \frac{\sum Relative\ Error\ \%}{Total\ Number\ of\ Data} \quad (13)$$

The contrast between actual and calculated data is shown in Figure 10. Actual and computed data are in close proximity to one another. To put it another way, the calculated data and the real data are closely connected. This indicates that the model has been thoroughly trained and is capable of simulating the dynamic viscosity of nanofluids.

The regression line is drawn between the actual and calculated data in Figure 11. The data have a correlation coefficient of 0.9963. The higher the correlation, the closer the value is to 1. The range of inaccuracy between real and estimated data is shown in Figure 12. The majority of the errors are under ten percent, with the greatest inaccuracy being over forty percent. Table 3 demonstrates that the mean relative error for this model is just 4%, which is acceptable given the MSE values of 0.0078, 0.1124, and 0.0597 for training, validation, and testing, respectively.

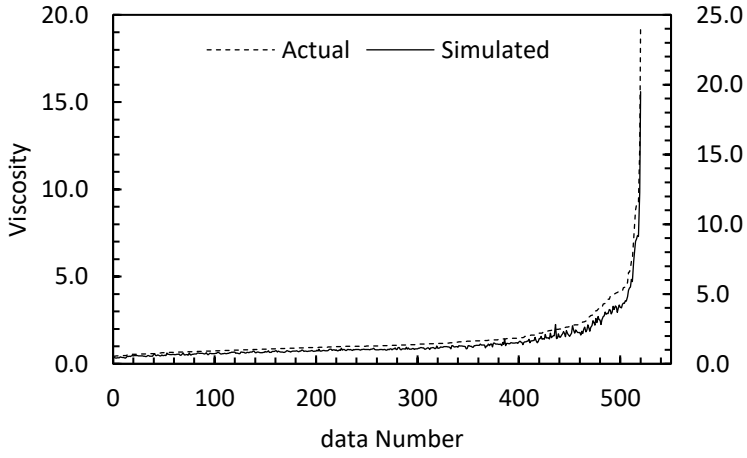

Figure 10. Comparison between actual data and simulated data

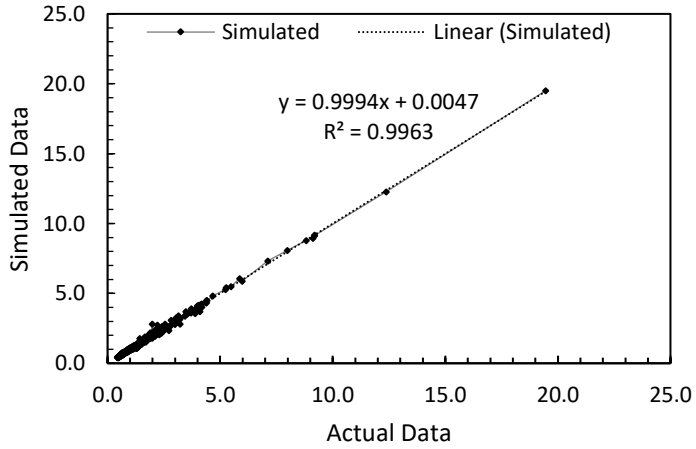

Figure 11. Regression line between actual data and simulated data

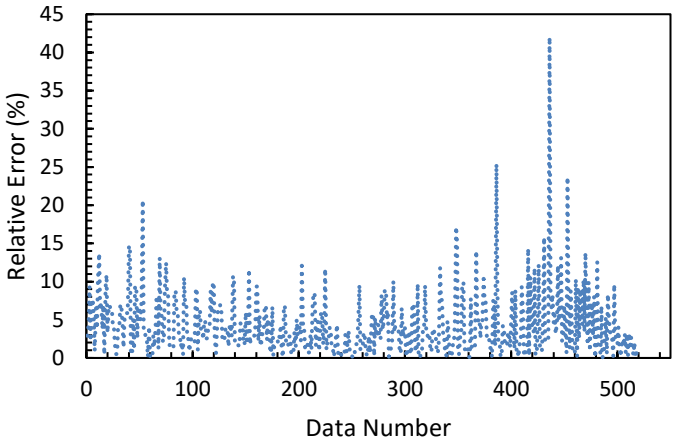

Figure 12. Range of error between actual data and simulated data

Table 3. Mean relative error between the simulated and actual data

	ANN-MLP
Mean Relative Error (%)	4.098634493

4. Conclusions

In this study, ANN-MLP was used to develop a formula for estimating the dynamic viscosity of nanofluids by taking into account the temperature, volume fraction, and size of nanofluids as input variables. The highest R^2 values were found with 17 hidden neurons: 0.9983 for training, 0.9847 for validation, and 0.9821 for testing. Size showed the lowest correlation with viscosity, with 0.0570, and volume fraction had the highest correlation, with 0.5131. Temperature and viscosity had a negative correlation of -0.2061. As a result, they all showed small correlations, indicating that the network is performing efficiently. Furthermore, simulation data obtained by the ANN algorithm was compared with actual data. The actual and estimated data represented a correlation value of 0.9963. The dynamic viscosity of nanofluids can be simulated using this approach. The average relative error, according to the error analysis, was 4.098%, indicating that the network is well-trained and capable of calculating viscosity.

References

1. Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L, & Sun, M., Graph neural networks: A review of methods and applications. 2018, arXiv preprint arXiv:1812.08434.

2. Irfan, S. A., Shafie, A., Yahya, N., & Zainuddin, N. (2019). Mathematical Modeling and Simulation of Nanoparticle-Assisted Enhanced Oil Recovery: A Review. Energies. 2019. **12**(8): p.1575.

3. Ahmadi, M. H., Mohseni-Gharyehsafa, B., Ghazvini, M., Goodarzi, M., Jilte, R. D., & Kumar, R. Comparing various machine learning approaches in modeling the dynamic viscosity of CuO/water nanofluid. Journal of Thermal Analysis and Calorimetry, 2020. **139**(4); p. 2585-2599.

4. Ramezanizadeh, M., Ahmadi, M. A., Ahmadi, M. H., & Nazari, M. A. Rigorous smart model for predicting dynamic viscosity of Al 2 O 3/water nanofluid. Journal of Thermal Analysis and Calorimetry.2019. **137**(1);p.307-316.

[5] Ahmadi, M. H., Mohseni-Gharyehsafa, B., Farzaneh-Gord, M., Jilte, R. D., Kumar, R., & Chau, K. W. Applicability of connectionist methods to predict dynamic viscosity of silver/water nanofluid by using ANN-MLP, MARS, and MPR algorithms. Engineering Applications of Computational Fluid Mechanics, 2019. **13**(1); p.220-228.

[6] Chen, Y. Y., Lin, Y. H., Kung, C. C., Chung, M. H., & Yen, I. Hsuan." Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes. Sensors, 2019. **19**(9); p.2047.

[7] Kamada, T., & Kawai, S. An algorithm for drawing general undirected graphs. Information processing letters, 1989. **31**(1);p.7-15.

[8] Gansner, E. R., Koutsofios, E., North, S. C., & Vo, K. P. A technique for drawing directed graphs. IEEE Transactions on Software Engineering, 1993. **19**(3); p.214-230.

[9] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. The graph neural network model. IEEE Transactions on Neural Networks, 2008. **20**(1); p.61-80.

[10] Zell, A. Simulation Neuronaler Netze (Simulation with Neuronal Networks). Wissenschaftsverlag: Oldenbourg.2003.

[11] Anand, R. An Illustrated Guide to Graph Neural Networks. Retrieved November 23, 2020, from https://medium.com/dair-ai/anillustrated. guide-to-graph-neural-networks-d5564a551783

[12] Hamilton, W., Ying, Z., & Leskovec, J. Inductive representation learning on large graphs. In Advances in neural information processing systems, 2017; p. 1024-1034.

[13] Goldberg, E., Scheringer, M., Bucheli, T. D., & Hungerbühler, K. Prediction of nanoparticle transport behavior from physicochemical properties: machine learning provides insights to guide the next generation of transport models. Environmental Science: Nano, 2015, **2**(4); p.352-360.

[14] Savyakhosla (2019, August 26). CNN: Introduction to Pooling Layer. Retrieved January 17, 2021, from https://www.geeksforgeeks.org/cnnintroduction. to-pooling-layer/

[15] Alvim, R. S., Babilonia, O. A., Celaschi, Y. M., & Miranda, C. R. (2017). Nanoscience applied to oil recovery and mitigation: a multiscale computational approach. MRS Advances, 2017. **2**(9); p. 477.

[16] Sangani, M. F., Owens, G., Nazari, B., Astaraei, A., Fotovat, A., & Emami, H. (2019). Different modeling approaches for predicting titanium dioxide nanoparticles mobility in intact soil media. Science of The Total Environment, 2019. **665**; p.1168-1181.

[17] Ahmadi, M. H., Sadeghzadeh, M., Maddah, H., Solouk, A., Kumar, R., & Chau, K. W. (2019). Precise smart model for estimating dynamic viscosity of SiO2/ethylene glycol□water nanofluid. Engineering Applications of Computational Fluid Mechanics, 2019. **13**(1); p.1095-1105.

[18] Yu, H., & Wilamowski, B. M. (2011). Levenberg-marquardt training. Industrial electronics handbook, 2011. **5**(12);p. 1.

[19] Cook, N. D. Correlations between input and output units in neural networks. Cognitive Science, 1995. **19**(4), p.563-574.

[20] Lee, J. H., Hwang, K. S., Jang, S. P., Lee, B. H., Kim, J. H., Choi, S. U., & Choi, C. J. Effective viscosities and thermal conductivities of aqueous nanofluids containing low volume concentrations of Al2O3 nanoparticles. International Journal of Heat and Mass Transfer, 2008. **51**(11-12); p.2651-2656.

[21] Esfe, M. H., Saedodin, S., Wongwises, S., & Toghraie, D., An experimental study on the effect of diameter on thermal conductivity and dynamic viscosity of Fe/water nanofluids. Journal of Thermal Analysis and Calorimetry,2015. **119**(3); p.1817-1824.

[22] Pastoriza-Gallego, M. J., Casanova, C., Legido, J. L., & Piñeiro, M. M., CuO in water nanofluid: influence of particle size and polydispersity on volumetric behavior and viscosity. Fluid phase equilibria, 2011.**300**(1-2); p.188-196.

[23] Nguyen, C. T., Desgranges, F., Roy, G., Galanis, N., Maré, T., Boucher, S., & Mintsa, H. A., Temperature and particle-size dependent viscosity data for water-based nanofluids□hysteresis phenomenon. International journal of heat and fluid flow,2007. **28**(6); p.1492-1506.

[24] Heidari, E., Sobati, M. A., & Movahedirad, S., Accurate prediction of nanofluid viscosity using a multilayer perceptron artificial neural network (MLP-ANN). Chemometrics and intelligent laboratory systems,2016. **15**; p.73-85.