International Journal of Computer Science and Control Engineering
2018; 6(2): 9-21
DOI: 10.5281/zenodo.1285164
http://www.openscienceonline.com/journal/csce

Prediction of Electrical Output Power of Combined Cycle Power Plant Using Regression ANN Model

Elkhawad Elfaki[1], Ahmed Hassan Ahmed Hassan[2, *]

[1]Department of mechanical Engineering, Bisha University, Bisha, Saudi Arabia

[2]Department of mechanical Engineering, Ondokuz Mayis University, Samsun, Turkey

Email address

aahmed@ub.edu.sa (E. Elfaki); 15210457@stu.omu.edu.tr (A. H. A. Hassan)

[*]Corresponding author

To cite this article

Elkhawad Elfaki, Ahmed Hassan Ahmed Hassan. Prediction of Electrical Output Power of Combined Cycle Power Plant Using Regression ANN Model. *International Journal of Computer Science and Control Engineering*. Vol. 6, No. 2, 2018, pp. 9-21, DOI: 10.5281/zenodo.1285164

Received: April 3, 2018; **Accepted**: April 23, 2018; **Published**: June 5, 2018

Abstract

Recently, regression artificial neural networks are used to model various systems that have high dimensionality with nonlinear relations. The system under study must have enough dataset available to train the neural network. The aim of this work is to apply and experiment various options effects on feed-foreword artificial neural network (ANN) which used to obtain regression model that predicts electrical output power (EP) of combined cycle power plant based on 4 inputs. Dataset is obtained from an open online source. The work shows and explains the stochastic behavior of the regression neural, experiments the effect of number of neurons of the hidden layers. It shows also higher performance for larger training dataset size; at the other hand, it shows different effect of larger number of variables as input. In addition, two different training functions are applied and compared. Lastly, simple statistical study on the error between real values and estimated values using ANN is conducted, which shows the reliability of the model. This paper provides a quick reference to the effects of main parameters of regression neural networks.

Keywords

Neural Networks, Regression, Combined Power Cycle, MATLAB Neural Networks Toolbox

1. Introduction

Electricity has been one of the main essential resources to the human activities. Power plant has been established to provide human communities with the needed amount of electricity. Power provided from power plants fluctuates through the year due to many reasons including the environmental conditions. The accurate analysis of thermodynamic power plants using mathematical models requires high number of parameters and assumptions, in order to represent the actual system unpredictability [1, 2]. Instead of mathematical modelling the system's thermodynamics, machine learning approaches can be used [3, 4]. One of these methods is artificial neural networks (ANNs). With the ability of artificial neural networks to address nonlinear relationships, environmental conditions are studied as inputs of the model, and the generated power as the output of the model. Using this model, we can predict the output power of the plant given the environmental conditions.

Artificial neural networks (ANNs) were originally proposed in the mid-20th century as a computational model of the human brain. Their use was limited due to the limited computational power available at the time, and some unsolved theoretical problems. However, they have been increasingly studied and applied with the recent existence of higher computational power and the availability of datasets [5]. In a typical modern power plant, a large amount of parametric data is stored over long periods of time; therefore, a large data based on the operational data is always ready without any additional cost [2].

Researches have considered ANN to model many various engineering systems [6-13]. Many researchers reported the feasibility and reliability of ANN models as simulation and analysis tool for power plant processes and components [14-22]. Relatively few studies have considered the Steam Turbine (ST) in a combined cycle power plant (CCPP) [2, 23-25]. In [1] the total power output of a cogeneration power plant with three gas turbine, three HRSGs and one steam turbine were predicted. Niu [24] studied the control strategy of the gas turbine in a combined cycle power plant using a linearization model technique. Samani [2] used two subsequent artificial neural networks to model combined cycle power plant with inputs as the relative humidity, atmospheric pressure, ambient temperature and the exhaust vacuum of the steam turbine. The exhaust steam pressure alone is a function of ambient conditions and is not a deterministic parameter. Tüfekci [23] and Kaya [25] compared various machine learning methods to predict the full load electrical power output of a base load operated combined cycle power plant. In this work, detailed study of regression ANN model is studied.

1.1. Combined Cycle Power Plant (CCPP)

One kind of power plants is the combined cycle power plant (CCPP), which is composed of gas turbines (GT), steam turbines (ST) and heat recovery steam generators (HRSG).

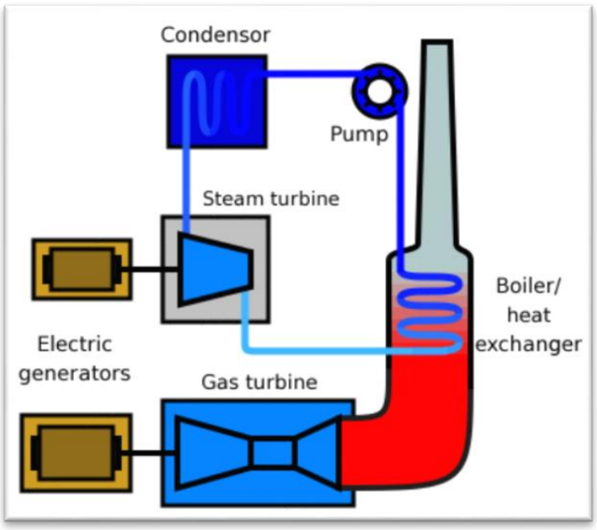

Figure 1. *Combined Cycle Power Plant CCPP diagram [23].*

In a CCPP, the electricity is generated by gas and steam turbines, which are combined in one cycle [24]. A combined-cycle power plant produces up to 50 percent more electricity from the same fuel than a traditional simple-cycle plant by routing the waste heat from the gas turbine to the nearby steam turbine, which generates extra power [26]. Combined cycle power plant mechanism can be stated as follows [26]:

1. Fuel burns at the gas turbine, makes the turbine blades spinning and driving electricity generators.
2. Heat Recovery Steam Generator (HRSG) captures exhaust heat from the gas turbine. The HRSG creates steam from the gas turbine exhaust heat and delivers it to the steam turbine.
3. Steam turbine uses the steam delivered by the heat recovery system to generate additional electricity by driving an electricity generator.

Gas turbine load is sensitive to the ambient conditions; mainly ambient temperature (AT), atmospheric pressure (AP), and relative humidity (RH). However, steam turbine load is sensitive to the exhaust steam pressure (or vacuum, V) [21 ,23].

Combined cycle power plants (CCPPs) have a higher fuel conversion efficiency compared to the conventional power plants, i.e. consuming less fuel to produce the same amount of electricity, which results in lower power price and less emission to the environment [26].

1.2. ANN Definition

Artificial neural networks (ANNs) or connectionist systems are a computational model used in computer science and other research disciplines, which is based on a large collection of simple neural units (artificial neurons), loosely analogous to the observed behavior of a biological brain's axons. Each neural unit is connected with many others, and links can enhance or inhibit the activation state of adjoining neural units. Each individual neural unit computes using summation function. There may be a threshold function or limiting function on each connection and on the unit itself, such that the signal must surpass the limit before propagating to other neurons. These systems are self-learning and trained, rather than explicitly programmed, and excel in areas where the solution or feature detection is difficult to express in a traditional computer program [5]. The training of ANN starts with random weights and then neurons works to make sure the error is minimal.

1.3. ANN Advantages

Major advantage of using ANN is non-parametric model while most of statistical methods are parametric model that need higher background of statistic. In addition, ANNs easily handles highly non-linear modelling (main advantage). However, ANN is a black box learning approach, cannot interpret relationship between input and output and cannot deal with uncertainties [5].

1.4. Regression ANN

Neural networks are good at fitting functions. In fact, there is proof that a fairly simple neural network can fit any practical function [5].

1.5. ANN in MATLAB

Neural Network Toolbox™ provides algorithms, functions, and apps to create, train, visualize, and simulate neural networks. It includes algorithms and tools for regression, pattern recognition, classification, clustering, deep learning, time series and dynamic systems, and many others which cover the usage of ANN models [27, 28]. The work flow for the neural network design process has seven primary steps:

1. Collect data
2. Create the network
3. Configure the network

4. Initialize the weights and biases
5. Train the network
6. Validate the network
7. Use the network

Some of these steps could be done automatically using default values and settings in the toolbox; however, user can set every detail by himself. Neural Network Toolbox offers four levels of design i.e. four different levels at which the Neural Network Toolbox™ software can be used.

The first level is represented by the GUIs. These provide a quick way to access the power of the toolbox for many problems of function fitting, pattern recognition, clustering and time series analysis. In addition a.m Matlab code script can be generated with the desired level of details copying settings used in the network study.

The second level of toolbox use is through basic command-line operations. The command-line functions use simple argument lists with intelligent default settings for function parameters. (user can override all of the default settings, for increased functionality.)

A third level of toolbox use is customization of the toolbox. This advanced capability allows user to create custom neural networks, while still having access to the full functionality of the toolbox.

The fourth level of toolbox usage is the ability to modify any of the code files contained in the toolbox. Every computational component is written in MATLAB® code and is fully accessible.

1.6. Regression ANN in MATLAB

Regression (Fit Data) in Neural Network Toolbox in Matlab can be accessed using GUI or command-line functions [5]. There are two GUIs can be used to design and train the network [15]:

1. Neural Network tool (nntool), which is the general neural network tool, offers full control of settings. Using this GUI, user can design any type of neural network, not only the regression ANN.
2. Neural Fitting tool (nftool), which leads user through solving a data fitting problem, solving it with a two-layer feed-forward network trained with Levenberg-Marquardt or scale conjugate gradient back-propagation. It has limited set of options. User can select data from the MATLAB® workspace or use one of the example datasets. After training the network, evaluate its performance using mean squared error and regression analysis. Further, analyze the results using visualization tools such as a regression fit or histogram of the errors. User can then evaluate the performance of the network on a test set.

1.7. Aim of the Study

Aim of this work is to apply and experiment various options effects on feed-foreword artificial neural network (ANN) which used to obtain regression model that predicts electrical output power (EP) of combined cycle power plant based on 4 inputs. More specifically, this work uses MATLAB neural networks toolbox to study stochastic behavior of the regression neural, effect of number of neurons of the hidden layers, effect of data subset size for training, effect of number of variables as input, different training functions results, data preprocessing, and statistical error study.

2. Method Description

In this study, MATLAB neural networks toolbox is used; database is obtained freely from [29]. Through this paper, terms Test and Performance have the following meanings:

1. Test: refers to the test of the whole dataset (9568 observations) which gives results that are more realistic.
2. Performance: means squared errors (MSE).

The main scheme in this study is conducting comparisons between resulted networks using various variations of options. Comparison will always be between the performances of networks on the whole dataset (Test dataset). The following subsections describe the data, shows how training sub dataset is obtained, illustrates which features (inputs) to be studied, discusses data normalization and shows selection of neural network structure size.

2.1. Data Overview

Dataset is obtained from online site [29]. The dataset contains 9568 data points collected from a Combined Cycle Power Plant over 6 years (2006-2011), when the power plant was set to work with full load. Features consist of hourly average ambient variables Temperature (AT), Ambient Pressure (AP), Relative Humidity (RH) and Exhaust Vacuum (V) to predict the net hourly electrical energy output (EP) of the plant. Dataset features are summarized at Table 1.

Table 1. *Features (dataset variables) summary.*

Feature	Symbol	Type	minimum	maximum	unit
Ambient temperature	AT	Input	1.81	37.11	°C
Ambient Pressure	AP	Input	992.89	1033.30	mbar
Relative Humidity	RH	Input	25.56%	100.16%	----
Exhaust Vacuum	V	Input	25.36	81.56	cm Hg
Net hourly electrical energy output	EP	Output	420.26	495.76	MW

Figure 2 shows the relationships between each of the variables (AT, AP, RH, and V) and output power (EP) with the linear regression for each chart.

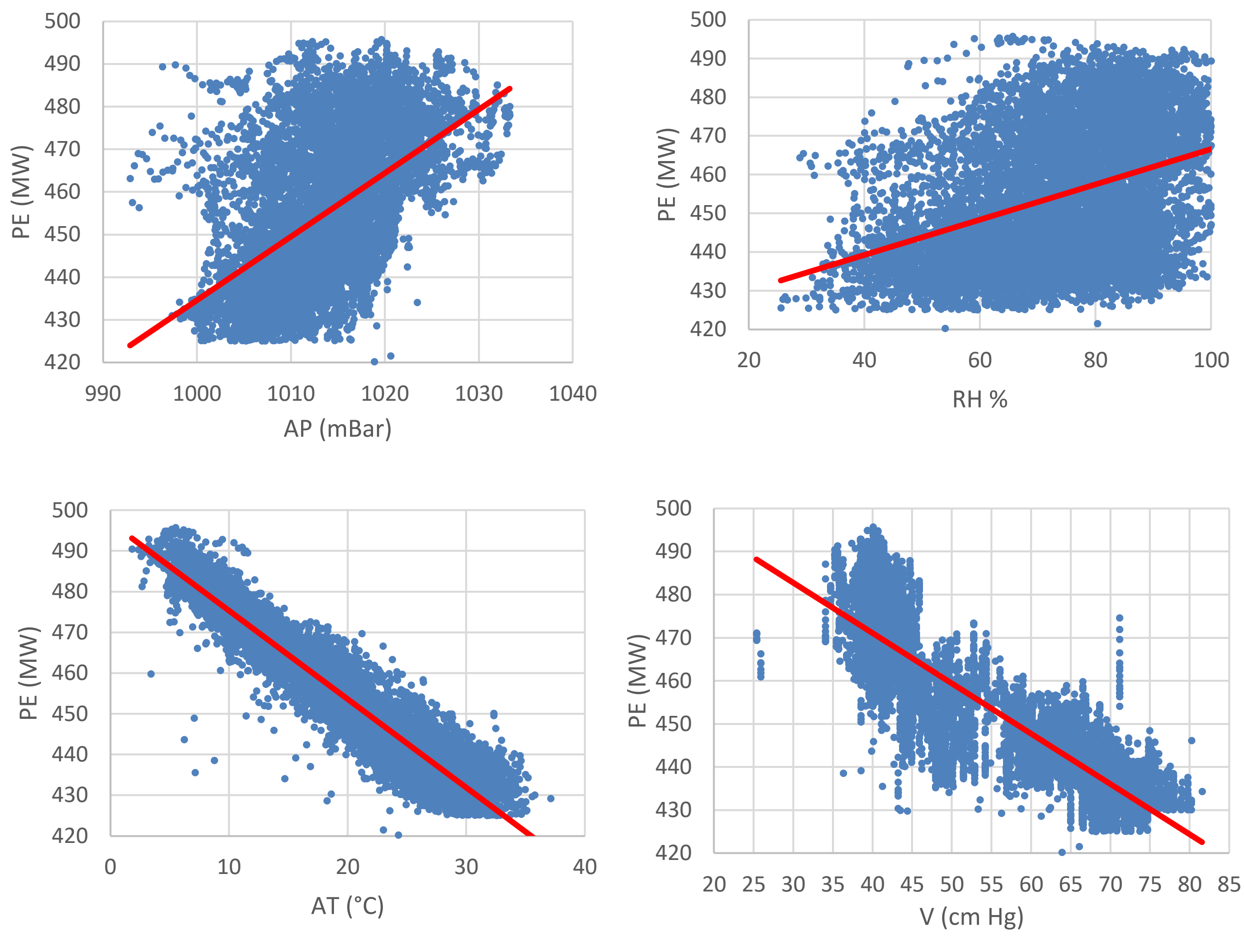

***Figure 2.** Linear Correlation between inputs and output PE.*

2.2. Sub Dataset Selection

Since the main goal of this work is to apply and test regression with neural network, so no need for all the dataset. Only a smaller subset is systematically picked from the dataset to train, validate and initial test of the network. Matlab Neural Network toolbox divides the dataset to subsets of train, validate and test, by default percentages as 75, 15 and 15%. Since we have huge dataset, we can run the test on it, so we will reduce the test subset to 0%, training subset will be 75% and validation subset will be 25%. Finally, test is performed with all the data points from the original dataset and compared with different subset sizes.

2.3. Normalization

One way to improve training networks is to linearly normalize inputs and outputs to certain range. The standard normalization maps the feature to the range of (-1, 1); which is default in the Matlab Neural Network toolbox; both inputs and outputs are normalized by default. Other range may be used, e.g. (0.01, 0.09). Another normalization practice is to map feature to a range with specified mean and variance. Typically the mean would be (0) and standard derivation would be (1). Since Matlab Neural Network toolbox makes this step for us, we do not have to worry about it. However, mapping to range (0.01, 0.09) is also applied and results are compared to the use of not normalized data.

2.4. Feature Selection

As shown in the Figure 2, AT and V have a strong negative linear relations to the output PE, AP and RH have weak positive linear relations to the output PE. This can be further shown with the correlation coefficients between inputs and the output (PE) as shown at Table 2.

***Table 2.** Variables correlation with output PE.*

Input	R	R^2	description
AT	-0.9481	0.8989	Negative & strong
V	-0.8698	0.7565	Negative & strong
AP	0.5184	0.2688	Positive & weak
RH	0.3898	0.1519	Positive & weak

The linear relations strength between the variables are shown in the Table 3 of correlation R and correlation strength R^2. It can be seen that AT and V are strongly linearly related to each other and to the output PE. While AP and RH have weak linear relations to all other variables and output.

Although it is obvious that the governing variables are AT and V, the effect of absence and presence of each of variables is studied.

Table 3. *Correlation between features.*

R	AT	V	AP	RH	PE
AT	1	0.84	-0.51	-0.54	-0.95
V	0.84	1	-0.41	-0.31	-0.87
AP	-0.51	-0.41	1	0.1	0.52
RH	-0.54	-0.31	0.1	1	0.39
PE	-0.95	-0.87	0.52	0.39	1

R^2	AT	V	AP	RH	PE
AT	1	0.71	0.26	0.29	0.9
V	0.71	1	0.17	0.1	0.76
AP	0.26	0.17	1	0.01	0.27
RH	0.29	0.1	0.01	1	0.15
PE	0.9	0.76	0.27	0.15	1

2.5. Setting of Networks

2.5.1. Two-Layer Feed-Forward Network

We are using the tan-sigmoid transfer function in the hidden layer, and a linear output layer. This is the standard network for function approximation [5]. This network has been shown to be a universal approximating network. The used network diagram is

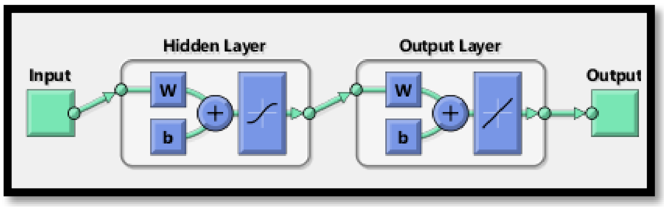

Figure 3. *Regression Neural Network Structure [28].*

2.5.2. Training Algorithms

Here we will examine two algorithms: (trainlm) Levenberg-Marquardt algorithm (default) and (trainbr) Bayesian regularization algorithm.

2.5.3. Hidden Layer size

The number of neurons in the hidden layer will depend on the function to be approximated. This is something that cannot generally be known before training. Levenberg-Marquardt algorithm needs the number of neurons (hidden layer size) to be given to the algorithm. However, the effect of the hidden layer size will be examined in this study by applying a various of hidden layer sizes.

3. Results and Discussion

Here are the results of many options variation on the neural networks:

3.1. Variation of Results Using Same Settings and Sub Datasets Data

Training the same network with the same settings and the same dataset gives different output each run; because of:

a. The randomness of the initial weights and bias at every training run of the neural network.
b. The randomness of dividing dataset into train, validate and test sets.

Table 4 shows the setting used for each run.

Table 4. *Settings of Experiment (Variation of results using same settings and sub datasets data).*

dataset size	50
variables used	AT, V, AP and RH
Hidden layer size (#neurons)	10
Training Function	Levenberg-Marquardt (trainlm)

Results are shown at Table 5; we can observe this behavior of variation in resulted network each run. By looking at the resulted performance (MSE) values for the same test data, we can see that it varies between 0.78 up to 0.94 and the resulted performance (MSE) values varies between 140 and 32.

Table 5. *Variation of the results using same settings and sub datasets data.*

Run	Training Performance	Validation Performance	Training Regression coefficient	Validation Regression coefficient	Test Performance	Test Regression coefficient	Stopping Criteria	#Epochs	Best Epoch
1	2.71	97.78	1	0.8	67.79	0.88	Validation	12	6
2	22.47	41.17	0.98	0.9	65.75	0.89	Validation	11	5
3	21.7	117.89	0.97	0.84	138.66	0.84	Validation	12	6
4	70.82	72.25	0.95	0.89	123.49	0.9	Validation	12	6
5	44.96	70.81	0.94	0.91	140.11	0.78	Validation	10	4
6	16.66	96.47	0.98	0.88	94.97	0.86	Validation	10	4
7	11.87	117.59	0.98	0.94	59.11	0.9	Validation	11	5
8	6.03	106.22	0.99	0.92	32.35	0.94	Validation	16	10
9	15.65	47.82	0.98	0.94	52.78	0.92	Validation	13	7
10	40.16	66.53	0.95	0.86	68.51	0.9	Validation	9	3

3.2. Effect of Different Values of Hidden Layer Size (Number of Neurons)

To examine the effect of the hidden layer size the network is trained with the settings shown at Table 6. Note that each value of hidden layer size is trained for 10 times, and only the best resulted network is considered. This is done to overcome the variation behavior shown in section (3.1).

Table 6. *Settings of Experiment (Effect of different values of hidden layer size).*

dataset size	50
variables used	AT, V, AP and RH
Hidden layer size (#neurons)	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 30, 40, 50, 70, 100]
Training Function	Levenberg-Marquardt (trainlm)

From Table 7 it can be observed that for the same dataset and settings higher size of the hidden layer is not always useful for the network. Comparing the performance (MSE) values for the test dataset, we can notice that at size 100 the worst network obtained, while the best performance obtained with size of 3. The tendency to have better network with smaller hidden layer size indicates that the relation is strongly linear; given that zero hidden layer size means just a linear relationship. Same results are shown in Figure 4.

Table 7. *Effect of different values of hidden layer size.*

Hidden Layer Size	Training Performance	Validation Performance	Training Regression coefficient	Validation Regression coefficient	Test Performance	Test Regression coefficient	Stopping Criteria	#Epochs	Best Epoch
1	34.38	30.09	0.95	0.97	27.21	0.97	Max mu.	23	22
2	27.98	39.5	0.96	0.93	24.76	0.96	Validation	9	3
3	22.1	43.33	0.97	0.94	23.56	0.96	Validation	18	12
4	21.38	67.48	0.97	0.93	24.12	0.96	Validation	13	7
5	18.98	68.08	0.97	0.95	25.59	0.96	Validation	9	3
6	27.78	33.05	0.95	0.98	26.53	0.95	Validation	11	5
7	25.9	67.61	0.97	0.92	46.96	0.95	Validation	9	3
8	12.4	82.42	0.98	0.85	36.1	0.94	Validation	19	13
9	27	58.42	0.96	0.95	38.19	0.95	Validation	10	4
10	16.74	70.42	0.98	0.91	38.33	0.94	Validation	10	4
15	50.09	40.39	0.96	0.93	93.12	0.92	Validation	9	3
20	2.56	113.75	1	0.87	148.28	0.82	Validation	13	7
25	17.9	94.46	0.98	0.92	104.4	0.88	Validation	9	3
30	616.45	460.98	0.85	0.85	817.02	0.85	Validation	7	1
40	0.14	366.38	1	0.76	195.48	0.76	Validation	10	4
50	49.72	365.31	0.96	0.84	551.68	0.76	Validation	6	2
70	1.88	102.07	1	0.89	256.06	0.66	Min gradient	6	2
100	3.6	1589.4	1	0.22	925.24	0.63	Min gradient	10	4

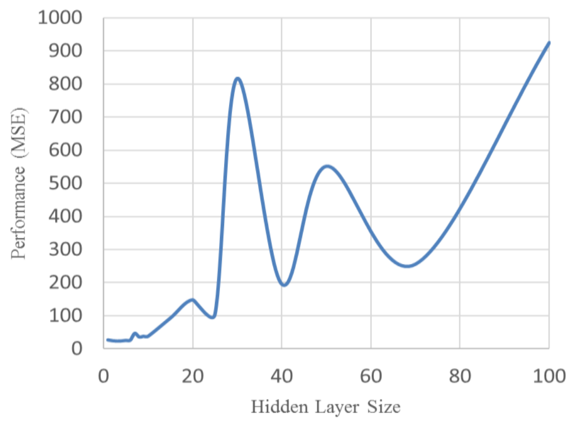

Figure 4. *Effect of different values of hidden layer size on performance (MSE).*

3.3. Effect of Different Train Dataset Size

Here we will examine different sizes of train datasets, which actually train and validate datasets. Settings for this experiment are shown at Table 8.

Table 8. *Settings of Experiment (Effect of different train dataset size).*

Dataset size	[30, 40, 50, 60, 100, 150, 200, 250, 300]
Variables used	AT, V, AP and RH
Hidden layer size (#neurons)	10
Training Function	Levenberg-Marquardt (trainlm)

As shown at Table 8, network is trained for each dataset size for 10 times to overcome the variation in results mentioned in section (3.1). From the results in the Table 9 we can observe that by increasing the train dataset size generally networks improves. Notice that at small dataset size any increase results in improved network performance. In the other hand, by reaching dataset size of 100 only little improvement is obtained by increasing in the dataset size. The same results are in the Figure 5.

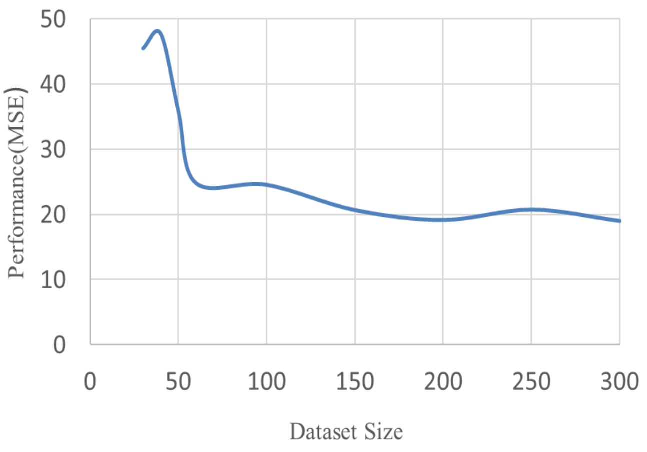

Figure 5. *Effect of different train dataset size on performance (MSE).*

Table 9. *Effect of different train dataset size.*

Dataset Size	Training Performance	Validation Performance	Training Regression coefficient	Validation Regression coefficient	Test Performance	Test Regression coefficient	Stopping Criteria	#Epochs	Best Epoch
30	42.13	68.15	0.94	0.96	45.49	0.92	Validation	8	2
40	15.91	91.37	0.99	0.81	47.73	0.94	Validation	8	2
50	14.54	108.62	0.98	0.85	35.85	0.94	Validation	11	5
60	25.87	15.17	0.96	0.97	24.79	0.96	Validation	11	5
100	21.77	18.57	0.97	0.97	24.55	0.96	Validation	11	5
150	20.06	17.45	0.97	0.97	20.66	0.96	Validation	11	5
200	18.82	15.28	0.97	0.97	19.14	0.97	Validation	13	7
250	19.68	40.41	0.97	0.93	20.73	0.96	Validation	9	3
300	17.4	25.72	0.97	0.96	19	0.97	Validation	10	4

3.4. Effects of Absence and Presence of Each Variable

Each variable has certain effect on the output, some has huge effect (main variables) while others may have little effect if at all. Here we will examine the four variables (AT, V, AP and RH), which makes 15 different combinations. Each is repeated for 10 times as to overcome problem of randomness discussed in section (3.1). These settings are shown at Table 10.

Table 10. *Settings of Experiment (Effects of absence and presence of each variable).*

dataset size	50
variables used	AT and/or V and/or AP and/or RH = 15 different combinations
Hidden layer size (#neurons)	10
Training Function	Levenberg-Marquardt (trainlm)

From the results shown at Table 11 it is obvious that presence of AT has the main effect of the quality of the network; actually, even just using AT alone gives us satisfying model. Introducing the remaining variables to the network increase its quality. In addition, V also has good impact on the model quality. AP and RH have just improving effect on the model. Notice that the best network obtained when using only AT, V and RH. It has the best performance and the best correlation (Regression) when tested on the complete dataset (9538 data point). Imposing AP has generally bad effect on the model quality. Therefore, we can conclude that: introducing some variables may act negatively on the network quality; i.e. not every added variable has improving effect on the model.

Table 11. *Effects of absence and presence of each variable.*

Variables	Training Performance	Validation Performance	Training Regression coefficient	Validation Regression coefficient	Test Performance	Test Regression coefficient	Stopping Criteria	#Epochs	Best Epoch
AT	28.88	25.95	0.95	0.96	29	0.95	Validation	11	5
AT, V	11.93	56.69	0.98	0.96	27.35	0.95	Validation	8	2
AT, V, AP	6.75	65.79	0.99	0.9	35.65	0.94	Validation	11	5
AT, V, AP, RH	21.02	41.98	0.97	0.96	34.31	0.94	Validation	9	3
AT, V, RH	13.18	74.47	0.98	0.94	26.88	0.96	Validation	12	6
AT, AP	21.46	71.32	0.97	0.91	33.22	0.95	Validation	10	4
AT, AP, RH	6.1	71.37	0.99	0.89	31.86	0.95	Validation	12	6
AT, RH	38.72	20.2	0.95	0.95	31.38	0.95	Validation	11	5
V	61.72	39.81	0.91	0.93	69.41	0.88	Validation	10	4
V, AP	18.46	136.06	0.96	0.85	66	0.88	Validation	15	9
V, AP, RH	35.73	88.24	0.94	0.9	69.86	0.87	Validation	12	6
V, RH	70.55	81.22	0.9	0.94	67.49	0.88	Validation	12	6
AP	241.5	215.07	0.48	0.44	222.95	0.49	Validatio	10	4

							n		
AP, RH	167.39	150.46	0.7	0.67	212.23	0.56	Validation	9	3
RH	223.65	281.37	0.55	0.43	281.45	0.35	Validation	9	3

3.5. Effect of Using Different Training Function

In all previous sections, we used Levenberg-Marquardt algorithm (trainlm) function. Here we will examine and compare another famous training function, Bayesian regularization (trainbr), which is an improved algorithm to the former one. Setting are shown at Table 12.

Table 12. *Settings of Experiment (Effect of using different training function).*

dataset size	50
variables used:	AT, V, AP and RH
Hidden layer size (#neurons)	10
Training Function	Levenberg-Marquardt (Trainlm); Bayesian regularization (Trainbr)

From the result Table 13, we can notice that for the same settings and dataset Bayesian regularization (trainbr) is better than Levenberg-Marquardt algorithm (trainlm) function. It is also notable that it has no validation sub set, only training set. Lastly, notice that the number of epochs needed to obtain the network; it is more than 10 times of those needed by Levenberg-Marquardt algorithm (trainlm) function.

Table 13. *Effect of using different training function.*

Used Function	Training Performance	Validation Performance	Training Regression coefficient	Validation Regression coefficient	Test Performance	Test Regression coefficient	Stopping Criteria	#Epochs	Best Epoch
Trainlm	39.79	12.86	0.95	0.98	35.84	0.95	Validation	8	2
Trainbr	29.38	-	0.96	-	25.04	0.96	Max mu.	99	37

3.6. Effect of Normalizing Dataset Before Manipulate It to Network Training

Here dataset is normalized to the range (0.01, 0.99) and the quality of the resulted network is compared to network trained with not normalized dataset, which is normalized by Matlab Neural Network toolbox; which has two options for normalization. The first is the standard normalization to the range of (-1, 1), using the function (mapminmax) which is the default in the toolbox. Secondly, is the normalization to a range with specified mean (typically 0) and standard variation (typically 1), using the function (mapstd). Settings for this experiment are shown at Table 14.

Table 14. *Settings of Experiment (Effect of normalizing dataset).*

dataset size	50
Variables used	AT, V, AP and RH
Normalization method	1. mapminmax normalization (default), 2. mapstd normalization, 3. normalized to the range (0.01, 0.99)
Hidden layer size (#neurons)	10
Training Function	Levenberg-Marquardt algorithm (trainlm)

Table 15. *Effect of normalizing dataset.*

Dataset used	Training Performance	Validation Performance	Training Regression coefficient	Validation Regression coefficient
mapminmax	26.11	20.02	0.96	0.98
mapstd	24.81	26.04	0.96	0.97
Normalized to range (0.01, 0.99)	0.01	0	0.96	0.97

Table 15. *Continued.*

Dataset used	Test Performance	Test Regression coefficient	Stopping Criteria	#Epochs	Best Epoch
mapminmax	30.33	0.96	Validation	8	2
mapstd	29.56	0.96	Validation	8	2
Normalized to range (0.01, 0.99)	0	0.96	Validation	9	3

From the result Table 15 notice that performance values of the normalized data are also normalized, so they are here very small values. By comparing the regression (correlation) and the epochs numbers, we could notice that these three methods are equivalent. Using the not normalized data is easier in reading results and more convenient since Matlab Neural Network toolbox does it for us anyway. Note that network training is run for 10 times for each set of settings as to overcome problem of randomness discussed in section (3.1).

3.7. Comparisons of Target and Resulted Outputs

Here we consider the two groups of resulted outputs.

a. Training & Validation group: outputs resulted from the network for the input data used in training and validation. This group gives a sense of the validity of the model.
b. Test (Complete dataset) group: outputs resulted from the network for the test input data, which in our study is the complete dataset. This group gives success measure for the network.

Comparisons are based on network with the settings and sub dataset size shown at Table 16.

Table 16. *Experiment Settings (Comparisons of target and resulted outputs).*

dataset size	250
Variables used	AT, V, AP and RH
Hidden Layer size (#neurons)	10
Training Function	Levenberg-Marquardt (Trainlm)

General comparison is presented visually in Figure 6 and Figure 7. It can be observed that the results from the network for input data used in train and validation are closer to their target outputs. Whereas the outputs resulted from complete dataset test are more deviated from their target outputs. That becomes clear when comparing the performance (MSE) of each output group as shown in Table 17.

Furthermore, comparison of error for the two groups is also shown in Table 17, Figure 8 and Figure 9. From Table 17, it is noticed the close values of error and standard deviation between the two studied groups.

Figure 9 shows error in results for the train and validation group and for the test group (complete dataset) along with the amount of instances at the group with the same error value. By first look we find that error distributes among each groups' instances as normal distribution. When compared to each other, it can be seen that they both have most error in range between -10 and +10, which agrees with the statistical fact that says 99.73% of data will fall in the range of 6 sigma (6σ), i.e. range of (μ-3σ, μ+3σ) [30-32], using data from Table 17 this range in our case is approximately (-12.5, 12.5). In addition, we can notice that the error range in test results (-44, +21) is double of the error range of thee train and validation group (-25, +11). The doubled error range resulted from doubled minimum and maximum errors in the two groups. Note that from Figure 7, the model tends to overestimate output at some points, since the far negative range is wider than the right side range. But by looking at Table 17, mean errors which are positive values near zero, so, it can be concluded these far negative error points are just few points and there are more points with positive errors.

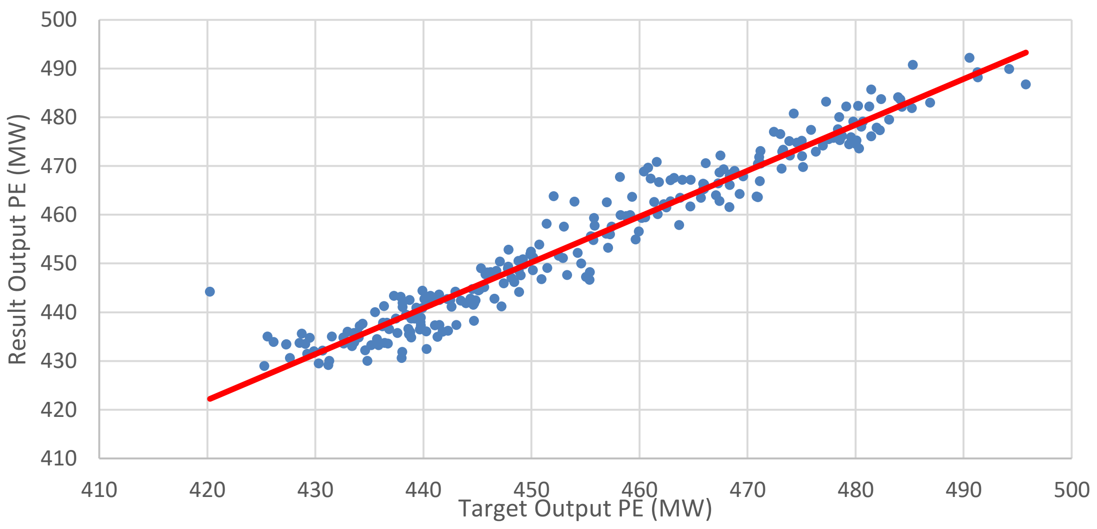

Figure 6. *Target (MW) vs. Result Output PE (MW) (Training and Validation).*

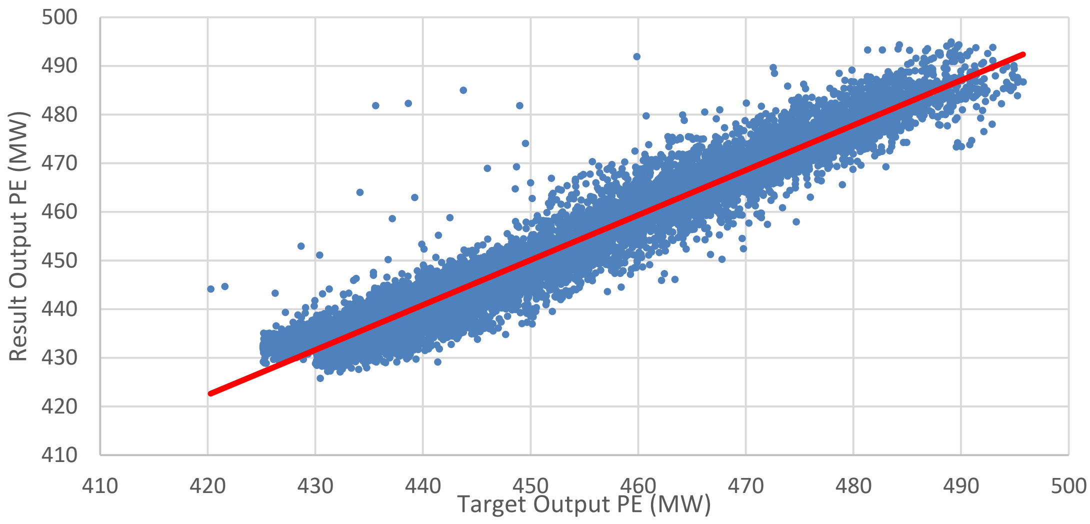

Figure 7. *Target (MW) vs. Result Output PE (MW) (Complete Database).*

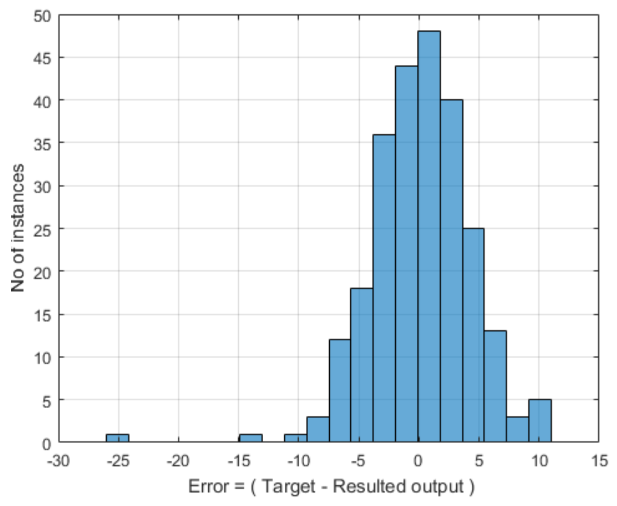

Figure 8. *Output power (PE) Error (MW) Vs. No of Instances (Training and Validation).*

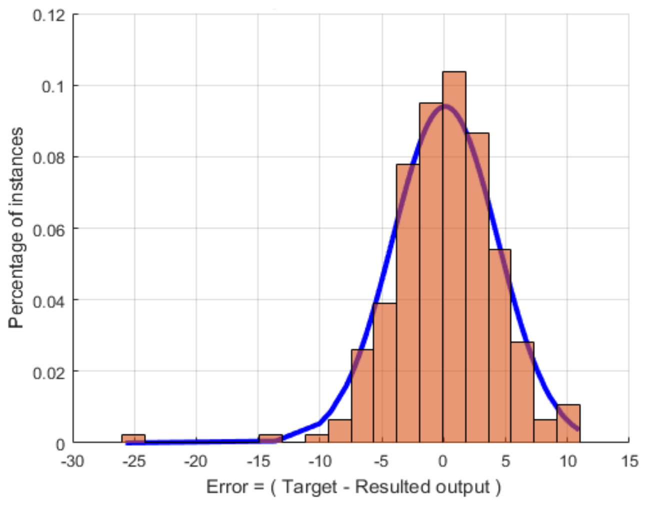

Figure 10. *Output power (PE) Error (MW) vs. percentage of instances (Training and Validation).*

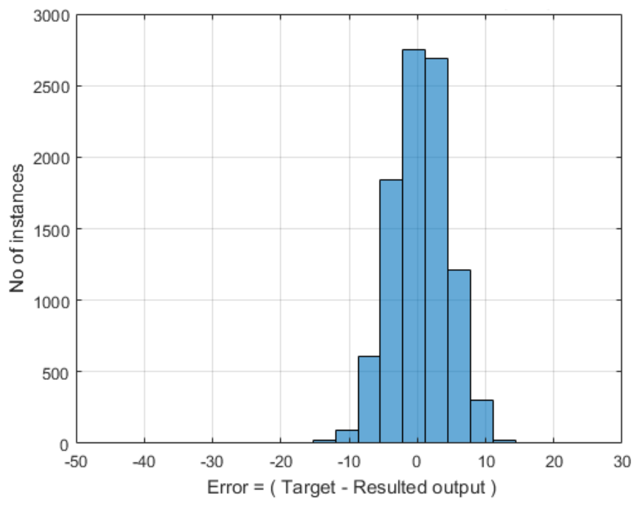

Figure 9. *Output power (PE) Error (MW) Vs. No of Instances (Complete Database).*

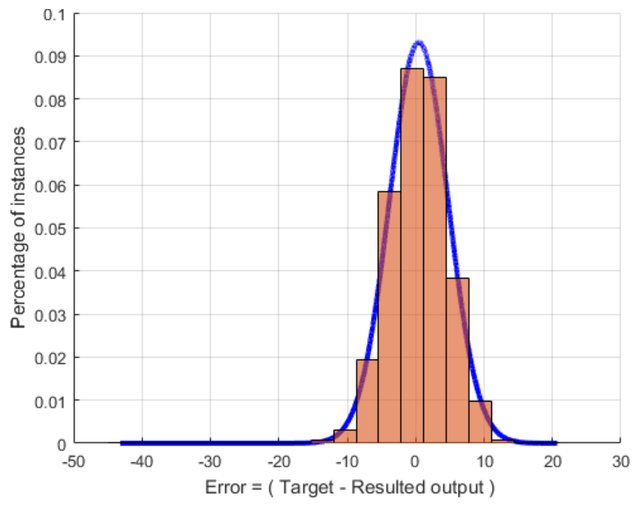

Figure 11. *Output power (PE) Error (MW) vs. percentage of instances (Complete dataset).*

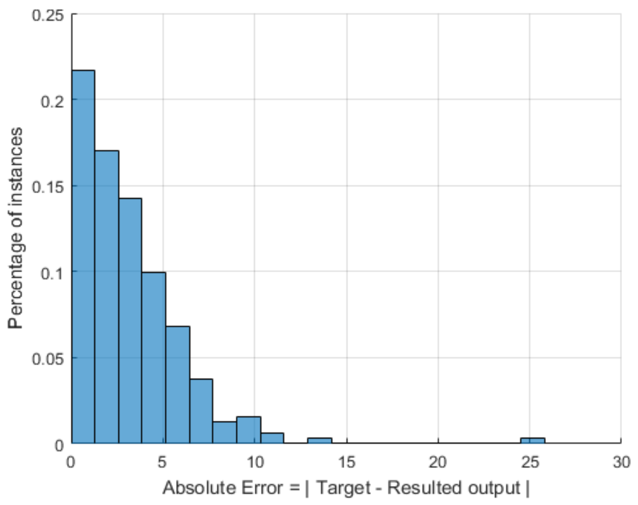

Figure 12. *Absolute Output power (PE) Error (MW) vs. percentage of Instances (Training and Validation).*

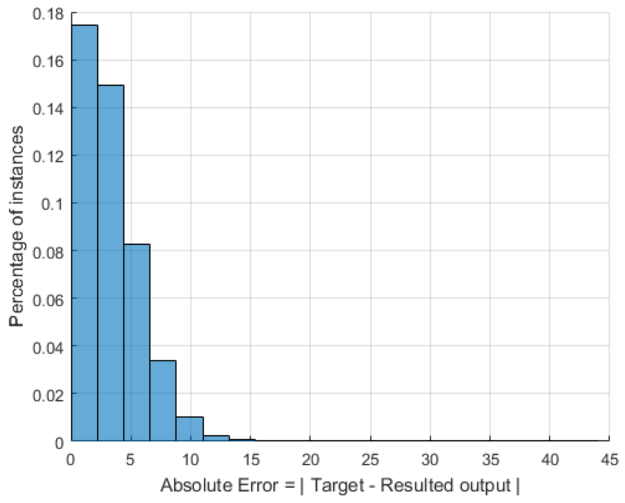

Figure 13. *Absolute Output power (PE) Error (MW) vs. percentage of Instances (Complete dataset).*

To compare the amount of instances vs. error between the two groups, it is more convenient to compare the normalized amount of instances, i.e. percentage of the group. This is shown in Figure 10 and Figure 11 along with lines of normal distribution of properties of mean and standard deviation shown in Table 17. The two charts are very similar to each other.

If the error sign (Positive or Negative) is to be neglected, as we want to describe how close the group results to its target values, we can make the same chart but with absolute values. This is presented in Figure 12 and Figure 13.

At this case, train and validation group has more percentage of its instances closer to zero mean error. It provides us with the same info we extracted from Figure 6 and Figure 7 that test results are more deviated from their target.

As experiment, 20 data points are selected randomly and tested, results and errors are shown in Figure 14 and Figure 15. They show that our findings of range of error between (-12.5, +12.5) hold nicely for these randomly selected points.

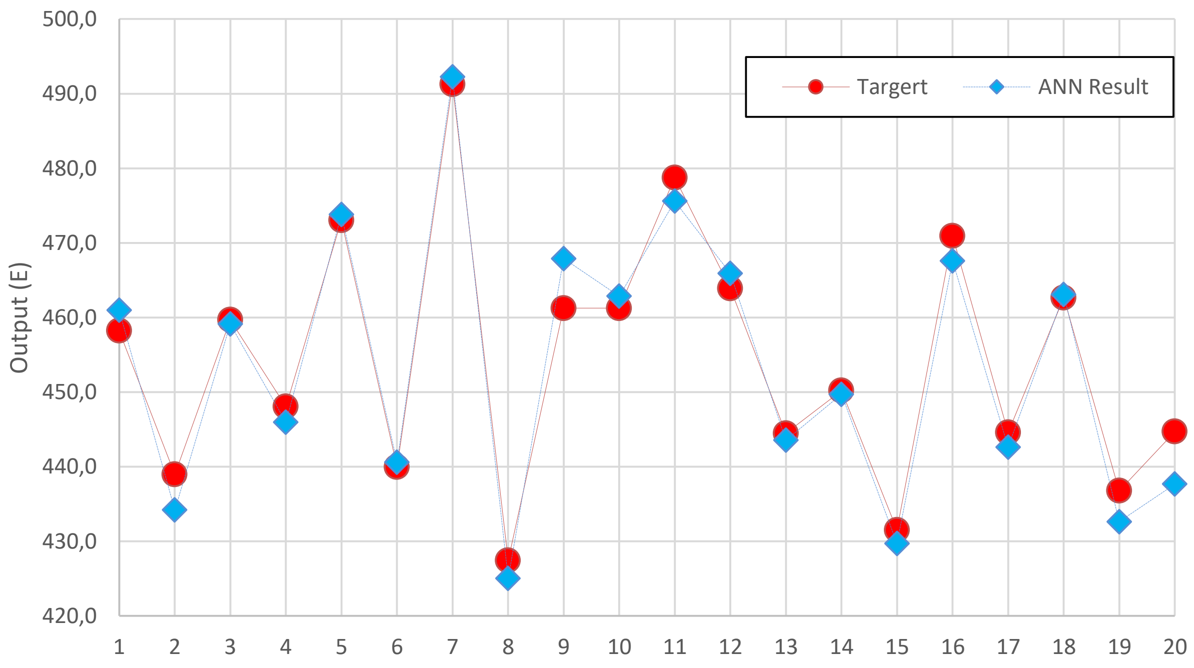

Figure 14. *Predicted Output Power (MW) and error (MW) at 20 random data points from test results.*

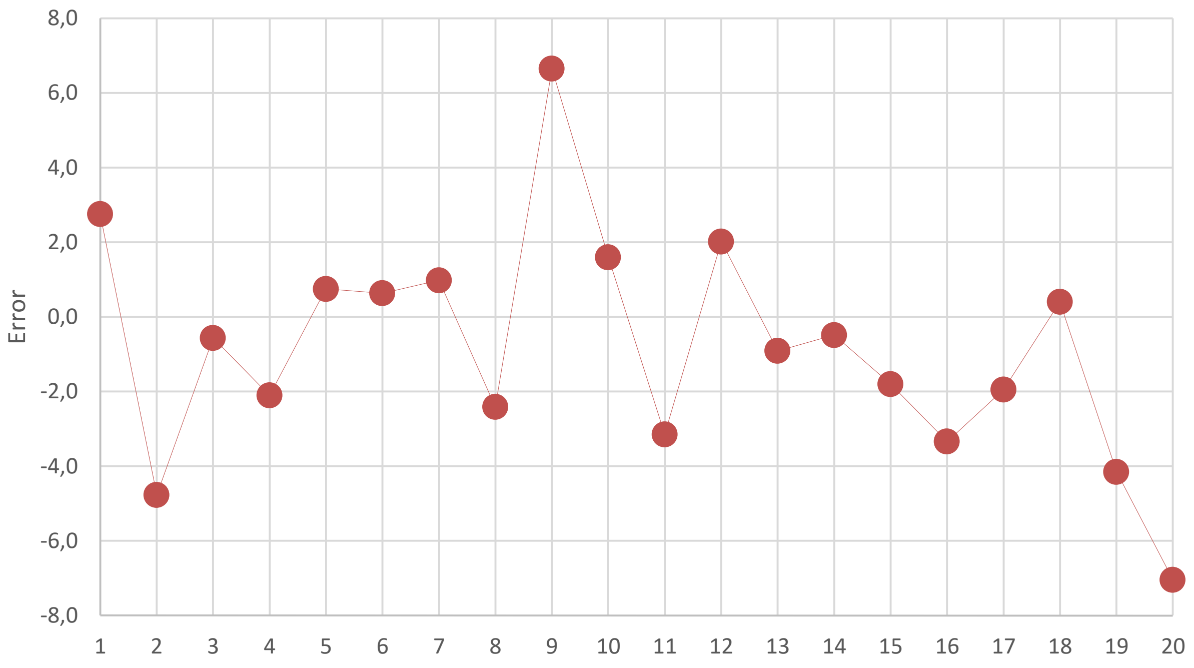

Figure 15. *Prediction error (MW) at 20 random data points from test results.*

Table 17. *Error in result output.*

Group	Train & Validation	Test (Complete Dataset)
Group size	250	9568
Performance MSE $(MW)^2$	17.942	18.682
Mean error μ (MW)	0.1108	0.4564
Standard deviation of error σ (MW)	4.2429	4.2825
Min error (Negative) (MW)	-25.7167	-43.1750
Max error (Positive) (MW)	10.9414	20.6876

4. Conclusion

Regression artificial neural networks (ANN) is used to model electrical output power (EP) of combined cycle power plant based on four inputs. Data are collected from published work freely available online. MATLAB neural networks toolbox is used to program the ANN model. The ANN model is applied and studied through experimenting various settings effects on the neural network performance. Total seven experiments are applied.

Results show the randomness of the ANN model performance for each time it trained, this is because randomness of the initial values of weights and bias. It is also observed that increasing in number of neurons at the hidden layer does not necessary lead to increased quality of the model; in fact, number of neurons has an oscillating effect on the model performance. Increasing dataset size (more data points for the same variables) provides better networks for some extend. Increasing the number of input variables does not always leads to better network quality; some variables when introduced reduce the quality of the model, others increase it. It has to be studied through correlation between variables themselves and between variables and output. In addition, different training functions are compared for the same setting and dataset; in this work Bayesian regularization performed better than Levenberg-Marquardt algorithm. Dataset normalization methods provided by the toolbox are also experimented.

Lastly, results are compared with target values of output for the train and validation group and for the test group, which is the complete dataset group. Comparison shows that results are very close to target outputs for both groups. In addition, it shows the normal distribution of error among each group with mean value of zero. The standard deviations of the error at the two groups are almost equal.

References

[1] U. Kesgin and H. Heperkan, “Simulation of thermodynamic systems using soft computing techniques,” *Int. J. Energy Res.*, vol. 29, no. 7, pp. 581–611, 2005.

[2] A. Dehghani Samani, “Combined cycle power plant with indirect dry cooling tower forecasting using artificial neural network,” *Decis. Sci. Lett.*, vol. 7, no. 2, pp. 131–142, 2018.

[3] P. R. Norvig and S. A. Intelligence, “A modern approach,”

Manuf. Eng., vol. 74, no. 3, pp. 111–113, 1995.

[4] E. Rich and K. Knight, "Artificial intelligence," *McGraw-Hill, New*, 1991.

[5] M. T. Hagan, H. B. Demuth, and M. H. Beale, "Orlando De Jesus," *Neural Netw. Des. 2nd Ed. Cengage Learn.*, 2014.

[6] D. Jahed Armaghani, M. F. Mohd Amin, S. Yagiz, R. S. Faradonbeh, and R. A. Abdullah, "Prediction of the uniaxial compressive strength of sandstone using various modeling techniques," *Int. J. Rock Mech. Min. Sci.*, vol. 85, pp. 174–186, May 2016.

[7] H. Moayedi and D. Jahed Armaghani, "Optimizing an ANN model with ICA for estimating bearing capacity of driven pile in cohesionless soil," *Eng. Comput.*, vol. 34, no. 2, pp. 347–356, Apr. 2018.

[8] M. Khandelwal *et al.*, "Implementing an ANN model optimized by genetic algorithm for estimating cohesion of limestone samples," *Eng. Comput.*, vol. 34, no. 2, pp. 307–317, Apr. 2018.

[9] A. Baghban, F. Pourfayaz, M. H. Ahmadi, A. Kasaeian, S. M. Pourkiaei, and G. Lorenzini, "Connectionist intelligent model estimates of convective heat transfer coefficient of nanofluids in circular cross-sectional channels," *J. Therm. Anal. Calorim.*, vol. 132, no. 2, pp. 1–27, May 2017.

[10] H. Khosravani, M. Castilla, M. Berenguel, A. Ruano, and P. Ferreira, "A Comparison of Energy Consumption Prediction Models Based on Neural Networks of a Bioclimatic Building," *Energies*, vol. 9, no. 1, p. 57, Jan. 2016.

[11] A. S. Jihad and M. Tahiri, "Forecasting the heating and cooling load of residential buildings by using a learning algorithm 'gradient descent', Morocco," *Case Stud. Therm. Eng.*, vol. 12, pp. 85–93, Sep. 2018.

[12] S. Sholahudin and H. Han, "Heating Load Predictions using The Static Neural Networks Method," *Int. J. Technol.*, vol. 6, no. 6, p. 946, Dec. 2015.

[13] S. M. A. N. R. Abadi, M. Mehrabi, and J. P. Meyer, "Prediction and optimization of condensation heat transfer coefficients and pressure drops of R134a inside an inclined smooth tube," *Int. J. Heat Mass Transf.*, vol. 124, pp. 953–966, Sep. 2018.

[14] C. Wan, Z. Xu, P. Pinson, Z. Y. Dong, and K. P. Wong, "Optimal prediction intervals of wind power generation," *IEEE Trans. Power Syst.*, vol. 29, no. 3, pp. 1166–1174, May 2014.

[15] F. Bizzarri, M. Bongiorno, A. Brambilla, G. Gruosso, and G. S. Gajani, "Model of photovoltaic power plants for performance analysis and production forecast," *IEEE Trans. Sustain. Energy*, vol. 4, no. 2, pp. 278–285, Apr. 2013.

[16] T. Mahmoud, Z. Y. Dong, and J. Ma, "An advanced approach for optimal wind power generation prediction intervals by using self-adaptive evolutionary extreme learning machine," *Renew. Energy*, vol. 126, pp. 254–269, Oct. 2018.

[17] A. Khosravi, S. Nahavandi, and D. Creighton, "Prediction intervals for short-term wind farm power generation forecasts," *IEEE Trans. Sustain. Energy*, vol. 4, no. 3, pp. 602–610, Jul. 2013.

[18] M. J. Embrechts, A. L. Schweizerhof, M. Bushman, and M. H. Sabatella, "Neural Network Modeling of Turbofan Parameters," in *Volume 4: Manufacturing Materials and Metallurgy; Ceramics; Structures and Dynamics; Controls, Diagnostics and Instrumentation; Education*, 2000, p. V004T04A008.

[19] C. Boccaletti, G. Cerri, and B. Seyedan, "A Neural Network Simulator of a Gas Turbine With a Waste Heat Recovery Section," in *Journal of Engineering for Gas Turbines and Power*, 2001, vol. 123, no. 2, p. 371.

[20] R. Bettocchi, P. Spina, and G. Torella, "Gas Turbine Health Indices Determination by Using Neural Networks," in *ASME Turbo Expo*, 2002, pp. 1–7.

[21] H. H. Erdem and S. H. Sevilgen, "Case study: Effect of ambient temperature on the electricity production and fuel consumption of a simple cycle gas turbine in Turkey," *Appl. Therm. Eng.*, vol. 26, no. 2–3, pp. 320–326, Feb. 2006.

[22] I. Ceylan, O. Erkaymaz, E. Gedik, and A. E. Gurel, "The prediction of photovoltaic module temperature with artificial neural networks," 2014.

[23] P. Tüfekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *Int. J. Electr. Power Energy Syst.*, vol. 60, pp. 126–140, 2014.

[24] L. X. Niu and X. J. Liu, "Multivariable generalized predictive scheme for gas turbine control in combined cycle power plant," in *2008 IEEE Conference on Cybernetics and Intelligent Systems*, 2008, pp. 791–796.

[25] H. Kaya, P. Tüfekci, and S. F. Gürgen, "Local and Global Learning Methods for Predicting Power of a Combined Gas & Steam Turbine," in *International Conference on Emerging Trends in Computer and Electronics Engineering (ICETCEE 2012)*, 2012, pp. 13–18.

[26] V. Ramireddy, "An Overview of Combined Cycle Power Plant," 2015. [Online]. Available: http://electricalengineering-portal.com/an-overview-of-combined-cycle-power-plant.

[27] M. H. B. M. T. H. H. B. Demuth, "Kalman Filtering and Neural Networks," *MathWorks*, 2001.

[28] H. Demuth, "Neural Network Toolbox," *Networks*, vol. 24, no. 1. pp. 1–8, 2002.

[29] R. POWER, "Combined Cycle Power Plant CCPP." 2011.

[30] Wikipedia, "68–95–99.7 Rule," *Wikipedia*, 2015. [Online]. Available: https://en.wikipedia.org/wiki/68–95–99.7_rule.

[31] "Six_Sigma." [Online]. Available: https://en.wikipedia.org/wiki/Six_Sigma.

[32] L. J. Kazmier, *Schaum's outline of theory and problems of business statistics*, 4th ed. McGraw Hill Professional, 1996.