Using Neural Network to Predict Diffuse Fraction k_d (Ratio of the Diffuse-to-Global Solar Radiation)

Oscar Alberto Santos Muñoz

Abstract—In this paper, an artificial neural network was used to predict values of the diffuse fraction, using global radiation, date and time as input parameters. The network did not show perfect accuracy, however it made very close predictions with a mean absolute error of 0.006887367 and mean relative error of 1.525050225%, because of the way the neural network was created. The problem was solved by classification instead of regression with Feedforward Neural Network architecture composed of 3 hidden layers with 128 neurons each. These results can be used as inputs to other neural networks, and thus contribute to and create more accurate neural networks with a higher number of inputs. And that they can improve certain applications and solar devices.

Index Terms—Artificial intelligence, computer science, classification model, data analysis, deep learning, diffuse solar radiation, diffuse fraction, machine learning, meteorological parameters, neural networks, predictive analytics, solar irradiance, time series.

1 Introduction

NEURAL networks have been a focus of interest since they emerged as a solution to time series prediction problems. In addition, organizations are looking for more precision in their processes. Certainly, there are measurement errors, inconsistencies and bad approaches due to small measurement issues, but a good model is achieved from high quality data [1], [2]. Past work has shown that accurate predictions are achieved through past data from the model or function to be predicted [3], [4].

There are research centers that are dedicated to collect data, some of them measure the radiation from the sun each hour of the day. Due to the high costs that this generates, there is not an accurate record for all areas. Radiation varies according to the conditions of each site and therefore measurements are limited, resulting in poor approximations or lack of data. Over time, models have been created to correlate variables [5] such as Global Horizontal Irradiance (GHI) and Diffuse Horizontal Irradiance (DHI).

Once, the correlations have being obtained, is easier to create models over the collected data rather than measure again with high costs. These relations help to establish solar models for certain applications such as the placement of photovoltaic panels and sun tracking. The simulations and performance on solar devices can be improved as well [5], knowing for example the diffuse fraction (K_d).

An artificial neural network (ANN) is a way of solving non-linear relationships in Multi-Inputs-Multi-Outputs (MIMO) systems. Neural networks are interconnections between neurons that are connected to each other and these connections have assigned weights and biases [6]. The branches that come out of a biological neuron are called dendrites, which receive the afferences from other neurons or from receptor cells [7]. The same occurs in an ANN, the hidden layer neurons receive information from the dendrites in the input layer and produce a corresponding output if the sum is greater than threshold value [6], this value is the bias. This would be the equivalent of the cell body of the biological neuron, which contains the nucleus and functions as the metabolic center of the cell. And finally, the axon of the cell is the one that specializes in conducting impulses from the cell body to another neuron or effector cell [7]. This is the same as the outputs of each neuron in an ANN. Once the calculations with the weights and biases have been made the outputs go to other neurons or the final response of the network.

This analogy with the biological neuron can be represented as shown in Fig. 1(a), where x_i are the nerve stimulation received at the brain from a receptor cell and y is the response to that stimulus. Fig. 1(b) shows an artificial neural network process, where x_i are the inputs, w_i are the weights of each connection and b is the bias or threshold value.

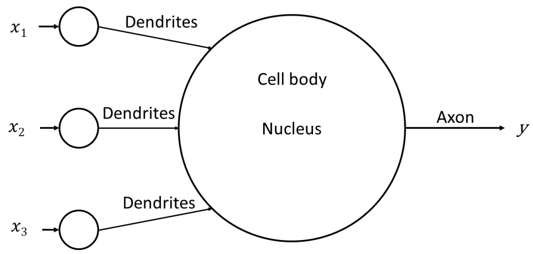
2 Data

For this article, the weather conditions for the city of Nagaoka, Niigata, Japan, were used as a basis. And the database consists of 20 excel files, with the 8760 hours of the year. The database contains geographical information about the city of Nagaoka, Niigata, Japan. In the Excel files it is possible to find data of the Direct Normal Irradiation/Irradiance (DNI), (DHI), (GHI), zenith, azimuth, Direct Normal Irradiation/Irradiance (DNI) with a tilt angle, (DHI) with a tilt angle, Global Tilted Irradiation/Irradiance (GTI) and the tilt angle in degrees.

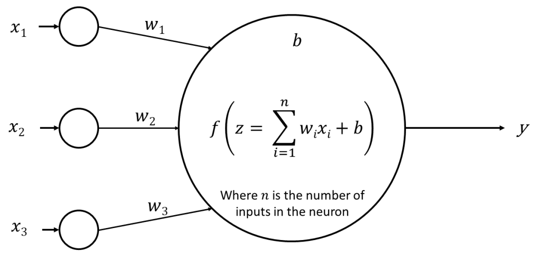
Date of current version January 28, 2019.

This work was supported by grants from the Nagaoka University of Technology by JASSO Program and the University of Guanajuato.

O. Santos is with the Energy Engineering Laboratory, Niigata 940-2188, Japan, and also with University of Guanajuato, Guanajuato, 36000, Mexico (e-mail: oa.santosmunoz@ugto.mx)



(a) Representation of the parts of a biological neuron



(b) Representation of an artificial network process

Fig. 1. Analogy between a biological neuron and an artificial neuron.

2.1 Understanding the behavior

First, it was considered prudent to conduct an analysis on the first year's data. This is in order to better understand our database. Although every year shows an irradiance with values typical of the conditions of the date, all assume an average behavior over time. To appreciate this, it was considered only the first year to understand the behavior of irradiance.

2.1.1 First year analysis

The first thing that was done was to create a python list that contains information needed to plot the whole year. To do this, a list was created that repeats the number "1" 24 times, then 24 times the number "2", then 24 times the number "3" and so on until "365". This indicates that each 24 elements in the list belong to one day of the year, since in the database the information is given by hour and it is required to section by day to have a record of the date.

In Fig. 2 it can be seen a plot of the days of the year vs irradiance. It includes 2 types of radiation, GHI and DHI, which are given by the database.

The plot of the diffuse fraction had also been plotted; however, it was difficult to analyze its behavior given the high amount of data and the peaks of each day. To better appreciate and understand the data it was decided to take the first day analysis instead of the first year.

2.1.2 First day analysis

As before, a Python list was created for which a 24-hour account is repeated. Since item 25 in the database corresponds to hour 1 of day 2, the list stores values from 1 to 24 every 24 items. This way, the database can be sectioned in hours of the day according to the date without having to handle hours of the whole year.

Similarly, it was considered prudent to visualize the radiation again over the course of a day. This occasion is shown in Fig. 3 both cases; GHI and DHI vs hours of the day and Diffuse fraction vs hours of the day.

2.1.3 Deep analysis on diffuse fraction

During the analysis process, the question arose whether radiation could be used as the only input, as the only training parameter. However, to visualize whether a linear regression could be used, the diffuse fraction vs GHI was plotted and DHI as a transparency parameter in that scatter plot type. When analyzing the results, it is clear that a linear regression cannot be used during the year nor during the day and therefore more input parameters must be taken for this prediction model. The results are shown in Fig. 4.

2.2 Data format

As mentioned above, a database consisting of 20 years of accurate information on the city of Nagaoka was used. But in order to use all this information as an input to our network some adjustments had to be made. One is the rearrangement of data, other is the scaling that will be given to the wide range of values in the data and finally split into training, validation and test data.

2.2.1 Rearrange of data

To do this, each year of data that had been stored in an individual list was added to a single Python list. The list is 175200 items long, that is 8760 hours of the year multiplied by 20 years. Once the list with the 175200 elements was taken, the lists of days of the year and hours of the year that had been created to graph the first year and day were multiplied by 20 so that they contained 175200 elements as well. After that, numpy arrays were handled that facilitated the operations to create the input matrix. This matrix contains the GHI data, date and time.

In addition, another of the training parameters is the diffuse fraction, which was calculated directly in each Excel file for the 20 years and handled through the same process of storing the 20 years in a single Python list. For this case it was not necessary to add date and time in a matrix, simply the diffuse fraction will be used, since the input matrix already contains such parameters.

The first column represents GHI, the second is the date of the year, therefore, that column indicates the day of the year that represents that row, the third is the time of the day that the second column indicates, so, for example, during the hour 1 and 2 of day 1 there is a value of 0 for GHI, this because they are the measurements of the night; 12:00 am and 1:00 am respectively. Row 24 is the 24th hour and it is the same case as the night; 11:00 pm. Row 25 as mentioned in previous sections, hour 25 represents hour 1 of day 2, so it is 12:00 am, but this time of day 2 and again it is night and the value for GHI is 0. Now the example where the value for GHI is not 0, hour 13 of day 2 which corresponds to 2:00 pm of day 2 contains a value of 952.623046875.

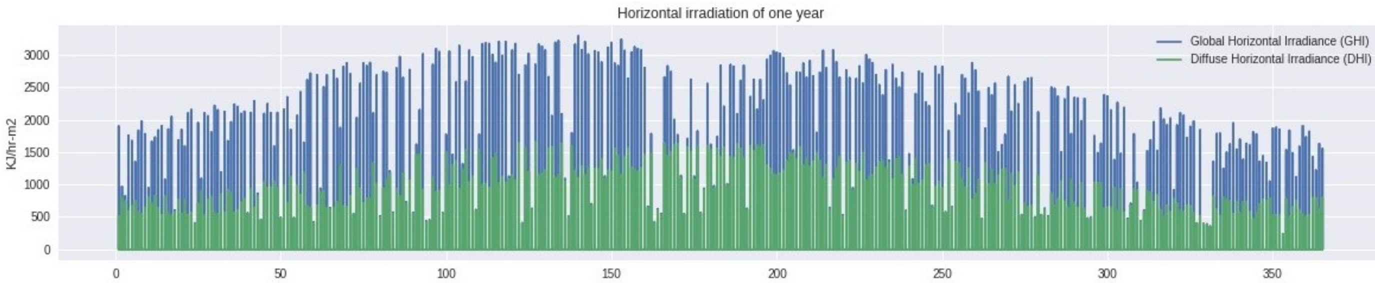


Fig. 2. General radiation behaviour during the first year of analysis.

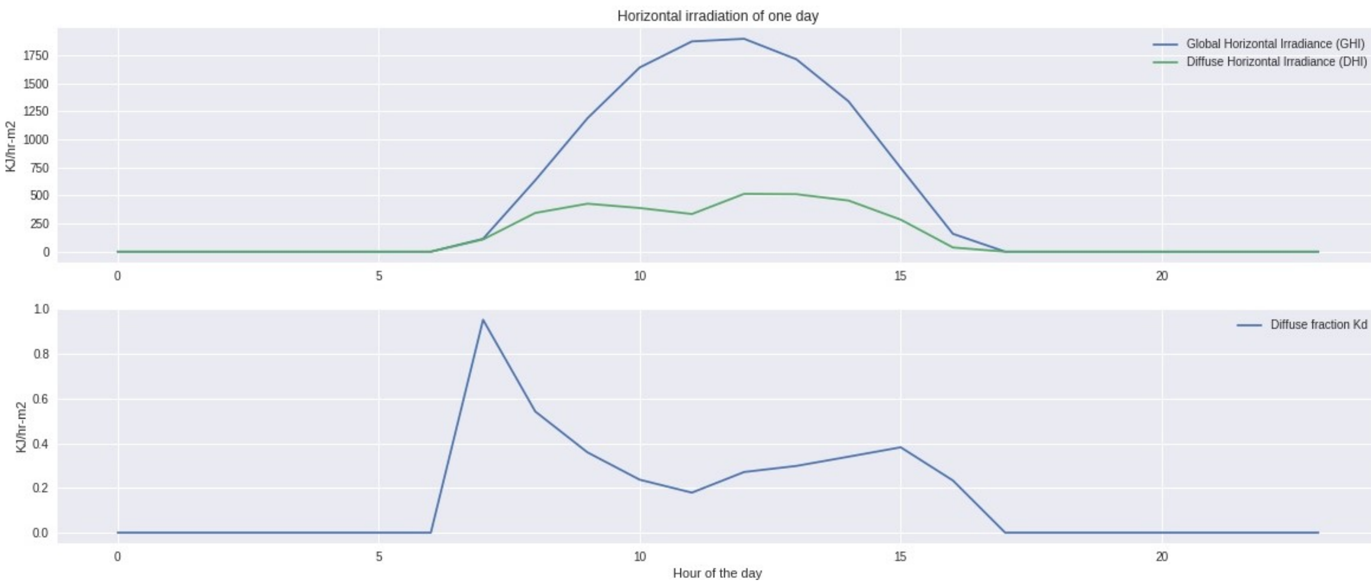


Fig. 3. General radiation behaviour during the first day of analysis.

The following is a representative example for the input matrix and the array that stores the diffuse fraction.

$$InputMatrix = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 2 \\ \vdots & \vdots & \vdots \\ 0 & 1 & 24 \\ 0 & 2 & 1 \\ \vdots & \vdots & \vdots \\ 952.623046875 & 2 & 13 \\ \vdots & \vdots & \vdots \\ 0 & 365 & 24 \end{pmatrix}$$

$$k_d array = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0.810689492212087 \\ \vdots \\ 0 \end{pmatrix}$$

In the case of the array it works in the same way, element 37 represents the 13th hour of the second day, and since there is a radiation value for that row, there is a value for the diffuse fraction which is 0.810689492212087 for this row.

2.2.2 Data standardization

Standardization allow us to have better results on our model as well as being trained faster. Standardizing the data involves rescaling the distribution of values so that the mean of observed values is 0 and the standard deviation is 1. This is useful when your data has input values with differing scales. A value is standardized as follows:

$$z = \frac{(x - \mu)}{\sigma} \tag{1}$$

2.2.3 Splitting the data

To prevent overfitting is better to split in training data, validation data and test data. It was decided to separate into 15 years of training, 2 years for validation and 3 years to test the network with data it did not see during the training. This separation may be changed in future work to try to train the network with more or less training data, more validation data or less test data. This could be easily changed with the library Sklearn available in python.

After separation, there are now 3 matrices and 3 arrays, a matrix of 131400 rows (15 times 8760 hours of the year) x

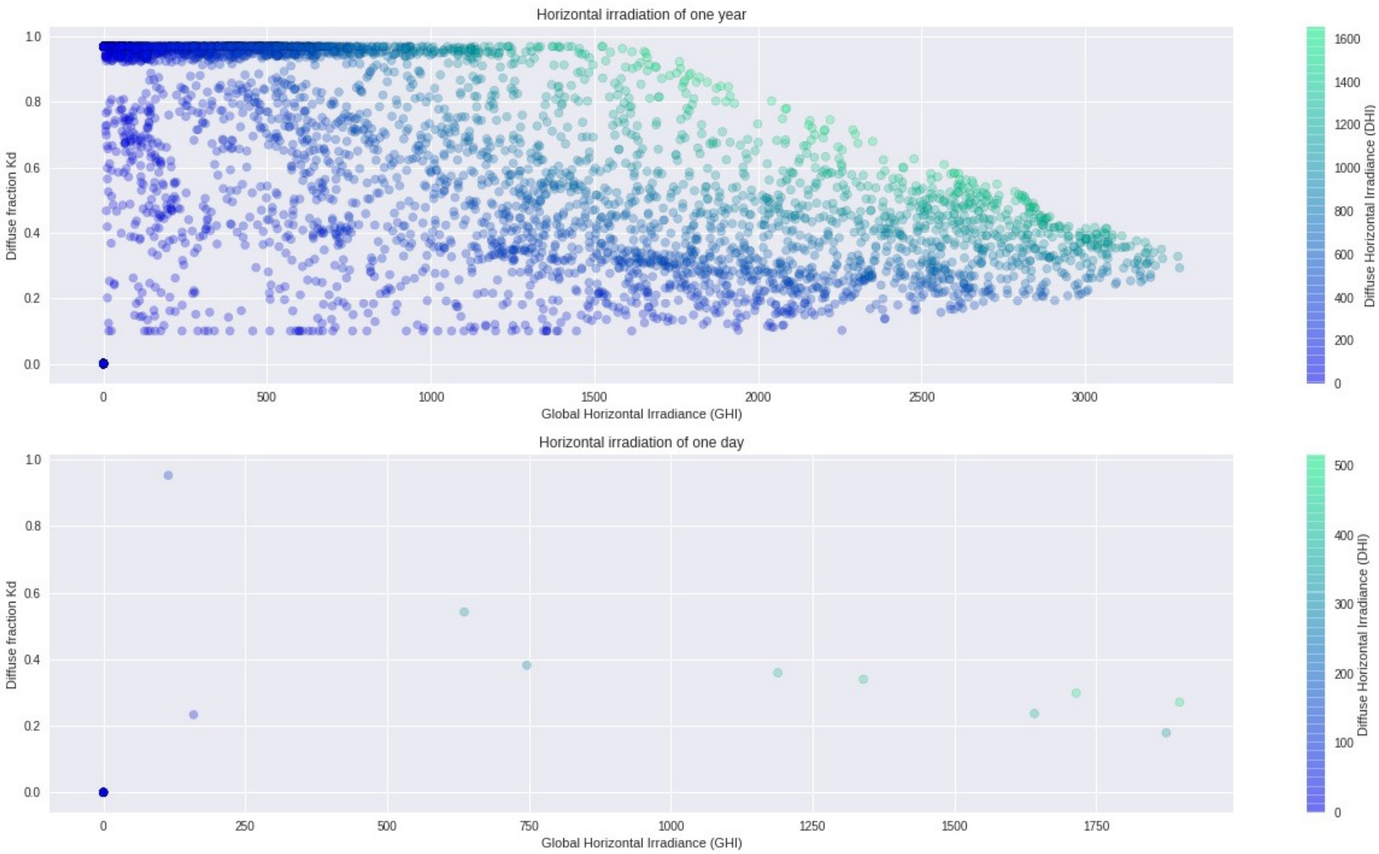


Fig. 4. General behaviour of the diffuse fraction during the first year and first day of analysis.

3 columns (GHI, Date and time), a matrix of 17520 rows (2 times 8760) x 3 columns and a matrix of 26280 rows (3 times 8760) x 3 columns. The sum of all the rows of these matrices is 175200 (20 times 8760) which are the rows that the previous matrix had. For the array is the same but without the date and time columns.

3 Architecture of the Neural Network

A Feedforward neural network was chosen as the ANN type. The architecture of the ANN can be seen in Fig. 5.

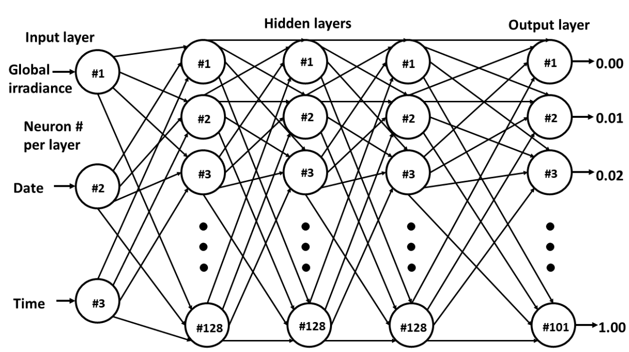


Fig. 5. Architecture of the ANN.

The activation functions for the 3 hidden layers is ReLU and for the output layer is Softmax, this will create an arrangement of probabilities for each prediction that the ANN makes. The 3 hidden layers have 128 neurons each, and the reason why there are 101 neurons at the output is due that the problem will be solved by classification instead of regression, this means that each neuron will have a value assigned to it for each class and there are 101 classes. In Fig. 5 it can be observed that the outputs are values between 0 and 1 with increments of 0.01 between each output.

4 Prediction Algorithm

4.1 Optimizer

Adam optimizer was chosen because of its benefits and it suits perfectly to our case due to is computationally efficient, has little memory requirements and is well suited for problems that are large in terms of data [8], since the database have a lot of information this requires a lot of memory.

4.2 Loss

The loss sparse_categorical_crossentropy was used. And the values of the diffuse fraction array were transformed to integers to be able to use this loss and make multi-class predictions with classification. The outputs will be in softmax activation function that was assigned to the output layer.

4.3 Metrics

In this case the most advisable is to evaluate the ANN with the accuracy it has with the training data and with the validation data. Therefore, accuracy was selected.

4.4 Training parameters

To train the ANN, the hyper-params were changed a few times, finally a batch_size of 128 was selected with 750 epochs. This took about one hour and a half to train using the GPU from Google Colaboratory environment.

5 Results and Discussion

The accuracy value that the ANN had was 78.09% during the training, and it could be thought that it is a not so accurate result, however, as each class has 0.01 increase from the previous one it turns out that very close results are obtained.

The analysis for the accuracy values with the training and validation data is shown in Fig. 6(a). It is evident that the value increases as the ANN learns with each epoch it performs. Each iteration the ANN trains with an amount of data equal to the input matrix divided by the batch_size. This is 131400 elements (The 15 years that were separated in a previous section) divided by 128 which was the number taken as the value of batch_size. Different configurations could be tried in future works, however, in this report it was considered that with the proposed architecture and the proposed hyper-params good results were obtained given the short time of training.

Fig. 6(b) shows the analysis for the loss and validation loss, as it can be seen, the ANN is reducing the value as the network is training and it is just what we are looking for.

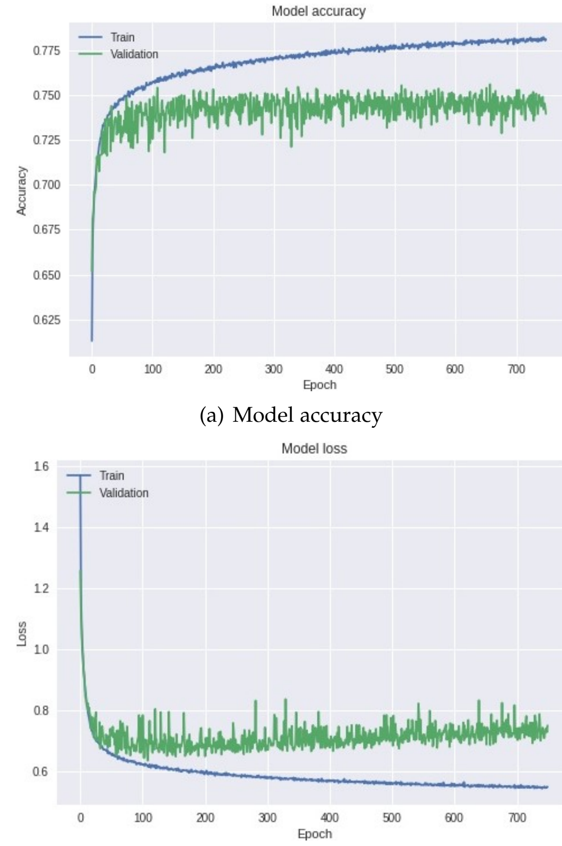
Accuracy was also evaluated with validation data and test data. Table 1 shows the results that were obtained with the model predictions.

The result of the test data is lower than the validation data since the validation data was seen by the ANN during the training, therefore, you will have a lower percentage in data that the network has not seen.

To better appreciate the results, the first week of predictions was plotted along with the actual value of the diffuse fraction. Comparisons for the predictions with the validation data are shown in Fig. 7(a) and comparisons with the test data are shown in Fig. 7(b).

As shown in Fig. 7(a) and Fig. 7(b) there is a very close prediction even when it fails. That is, if the actual value does not match, the predicted value is mostly of some class nearby, i.e. the output that had high weight values in its connections is a neuron of a class nearby.

First, only 11 outputs had been established, which were equally in the range of 0 to 1 but with 0.1 increments between each output neuron. This resulted in much greater accuracy in choosing the right class. However, having a low decimal accuracy resulted in more distant predictions of the actual value. For that reason, it was decided to take 101 classes and in that way 101 outputs could be used with 0.01 increments as mentioned in a previous section.



(a) Model accuracy

(b) Model loss

Fig. 6. Analysis and comparison between training and validation data metrics.

To prove this, what was done was to evaluate the results mathematically. The results were exported to a csv file. The error was calculated using the following equations:

$$AbsoluteError = |Real - Predicted| \tag{2}$$

$$RelativeError = \frac{|Real - Predicted|}{Real} * 100 \tag{3}$$

Once the calculation was made for each prediction, the mean was calculated. The mean result for each value of equation (2) and (3) of all predictions is shown in Table 2.

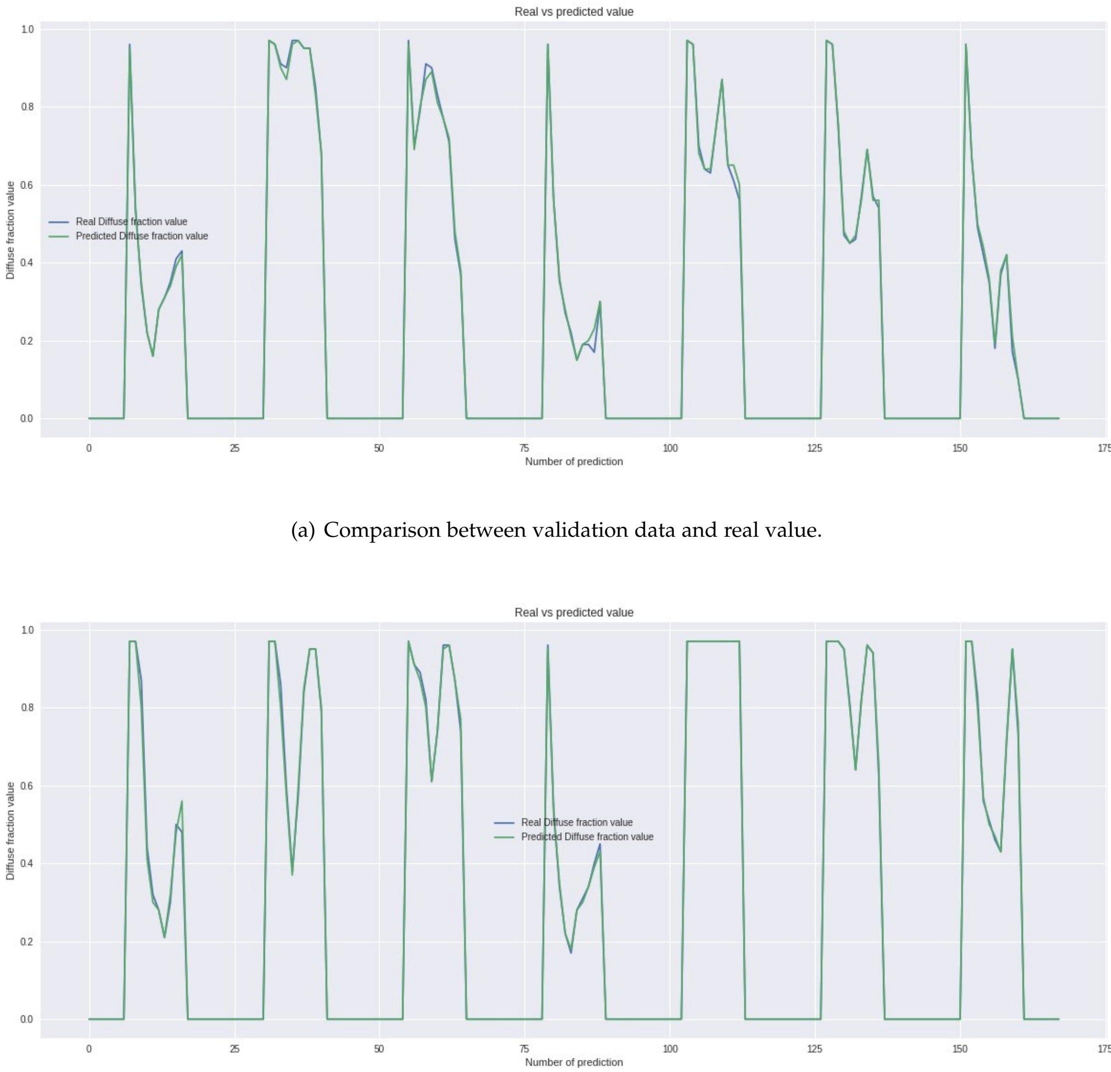
When errors accumulate during training there is a high impact on neural networks based on iterate-based approaches. And computational costs are factors to consider [9]. The future of this work is to implement techniques such as dropout or changing the architecture of the network to

TABLE 1
Model accuracy percentages

Validation data	Test data
73.93835783004761%	73.5502302646637%

TABLE 2
Mean values of equation (2) and (3) with the test data

Mean absolute error	Mean relative error
0.006887367	1.525050225%



(a) Comparison between validation data and real value.

(b) Comparison between test data and real value.

Fig. 7. Plot of predictions and real values of the diffuse fraction.

reduce the number of errors and thus the computational costs.

6 Conclusion

In this paper, a Feedforward neural network is proposed. This model describes a neural network that estimates the diffuse fraction value over the time. The network architecture could be used for prediction, system modeling, time series and interpolations.

No doubt there is the possibility of increasing the accuracy of this network, future work could implement techniques or changes in hyper-params and see what happens in training, in addition to changing the architecture or separating the data differently. And something important to watch out for is overfitting.

Finally, it was a surprise to see the good performance of the predictions in spite of the accuracy.

Acknowledgments

The author wishes to express his gratitude to Dr. YAMADA Noboru, Manager of the Energy Engineering Laboratory, for his support and encouragement, to B.S. in M.E. Kiyoshi, to B.S. in M.E. Otaki for their explanations on Artificial Neural Networks for a classification problem and their constant support, and finally, of course to University of Guanajuato and Nagaoka University of Technology for giving to the author this wonderful opportunity to participate in the JASSO program.

References

[1] T. Pełech-Pilichowski, "On adaptive prediction of nonstationary and inconsistent large time series data," *2018 41st International Convention on Information and Communication Technology*, Electronics and Microelectronics (MIPRO), Opatija, 2018, pp. 1260-1265. doi: 10.23919/MIPRO.2018.8400228

[2] T. Peech-Pilichowski, Adaptive algortithms of event detection from time series, PL: Adaptacyjne algorytmy detekcji zdarze w szeregach czasowych, PhD Thesis, AGH University of Science and Technology, 2009.

[3] R. Chandra, ”Multi-Task Modular Backpropagation For Dynamic Time Series Prediction,” *2018 International Joint Conference on Neural Networks (IJCNN)*, Rio de Janeiro, 2018, pp. 1-7. doi: 10.1109/IJCNN.2018.8489740

[4] K. Chakraborty, K. Mehrotra, C. K. Mohan, and S. Ranka, “Forecasting the behavior of multivariate time series using neural networks,” *Neural networks*, vol. 5, no. 6, pp. 961–970, 1992

[5] Yong Zhou, Yanfeng Liu, Yingya Chen, Dengjia Wang, General models for estimating daily diffuse solar radiation in China: Diffuse fraction and diffuse coefficient models, *Energy Procedia*, Volume 158, 2019, Pages 351-356, ISSN 1876-6102,

[6] R. Meenal and A. I. Selvakumar, ”Review on artificial neural network based solar radiation prediction,” *2017 2nd International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, 2017, pp. 302-305. doi: 10.1109/CESYS.2017.8321285

[7] S. Fox, Fisiología humana (13a. ed.), McGraw-Hill Interamericana, 2014. ProQuest Ebook Central, https://ebookcentral.proquest.com/lib/ugtomhe/detail.action?docID=3221049.

[8] D. KINGMA, BA, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[9] Q. Xiao and Y. Si, ”Time series prediction using graph model,” *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, Chengdu, 2017, pp. 1358-1361. doi: 10.1109/CompComm.2017.8322764

PLACE PHOTO HERE

Oscar Alberto Santos Muñoz is a bachelor student in the University of Guanajuato, Salamanca, GTO, 36885. He has worked in Dual-Axis Sun Tracking and applied Neural Networks for photovoltaics. His research interests includes energy engineering, computer science, intelligent mechanics, robotics and control systems.