AMERICAN UNIVERSITY OF BEIRUT

Solar Irradiance Forecasting via Artificial Neural Networks

by

Mohammad F. El Hajj Chehade, Intelligent Power Systems
Taha Koleilat, A.I. & Machine Learning

Instructor: Dr. Maher Nouiehed

A project submitted for the course
ENMG 616 - Advanced Optimization & Techniques
offered by the Department of Industrial Engineering and Management
of the Maroun Semaan Faculty of Engineering and Architecture (MSFEA)
at the American University of Beirut
Beirut, Lebanon

Abstract

The increasing demand for energy has led to the worldwide adoption of solar photovoltaic (PV) systems. The integration of PV systems into the national electricity grid requires accurate forecasts of solar irradiance. A method based on artificial neural networks (ANN) is proposed to predict solar irradiance using various weather parameters. The ANN model is implemented from scratch and optimized using gradient descent (GD) and stochastic gradient descent (SGD) methods. Both methods show good convergence, with the SGD performing better in terms of mean-squared error (MSE). Test results show the superiority of the proposed model when compared to a a linear regression model based on MSE, mean-absolute error (MAE) and root mean-squared error (RMSE).

Keywords - solar photovoltaic - solar irradiance - artificial neural networks - gradient descent - stochastic gradient descent - linear regression - mean-squared error

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The population worldwide has been on the rise for the past decades, which resulted in an increase in the basic needs of the everyday lives of humans [1]. Energy tops the lists of resources that must be increased, with an estimated increase of electricity demand expected to reach 70% from 2015 in the next couple of years [2]. The world has been relying for the past century on fossil fuels for power generation that are not only depletable but also suffer from heavy environmental drawbacks [3].

As a response to that, many countries have been recently investing in renewable energy. Solar energy in particular has been deemed the most promising source due to the abundancy of solar radiation [4]. Solar Photovoltaics (PV), in particular, have been gaining attention in electricity production due to their economic and environmental benefits. Its technology is based on converting the sunlight irradiance into electricity through the photovoltaic effect [5].

Although PV energy offers itself as a cheap and eco-friendly alternative to traditional thermal sources, the integration of PV into the national grid suffers from several drawbacks. PV, like other renewable sources, is intermittent by nature [1]. In other words, solar energy production depends on weather factors that vary with time, resulting in a very chaotic and uncontrollable energy output [6]. When integrated with the electricity grid on a large-scale, PV systems may cause reliability issues due to underproduction, and excessive costs during overproduction, and may consequently degrade the grid [7].

For a reliable and economic integration of PV, grid operators must continuously receive accurate forecasts of solar irradiance in real-time [3]. Solar irradiance is mainly a function of several other meteorological factors, such as temperature, wind speed and humidity [8]. Since accurate forecasting methods have been developed for these factors, models have been developed to deduce the irradiance from those forecasts [9].

Most of the models developed for solar irradiance forecasting rely on numerical weather prediction (NWP) [10]. Although they are widely accepted as a decent forecasting technique, they are computationally expensive and require the processing of large datasets [11]. Consequently, they fail in the case of short-term forecasting needed by energy control centers. Some statistical methods that use regression and time-series techniques have also been utilized. Nonetheless, the non-stationary and non-linear nature of solar irradiance has limited their success [12].

Artificial neural networks (ANN) have emerged recently in the area of machine learning as a successful forecasting model [1, 3, 6, 7, 13, 14]. ANNs are powerful function approximators that learn data accurately through multiple processing layers, which deal with representations of data with high levels of abstraction [15].

The aim of this work is to develop an ANN model to forecast solar irradiance based on various weather parameters. The ANN model is built from scratch, and its parameters are optimized via gradient descent (GD) and stochastic gradient descent (SGD) schemes. The performance of the model is tested against a linear regression model. The rest of the paper is divided as follows: Chapter 2 formulates the problem mathematically, Chapter 3 describes the proposed solution, Chapter 4 outlines the training scheme and convergence performance, Chapter 5 evaluates the performance of the model, and Chapter 6 concludes the report.

Chapter 2

Mathematical Formulation

2.1 ANN

Neural networks are structures formed of several layers, each having a certain number of neurons. A simple neuron network is shown in Fig. 2.1. Each color represents a layer, and each circle represents a neuron. The first layer of an ANN is the input layer, where a vector of input features is fed. A notable advantage of neural networks is that the input data does not need to be processed before being fed to the network. The input then propagates through a certain number of hidden layers, the result of which is the final output layer. This layer consists of one or several neurons depending on the problem. ANNs can be used for both classification and regression problems based on the loss function.

The propagation of the data through the neural network is a function of the network parameters, a set of weights $W = \{ w_1, w_2, ..., w_i, ..., w_{m-1}\}$ and biases $B = \{ b_1, b_2, ...,b_i,... , b_{m-1} \}$ where m is the number of layers in the network. Each weight w_i is a matrix of dimensions $l \times k$ and each bias b_i is a vector of dimension l, where k is the number of neurons in the layer $i-1$ and l the number of neurons in layer i.

The value of each neuron in layer i is a linear combination of the neurons of the previous layer followed by a non-linear activation function. This function is famously used to be a a hyperbolic tangent, a rectifier (ReLu), or a sigmoid. The parameters used for the linear combination are the weights and biased and are the ones optimized. Eq. 2.1 illustrates this relation.

$$a_i = f(w_{i-1}a_{i-1} + b_{i-1}) \tag{2.1}$$

where a_i is known as the activation vector that represents the values of the neurons at layer i, f is the activation function, and w_{i-1} and b_{i-1} are the weights and biases respectively.

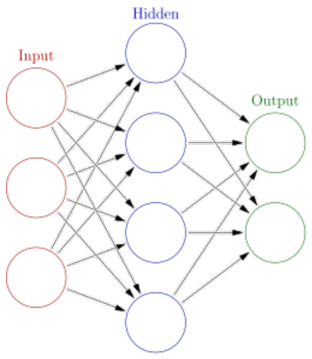

Figure 2.1: A simple neural network.

A crucial property of neural networks is that they are function approximators. A theorem known as the Universal Approximation Theorem states that any function f can be approximated using a neural network with enough neurons and layers.

2.2 Optimization Problem

The solar irradiance forecasting problem can be formulated as an unconstrained optimization problem. The objective function in Eq. 2.2 is a mean squared error (MSE) function, where the average of the squared difference between the actual and forecasted values of irradiance is minimized. The predicted value is a continuous output from the neural network found in its output layer. Since layers of a neural network are related to each other through Eq. 2.1, this error propagates back through the layers. As a result, the parameters of the ANN, namely the weights and biases, are tuned using a reverse-propagating mechanism known as back-propagation and discussed in the next Chapter.

$$\min_{W,B} \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{2.2}$$

where the $\hat{y}$ and y are respectively the forecasted and actual value of solar irradiance in W/m^2, n the number of samples, and W and B are the sets of weights and biases respectively.

Chapter 3

Proposed Solution

3.1 Weights Initialization

When the weights are initialized using a Normal Distribution with a fixed standard deviation, deep Neural Network models have trouble converging to an optimal point. This occurs as a result of the weights' variance not being taken into account, which causes very large or small activation values and, as a result, exploding or vanishing gradient problems during back-propagation which lead to the weights diverging in the former and stop learning in the latter. As the depth and layers of the artificial neural network is increased, the problems becomes worse since gradients are multiplied with each other in the update process.

In this neural network architecture, we initialize the weights of the Dense Layers using an initialization method proposed by Kaiming He. He and his colleagues [16] developed a technique that randomly samples the weights from a Gaussian Distribution with mean 0 and variance equal to $\sqrt{\frac{2}{n^L}}$ where n^L is the number of nodes or units in layer L of the Dense Neural Network. With this initialization method, we can expect convergence to be faster and the final optimal solution to be better all the while eliminating exploding and vanishing gradients.

3.2 Back-propagation

As mentioned in the previous chapter, the error propagates back from the output to the input. The weights and biases in each layers are updated in a recursive manner through a method known as back-propagation. Back-propagation may use a first-order method such as gradient descent in order to update the values of the ANN parameters. Let $L(w_i, b_i)$ be the RMSE loss function. The gradient of L is defined as shown in Eq. 3.1.

$$\nabla \mathbf{L} = (\ldots, \frac{\partial L}{\partial w_i}, \ldots, \frac{\partial L}{\partial b_i}, \ldots) \tag{3.1}$$

The partial derivative of the loss function with respect to the weight w_i linking layers i and $i+1$ is:

$$\frac{\partial L}{\partial w_i} = \frac{\partial f}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a_{m-1}} \ldots \frac{\partial a_{i+2}}{\partial a_{i+1}} \frac{\partial a_{i+1}}{\partial w_i} \tag{3.2}$$

where

$\frac{\partial L}{\partial \hat{y}} = \frac{2}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)$

$z_m = w_{m-1} a_{m-1} + b_{m-1}$ and $a_m = f(z_m)$

$\frac{\partial \hat{y}}{\partial a_{m-1}} = \frac{\partial \hat{y}}{\partial z_m} \frac{\partial z_m}{\partial a_{m-1}} = f'(z_m) w_{m-1}$

$\frac{\partial a_{i+2}}{\partial a_{i+1}} = f'(z_{i+2}) w_{i+1}$

$\frac{\partial a_{i+1}}{\partial w_i} = \frac{\partial a_{i+1}}{\partial z_{i+1}} \frac{\partial z_{i+1}}{\partial w_i} = f'(z_{i+1}) a_i$

3.3 Iterative Descent Methods

After performing back-propagation, a gradient descent scheme is adopted as shown in Eq. 3.3. Full-batch and mini-batch gradient descent algorithms are used. The full-batch method uses all data points in each iteration while the mini-batch uses a subset of the samples chosen at random. The pseudo-code for the descent method is shown in Algorithm 1.

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla \mathbf{L} \tag{3.3}$$

where w is the vector of parameters and α is the constant step-size.

3.4 Gradient Clipping

To further enhance the performance of our model, we will implement gradient clipping which will trim a gradient if it grows too large in order to maintain an acceptable value. Having large gradients in the back-propagation phase will lead to the weights diverging to a bad region. Gradient clipping guarantees that the norm of the gradient vector $\nabla \mathbf{L}$ is at most T, which is a threshold value. In other words, if $\|\nabla \mathbf{L}\| \geq$ T, then $\|\nabla \mathbf{L}\| =$ T. So, we basically bound $\|\nabla \mathbf{L}\| \in$ [-T,T].

This enables gradient descent to behave reasonably even when the model's loss

Algorithm 1 Iterative Descent Scheme

```
if full-batch then
    for iterations do
        perform forward propagation using all samples
        perform back propagation using all samples
    end for
else if mini-batch then
    for iterations do
        for batch size do
            randomly select a sample
            perform forward propagation using the sample
            perform back propagation using the sample
        end for
    end for
end if
```

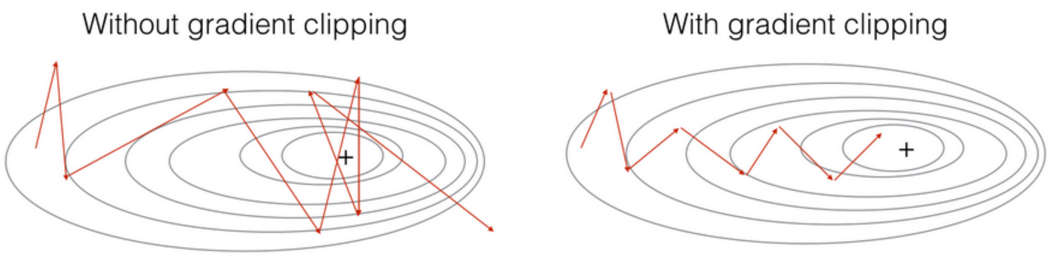

Figure 3.1: Gradient Descent with and without gradient clipping

landscape is erratic. As described in Fig. 3.1, gradient clipping ensures that the update for the parameters is constrained and not too large so that gradient descent is not too rapid to diverge to a "bad" region.

3.5 Data Pre-processing

The solar data was pre-processed to extract 16 features which will be used in our model to predict solar irradiance. Some features that will be used include temperature, pressure, humidity, wind direction in degrees, sunrise time, sunset time along with the time data. We extracted the month, day, hour, minute, and seconds from the time given in the data, along with the hours and minutes for sunrise and sunset times. We finally digitize (convert continuous data to discrete) the humidity and wind direction into bins as seen in Fig. 3.2, Fig. 3.3, and Fig. 3.4. To better understand the patterns and relations in the features extracted, we plot a matrix that calculates the correlation between all possible feature pairs in the data. To finally prepare the data to be used for model training, we split the data into 80% training data and 20% testing data.

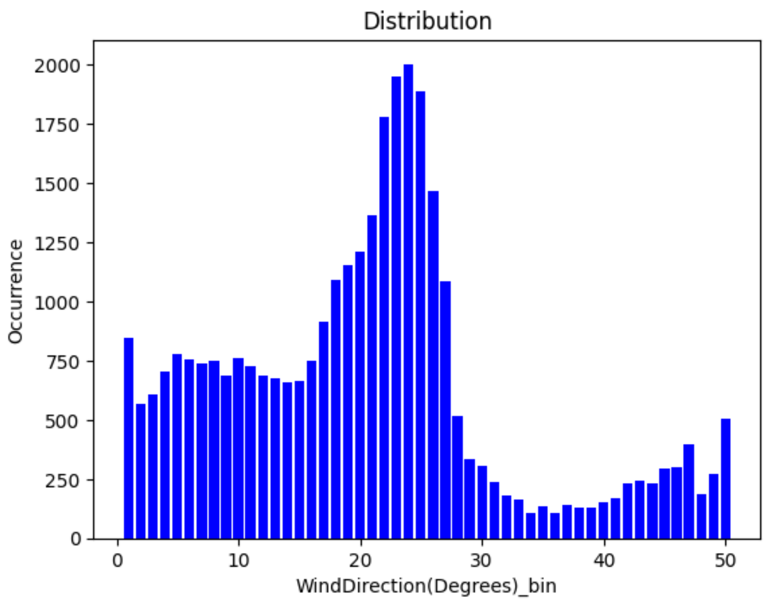

Figure 3.2: Digitization of the wind direction into bins

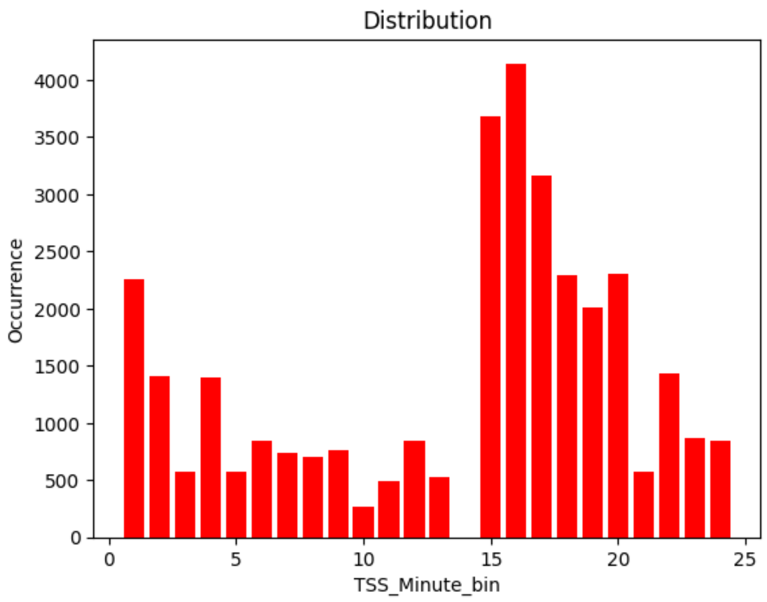

Figure 3.3: Digitization of the sunset time into bins

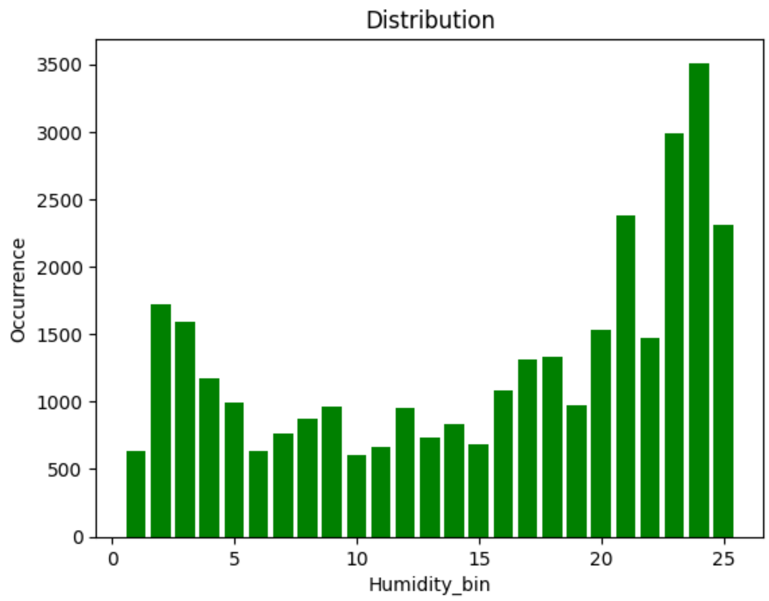

Figure 3.4: Digitization of the humidity into bins

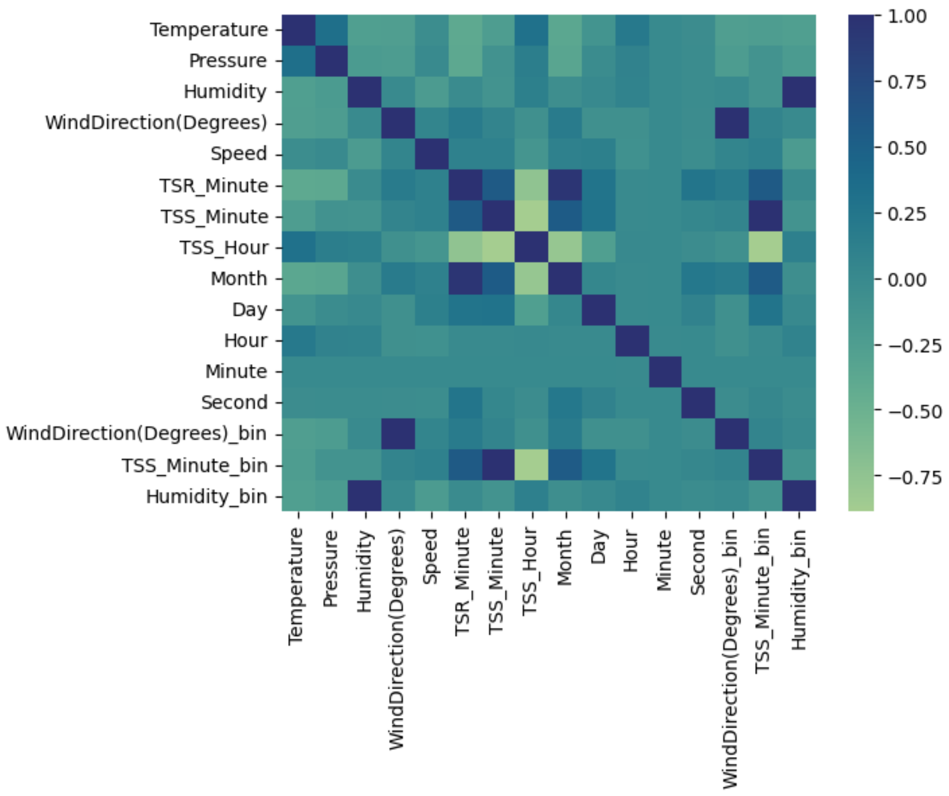

Figure 3.5: Correlation matrix of the 16 extracted features.

Chapter 4

Convergence Analysis

Full-Batch and Stochastic Gradient Descent (SGD) optimization methods with different hyper-parameters were utilized to decrease the MSE Loss with each epoch. As seen from Fig, 4.1, Batch Gradient Descent was the slowest to converge where it took 1,200 epochs to converge to an MSE of 10,919.83 on the Training set, whereas an MSE of 11,119.12, mean absolute error (MAE) of 51.67, and a RMSE of 105.44 on the Testing set. Increasing the learning rate would have led to divergence, and without using a diminishing learning rate throughout each 100 epochs, the training would not have converged since large updates would have kept the loss continuously oscillating.

On the other hand, we can see from Fig. 4.2 how SGD with a learning rate of 0.0001 and a batch size of 128 samples converged really fast to achieve an MSE of 7806.26 on the Training set, where in just 100 epochs the performance on the Testing set was 8559.51, 41.79, and 92.52 for the MSE, MAE, and RMSE metrics respectively. We can notice slight over-fitting indicated by the big discrepancy of the Training and Testing MSE Loss.

Increasing the Learning rate to 0.001 for the SGD optimizer introduced ”noise” in the training process as can be seen in Fig. 4.3; in other words, the loss decreased but with many oscillations as the number of epochs increased due to large updates. The Final MSE Loss on the training set was 6936.05 while the MSE, MAE, and RMSE on the testing set were 9705.06, 46.37, and 98.52 respectively.

We finally optimize using SGD with a diminishing learning rate starting off with a value of 0.0001, and we notice in Fig. 4.4 how having a diminishing step-size decreased the over-fitting at the end even with training for the same number of epochs. Another possible solution to this over-fitting problem would have been to utilize Dropout layers between each Dense layer which would regularize the weights by nullifying certain nodes in a layer randomly according to a certain rate. The MSE on the training set was 8351.33, whereas the MSE on the testing

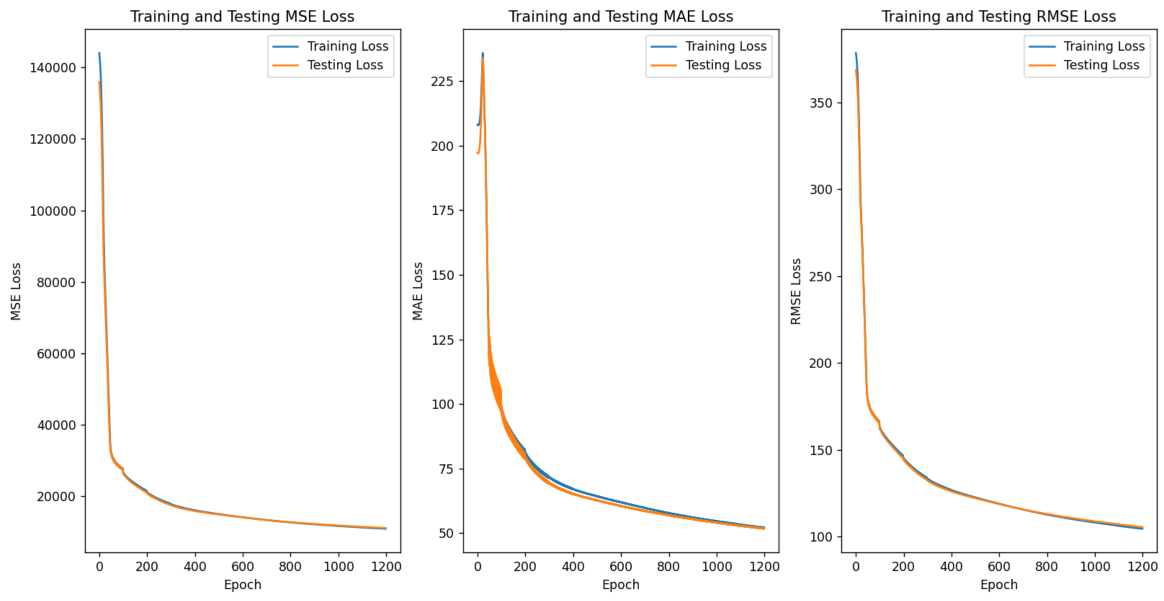

Figure 4.1: Loss over 1200 Epochs using GD with diminishing step size with 0.001 Learning Rate

set was 8887.98 with an MAE of 43.08 and a RMSE of 94.27.

It is important to note that increasing the batch size would make SGD to converge more slowly, and taking the whole batch would make it equivalent to Batch Gradient Descent, whereas decreasing the batch size might not guarantee convergence due to more oscillations being introduced just as described in Fig. 4.5, where a batch size of 32 was utilized which achieved an MSE of 8,752.26, an MAE of 40.83, and an RMSE of 93.55 on the testing data.

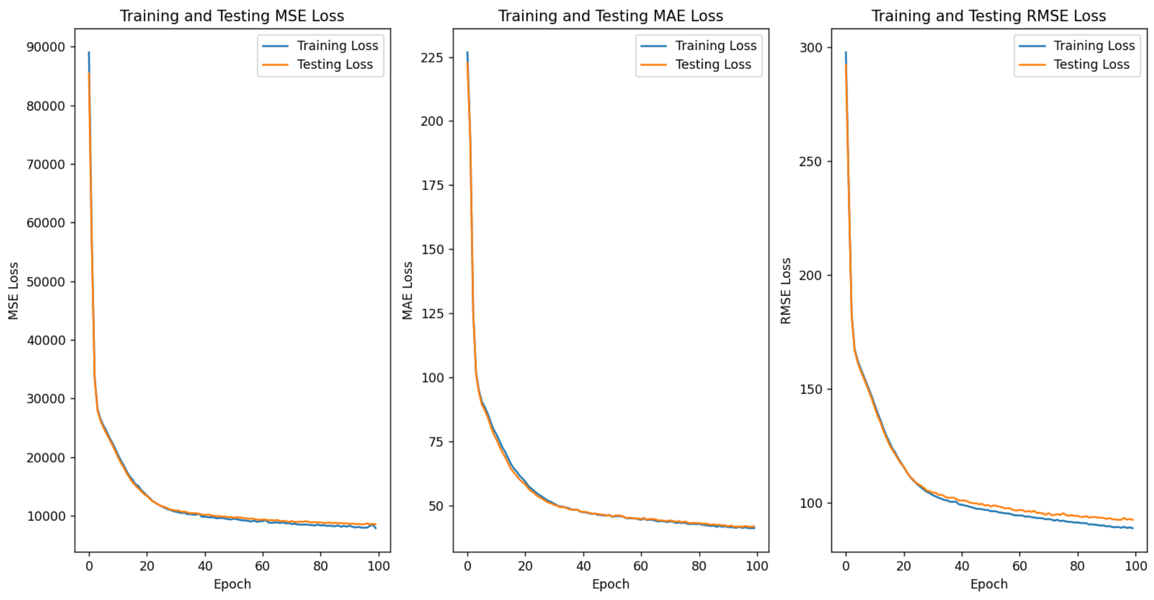

Figure 4.2: Loss over 100 Epochs using SGD with 0.0001 Learning Rate and 128 Batch Size

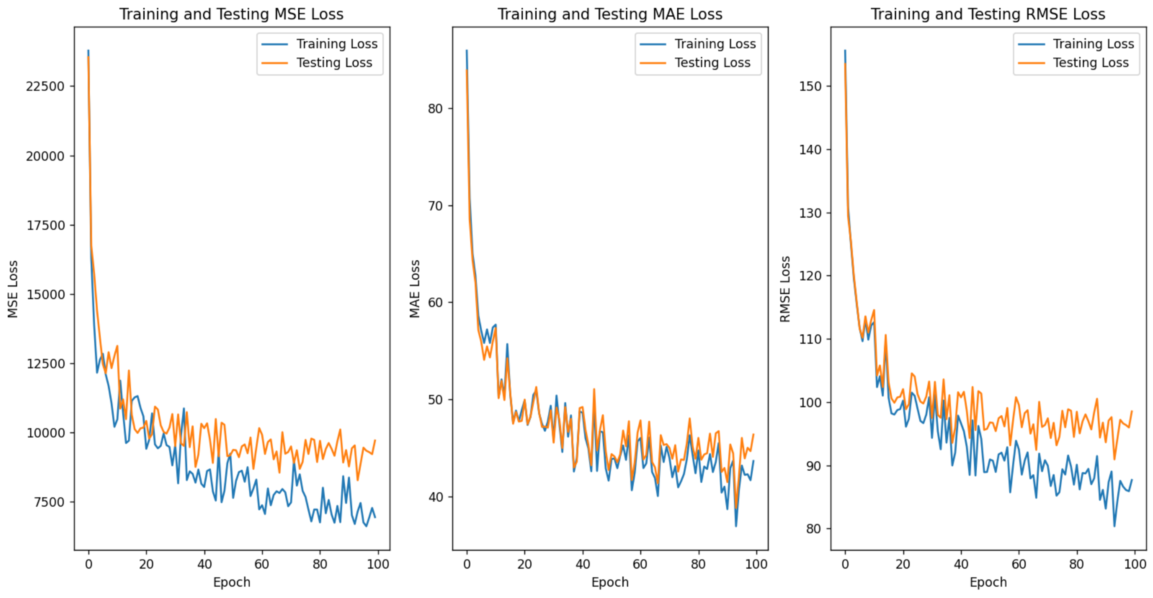

Figure 4.3: Loss over 100 Epochs using SGD with 0.0001 Learning Rate and 128 Batch Size

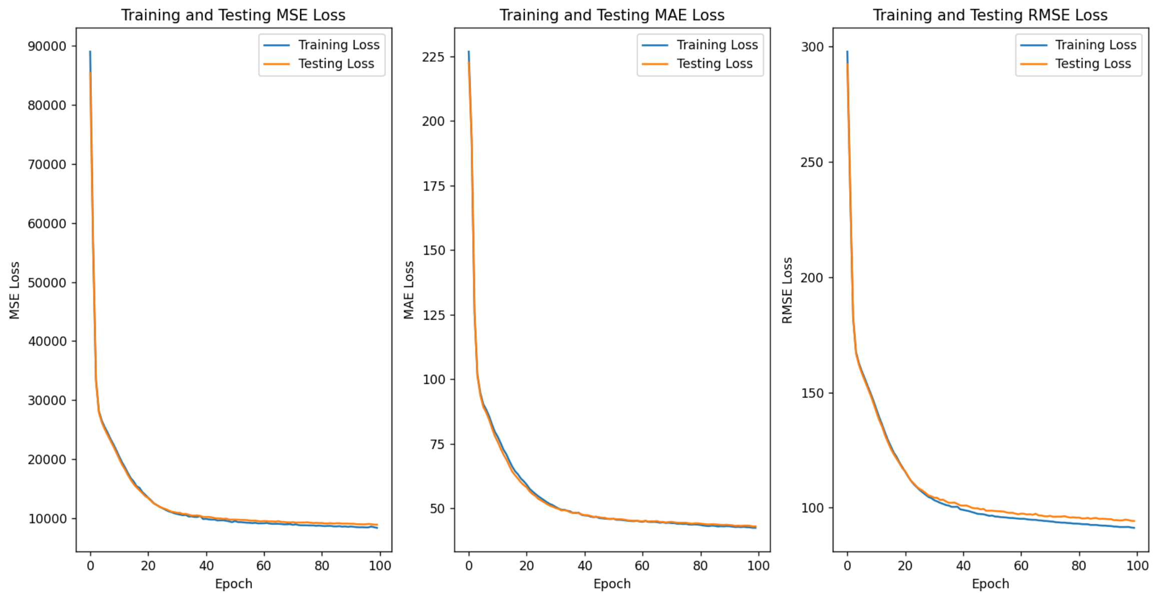

Figure 4.4: Loss over 100 Epochs using SGD with diminishing step size with 0.001 Learning Rate and 128 Batch Size

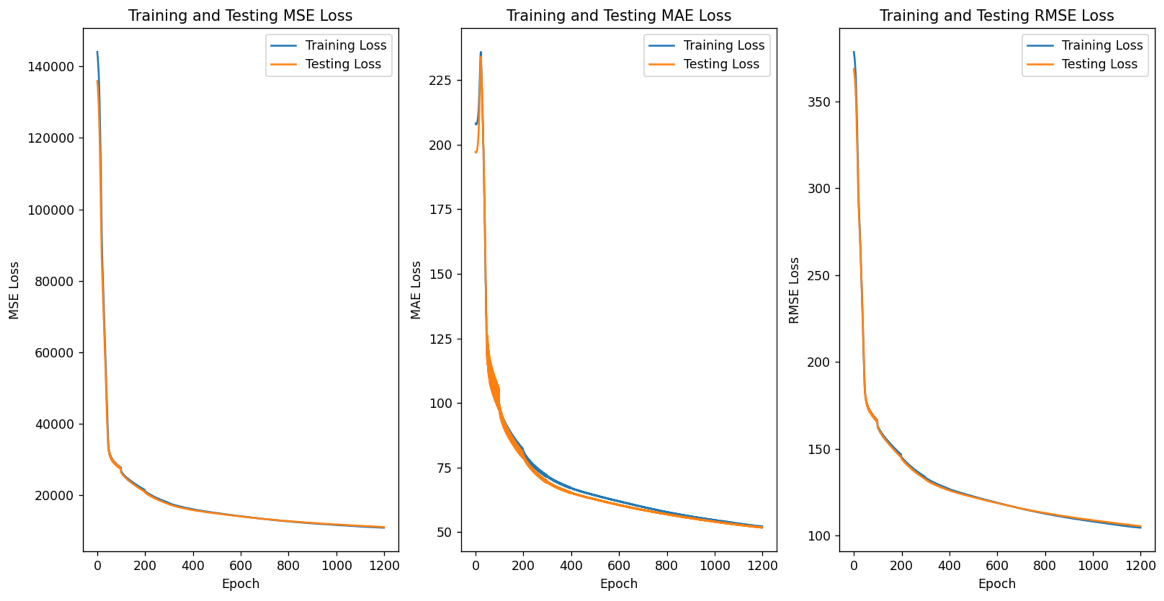

Figure 4.5: Loss over 100 Epochs using SGD with diminishing step size with 0.001 Learning Rate and 32 Batch Size

Chapter 5

Experimental Results

5.1 Comparison between Models

To put things into perspective, we compare our Neural Network model with a linear regression model. The linear regression model has achieved an MSE of 35,964.13 and 35,658.31 on the training and testing data respectively. In comparison, the Neural Network model outperformed the linear regression model where the best neural network model achieved an MSE, MAE, and RMSE loss of 8,752.26, 40.83, and 93.55 on the testing data respectively. On the other hand, the best linear regression model achieved 35,658.31, 141.90, and 188.83 respective values on the testing set using SGD with a batch size of 32 trained for 100 epochs as shown in Fig. 5.1. The results are reasonable as the relationships between the input and output data is highly non-linear. We summarize the performance metrics obtained for the best models in Table 5.1 where we conclude that the neural network model is the best model we got.

Table 5.1: Performance Metrics on the testing set for the best models obtained using SGD and GD

Models	MSE	MAE	RMSE
Neural Network SGD	8752.26	40.83	93.55
Neural Network GD	11119.12	51.67	105.44
Linear Regression SGD	35658.31	141.90	188.83

5.2 Real-time prediction on a recent weather data sample

We sample a weather forecast for Dallas, Texas containing weather conditions that we will use as features for our model input to forecast solar irradiance for

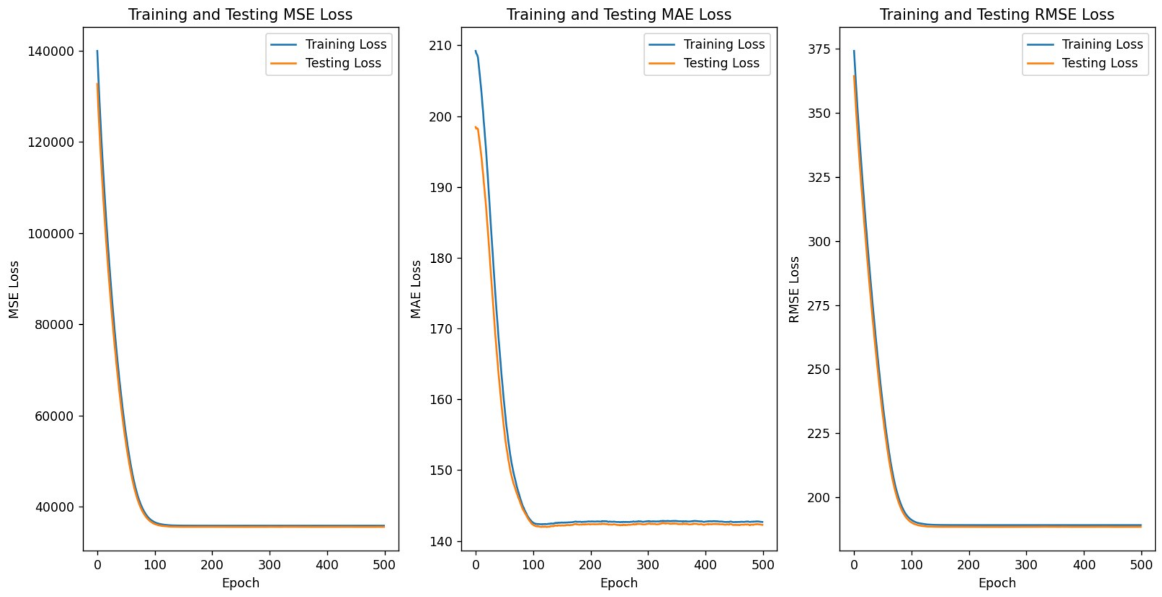

Figure 5.1: Loss for Linear Regression Model over 100 Epochs using SGD with Batch Size of 32

Saturday, November 12, 2022. After obtaining the required input feature values, we predict the solar irradiance for this example using the best trained model that we optimized. The solar irradiance that the model was forecasting was a value of 600 which is very close to predicted solar radiation numbers from this website which states that Dallas, Texas will experience solar irradiance values ranging from 511 up to 597 w/m^2 between 11 AM and 2 PM this Saturday, November 12, 2022.

Chapter 6

Conclusion

In this project, an artificial neural network (ANN) model is built for solar irradiance forecasting based on several meteorological parameters. The weights of the neural network are first initialized using a special initialization scheme. Then, the parameters are updated using gradient descent (GD) and stochastic gradient descent (SGD) methods. Gradient clipping is proposed to prevent network divergence. The ANN shows satisfactory convergence with the SGD model reaching a mean-squared error (MSE) of about 9,000, and the GD approaching an MSE of 11,000. Test results show that the proposed ANN structure outperforms the linear regression model. Future work may include incorporating time series irradiance data through optimizing a long short-term memory (LSTM) neural network model and a temporal convolutional network (TCN).

Bibliography

[1] P. Kumari and D. Toshniwal, “Deep learning models for solar irradiance forecasting: A comprehensive review,” *Journal of Cleaner Production*, vol. 318, p. 128566, 2021.

[2] A. Duffy, M. Rogers, and L. Ayompe, *Renewable energy and energy efficiency: assessment of projects and policies.* John Wiley & Sons, 2015.

[3] P. Kumari and D. Toshniwal, “Long short term memory–convolutional neural network based deep hybrid approach for solar irradiance forecasting,” *Applied Energy*, vol. 295, p. 117061, 2021.

[4] F. Wang, Z. Xuan, Z. Zhen, Y. Li, K. Li, L. Zhao, M. Shafie-khah, and J. P. Catalão, “A minutely solar irradiance forecasting method based on real-time sky image-irradiance mapping model,” *Energy Conversion and Management*, vol. 220, p. 113075, 2020.

[5] P. G. V. Sampaio and M. O. A. González, “Photovoltaic solar energy: Conceptual framework,” *Renewable and Sustainable Energy Reviews*, vol. 74, pp. 590–601, 2017.

[6] B. Brahma and R. Wadhvani, “Solar irradiance forecasting based on deep learning methodologies and multi-site data,” *Symmetry*, vol. 12, no. 11, p. 1830, 2020.

[7] M. Abdel-Nasser, K. Mahmoud, and M. Lehtonen, “Reliable solar irradiance forecasting approach based on choquet integral and deep lstms,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 1873–1881, 2020.

[8] N. Sharma, P. Sharma, D. Irwin, and P. Shenoy, “Predicting solar generation from weather forecasts using machine learning,” in *2011 IEEE international conference on smart grid communications (SmartGridComm)*, pp. 528–533, IEEE, 2011.

[9] J.-P. Lai, Y.-M. Chang, C.-H. Chen, and P.-F. Pai, “A survey of machine learning models in renewable energy predictions,” *Applied Sciences*, vol. 10, no. 17, p. 5975, 2020.

[10] A. Murata, H. Ohtake, and T. Oozeki, "Modeling of uncertainty of solar irradiance forecasts on numerical weather predictions with the estimation of multiple confidence intervals," *Renewable energy*, vol. 117, pp. 193–201, 2018.

[11] Y. Hao and C. Tian, "A novel two-stage forecasting model based on error factor and ensemble method for multi-step wind power forecasting," *Applied energy*, vol. 238, pp. 368–383, 2019.

[12] G. Reikard, "Predicting solar radiation at high resolutions: A comparison of time series forecasts," *Solar energy*, vol. 83, no. 3, pp. 342–349, 2009.

[13] F. Wang, Y. Yu, Z. Zhang, J. Li, Z. Zhen, and K. Li, "Wavelet decomposition and convolutional lstm networks based improved deep learning model for solar irradiance forecasting," *applied sciences*, vol. 8, no. 8, p. 1286, 2018.

[14] X. Huang, Q. Li, Y. Tai, Z. Chen, J. Zhang, J. Shi, B. Gao, and W. Liu, "Hybrid deep neural model for hourly solar irradiance forecasting," *Renewable Energy*, vol. 171, pp. 1041–1060, 2021.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," 2015.