IMPROVING HOUR-AHEAD HYBRID SOLAR IRRADIANCE PREDICTION USING DEEP LEARNING AND SKY IMAGES

by

BENJAMIN MANNING

(Under the Direction of Kyle Johnsen)

ABSTRACT

This research was performed to increase the optimization of hour-ahead hybrid solar irradiance prediction methods through the design and implementation of a new hybrid system and model that utilized sky images as a replacement variable for cloud types identified by overhead satellites. Improving current solar radiation prediction methods will benefit electricity producers that need to better understand the availability of solar irradiance and how it may impact their forecasting. Phase one outlines a comparison between current supervised learning methods and deep learning methods. Recurrent Neural Networks produced lower RMSE and higher R^2 values and outperformed supervised learning methods.

Phase two outlines building and validating a new one-hour ahead hybrid prediction model by combining a deep learning approach with a replacement feature derived from real-time image collection and location specific numerical weather features. This replacement feature was the percentage of sky cover from an observation point on the ground and was called Sky Types. Sky Types are less expensive to obtain and can be collected at any location, which also makes the prediction of solar irradiance specific to the same location. Deep learning models validated that the use of Sky Types was not only a valid substitution for cloud types, but were more optimal for training as model performance improved with reduced network topology and while still being optimal for hour-ahead predictions.

To use Sky Types in the new hybrid prediction model, a system was designed to collect the sky condition information from the National Weather Service and relevant images representing the sky condition; both were captured at the same time intervals. Phase three outlines the creation of a system used for collecting images and weather data, preparing images for use by the new hybrid prediction model and building a classification model using a Convolutional Neural Network.

The system's GHI predictions were validated using hour-ahead ground truth solar irradiance amounts from ten locations and averaged an RMSE of 41.26 W/m^2 and outperformed GFS forecasted GHI by 32% on highly variable weather days. This new hybrid system can be used anywhere numerical weather data and sky images can be captured.

INDEX WORDS: machine learning, artificial intelligence, solar irradiance, solar energy

IMPROVING HOUR-AHEAD HYBRID SOLAR IRRADIANCE PREDICTION USING DEEP LEARNING AND SKY IMAGES

by

BENJAMIN MANNING

BS, University of Southern Mississippi, 1997

MS, University of Southern Mississippi, 1999

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2018

IMPROVING HOUR-AHEAD HYBRID SOLAR IRRADIANCE PREDICTION USING DEEP LEARNING AND SKY IMAGES

by

BENJAMIN MANNING

Major Professor:	Kyle Johnsen
Committee:	Brian Bledsoe
	David Gattie
	Lakshmish Ramaswamy

DEDICATION

To my beloved wife, Penny; this is as much yours as it is mine. For all the years you heard me say I would finish this – *we did this together*. To my brave and courageous son, Blake; I am so proud of the man you have become – never stop learning and questioning. To Mom and Dad – thank you for not giving up on me. Rest in Peace, Mom.

ACKNOWLEDGEMENTS

I would like to thank my dissertation committee for their help in creating this work. A special thanks to Dr. Kyle Johnsen for giving an old man the chance to finish something he started too long ago. Lastly, I would like to thank Dr. Ray Bareiss and Dr. Roger Schank for their guidance, understanding, mentoring and friendship.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

CHAPTER ONE

INTRODUCTION

The need for adequate renewable energy resources has never been more prevalent than in today's times. As we progress as a society and the population continues to expand, we are quickly using more of our current energy resources than we can adequately replace in some type of manageable way. Our climate is changing rapidly and decades of mining our planet's precious resources have resulted in a steep decline in their availability and an increase in the research, investigation and production of new methods for different applications and implementations of renewable energy resources (Bauner & Crago, 2015). Solar energy shows great promise in filling this role and, as solar photovoltaic hardware continues to develop and advance, the full understanding of the potential solar energy can provide to our society has never been more important (Best & Burke ,2018). In addition, rapidly advancing technologies such as blockchain are expanding the availability of solar energy across regions where it would have otherwise been incapable of being used (Basden & Cottrell, 2017). Neighborhoods are beginning to share solar energy resources through the expanded use of micro-grids (Mengelkamp et al., 2018); rural farmlands in Australia are being powered by distributed solar energy resources (Kim, Park & 2018) and even energy sharing electric car installations are being proposed (Financial Review, 2018). This is only the beginning of the widespread adoption of solar energy and, as the advancement of solar photovoltaic panels also increases, so will their adoption, the benefits of their use and the size of the industry and its investment potential (Pieroni, 2018).

This work is being presented to improve current hour-ahead solar irradiance prediction methods, particularly the value of sky imaging and deep learning to predictions. Electricity producers may need to better understand the availability of solar radiation and how it may impact their forecasting. Moreover, predicting both availability and production capacity is necessary for preventing the overproduction of electricity and vital to demonstrating the full potential solar energy can provide in many different areas and industries.

Prediction methods not only help in these areas, but can also help increase awareness and understanding of the optimal return on investment that is available. This work will begin with an introduction and explanation of solar irradiance and the current solar model before introducing machine learning, which is the primary methodology under evaluation. In addition, the relevant methods and studies investigating hour-ahead solar irradiance prediction that have come before this work will also be presented. This will provide a foundation for this research and provide success metrics. These previous methods are divided into two branches that are separated by their implementation: hybrid approaches that utilize numerical weather data and other external features that impact solar irradiance and singular approaches that only use numerical weather data. Our primary contributions surround a new hour-ahead hybrid system and are presented in three main phases.

Chapter 4 outlines an evaluation of modern GHI hour-ahead prediction methods, including descriptions of all variables used in current prediction processes through exploratory data analysis. As part of this evaluation, classification and regression models were constructed using five different supervised machine learning algorithms. This effort established a baseline for understanding and improving upon existing methods.

Solar irradiance is impacted most by clouds (Perez et al., 2007) and the current Solar PV model uses regional satellite imagery to ascertain the types of clouds that are in a region. This is one of the main variables used in predicting solar irradiance, but it is sparsely collected, contains missing observations and it is related to regions and thus cannot be used for making predictions at specific locations. Chapter 5 presents the novel application of sequential deep learning methods to build classification and regression models to the prediction tasks. Evidence supports that deep learning methods outperform current supervised machine learning methods for this type of application. Further, we introduce an extension of this deep learning approach with a replacement variable drawn from real-time data collection **and location specific** numerical weather data, the percentage of sky cover taken from an observation point on the ground. The new hybrid model will be less costly to implement and will be relevant to specific and individualized locations where more regionally-based hour-ahead solar prediction models fall short. Using variables from hour-ahead weather forecasts from the Global Forecasting System, the new hybrid model will be validated to be more accurate than current hour-ahead GHI prediction supervised learning models from Chu et al (2014) and Amrouche and Le Pivert (2014) by 10%. In addition, this new hybrid prediction model is applicable to all weather conditions wherein the two previously cited models are not utilized in all weather conditions.

Chapter 6 outlines the design and implementation of an image collection system that obtains real-time images for use in the new hybrid model from chapter 5. This new system collects real-time images from publicly available cameras and processes them for use by the new hybrid prediction model. This new system is not only less expensive to implement but can also be scaled and is specific to any location where the technology is available. In future work, this new system can be used to collect and store local data that could be used to predict local solar irradiance ahead every fifteen minutes.

Hybrid approaches that use both numerical weather data and external variables that impact solar irradiance are already beginning to show great promise in solar irradiance prediction and these will continue to grow. This growth is relative to the expansion and advancement of technology which is greatly impacting accuracy and error rates in similar studies that utilize more traditional approaches for prediction. This research not only provides a foundation for ongoing and future research, but already directly challenges current methods.

CHAPTER TWO

BACKGROUND KNOWLEDGE

2.1. Solar Irradiance

We often don’t think of the sun as a large star that is constantly producing usable energy. We could empower the entire world with a continuous 1.2 x 10^5 terawatts (Crabtree & Lewis, 2007) of energy if we could fully harness the power of the sun. At any given time, the sun’s solar irradiance is 1000 watts/m^2 at any point in space within our local atmosphere, but that energy is most often refracted by numerous elements within our atmosphere (Lave, Hayes, Pohl & Hansen, 2015) breaking solar irradiance into two different components: Direct Normal Irradiance (DNI) and Diffuse Horizontal Irradiance (DHI) (Marion, 2015). These two components present the entire amount of solar irradiance that is available at any given time, which is referred to as Global Horizontal Irradiance or GHI.

Direct Normal Irradiance, redefined DNI, is the amount of the sun’s solar irradiance that can be captured by a perpendicular surface when the sun is at a relative perpendicular position in the sky as shown in Fig. 1. This single component is vital to the overall calculation of solar irradiance because it is the most prevalent component of the two (Bird, 1984). It is often measured using a solar photometer and is often used to in correlation with the movement of the sun to calibrate solar photovoltaic systems that include the use of movable or solar panels that track the movement of the sun to keep their orientation perpendicular to its zenith angle. Blanc et. al., (2014) calculated DNI for any given direct normal irradiance of the sun on a perpendicular surface for any wavelength λ as:

$$I_{d\lambda} = H_{O\lambda} D T_{r\lambda} T_{a\lambda} T_{w\lambda} T_{o\lambda} T_u \qquad (1)$$

Whereas $H_{O\lambda}$ is the intensity of the power of the sun at the given distance between the Earth and Sun and D a given correction factor for such distance. The remainder of the equation represents transmission functions of the atmosphere at the same given wavelength.

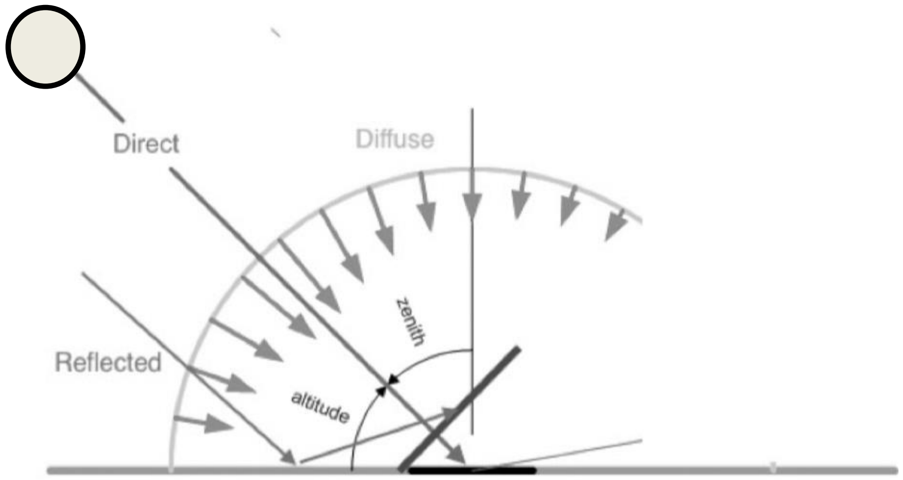


Fig 1: DNI and Solar Zenith Angle. (Performance, 2018)

The sun's incoming rays can be impacted by the Earth's atmosphere and many physical objects such as clouds and physical structures; this portion can be calculated using functions of solar altitude, atmospheric vapor content, dust content, ozone content and other features (Liu & Jordan, 1960) that it interacts with on its path to the ground. In many instances solar radiation can also be reflected or refracted from its direct impact with the ground itself sending it back into the atmosphere. As DNI encompasses the direct or normal measure of the Sun's beam, Diffuse Horizontal Irradiance or DHI is represented by this remaining solar irradiance that is reflected or refracted within the Earth atmosphere. The proportion of the incident light or radiation that is reflected by a surface is commonly referred to as Albedo. DNI, DHI and Albedo are shown is Fig. 2 and all three are considered part of the Global Horizontal Irradiance factors since all three features or components contain some type of measurement, whether directly or

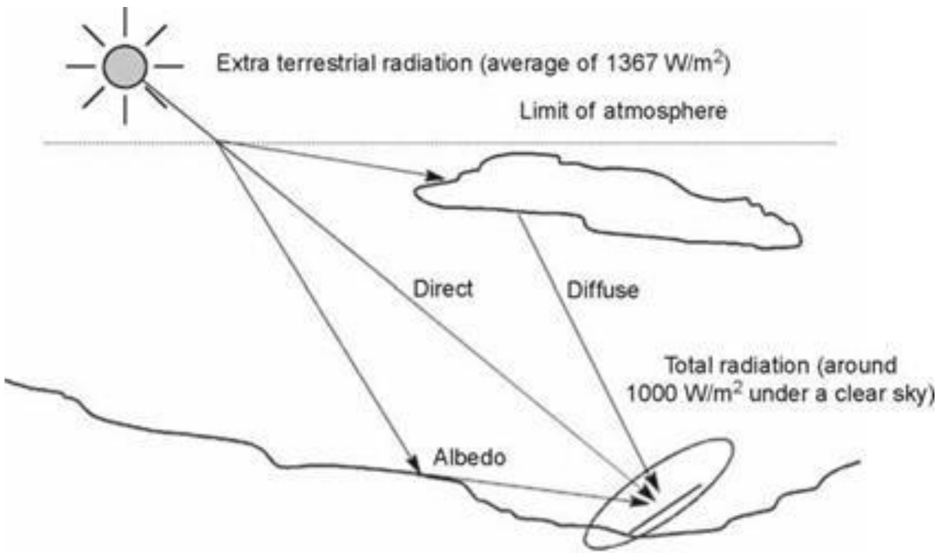

Extra terrestrial radiation (average of 1367 W/m^2)
Limit of atmosphere
Direct
Diffuse
Total radiation (around 1000 W/m^2 under a clear sky)
Albedo

Fig 2: DNI, DHI and Albedo. Data Irradiation. (2018).

indirectly, of the average 1000 W/m^2 amount of solar irradiance present at any given time on a horizontal surface.

Two main emphasis areas are commonly addressed in all GHI prediction methods. The first method uses variables from historical numerical weather data and includes cloud related features (Matuszko, 2012) that have a direct impact on DNI and as a direct correlation the amount of GHI. This method is a **hybrid prediction model** because of its usage of both historical weather-related data and cloud types, sky conditions or specific cloud features; these models have also been implemented as in the cases of Shamim, Bray, Remesan & Han (2015) and Cheng & Yu (2015). The second method utilizes predictions made with quantitative weather models (Amrouche & Le Pivert, 2014); this method **only** uses variables from numerical weather data such as wind speed, wind direction, temperature humidity, etc. and relates how each if these impacts the amount of GHI available at any given time. The main findings and related work described later is directly related to a new **hybrid approach** that will serve as one of the objectives of this work.

2.2. Forecasting Methodologies

Aman, et al. (2014) outlined four areas of energy demand and GHI forecasting: averaging models, regression models, time series and Artificial Intelligence (AI) approaches. The main difference between these four approaches is centered on the concept of how the model is constructed and the how the relationships between the variables are used, which also forms the basis of the main differences between statistical approaches and traditional machine learning and artificial intelligence methods.

In traditional statistical-based approaches, mathematical relationships are formed between independent and dependent variables; these relationships formulate or compose the structured model that is then used to infer or make predictions in future data that is structured in the same manner. In traditional machine learning inferences are not made, rather algorithms are programmed to "learn" the complex relationships among variables. Most statistical methods require a previous understanding of data collection, statistical significance and a broad understanding of procedure, whereas most machine learning processes and algorithms can operate independently without satisfying these objectives beforehand. For example, a Statistician would perform many statistical tests on data to ascertain how the variables might be related or find usable relationships in the data to build model with. Machine learning practitioners often study the data as well, but for purposes of either transforming it and/or understanding how variables are distributed to make informed decisions about the specific machine learning processes to use to on it. In addition, machine learning processes can be highly optimized when many of these objectives are known as well.

Time series methods are often considered to be statistical methods mainly due to many of the previously described processes with the addition of a time component. It is a statistical model that is repeated at various intervals of a predetermined time frame. At every interval, a new regression model is built with rules from the previous analysis and the error rate is measured between the two iterations. This interval was defined by Box et al. (1994) as a backshift operator (B) of the time series where $Bz_t = Z_{t-1}$ where z_t and Z_{t-1} are represented as two consecutive time series. This type of statistical approach is also not dependent on a given set of known features, but normally applied between one or many independent variables and one dependent variable where direct relationships can be derived before the analysis.

Regardless of the approach taken, statistical methods and machine learning methods are highly focused on a set of independent variables that are often referred to as a **feature set** in statistical methods and a **feature space** in machine learning methods as shown in Table 1. This set of variables provides input(s) to the model and establishes how predictions are made going forward. Variables in this space are often simply regarded as **features**.

The relationship, not only among features but also *between* features and known outcomes or dependent variables, is important to understanding how either of these processes work, especially in the case of machine learning.

Machine learning algorithms can be programmed to make weighted adjustments among the features that can be highly impactful to their priority and use within a model. Traditional statistical methods generally do not have this ability and any weighting of the features must occur manually, which is another reason it is mandatory to know such relationships beforehand.

Table 1: Example of a Feature Space/Set

Feature Space (Independent Variables)				**Dependent Variable**
Month	Day	Hour	Minute	GHI
1	1	0	30	0
1	1	1	0	0
1	1	1	30	0

2.3 Global Forecasting System

The most prevalent GHI prediction models (Perez et al., 2013) use weather variables taken from the National Centers for Environmental Prediction (NCEP) as inputs and the same were used for this study. The NCEP produced the Global Forecast System (GFS), which is a weather model containing various weather-related variables like temperature, wind speed, humidity, etc. The GFS can predict weather-related variables out 16 days in the future, which makes it a good model to obtain the numerical weather data for predicting GHI. In this study, predictive models will be created using a dataset of historical GFS variables and the actual GHI measurements at the same time as shown in Table 2. These predictive models will then apply any rules learned from historical weather data to predict GHI amounts one-hour ahead of the current time by using variables from one-hour ahead weather forecasts from the GFS.

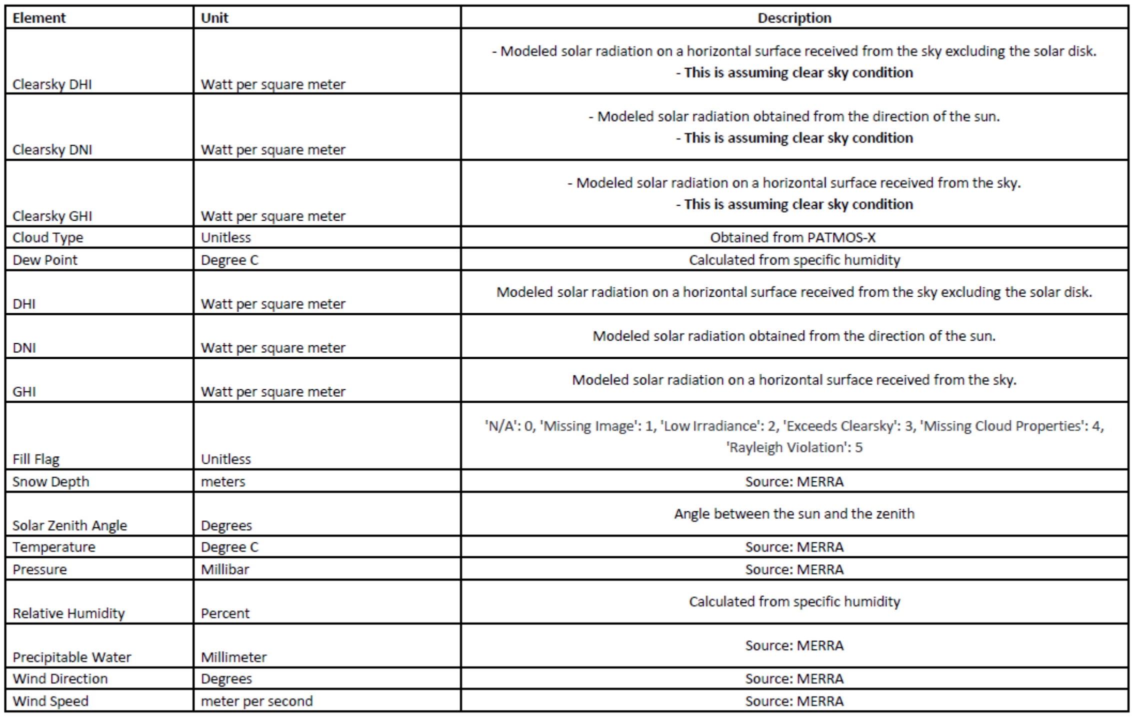
Table 2: Example of Historical Average Hourly GFS Weather Variables

Year	**Month**	**Day**	**Hour**	**Dew Point**	**Temperature**	**Humidity**	**Cloud Type**	**GHI**
2010	12	28	7	39.2	41	100	7	0
2010	12	28	8	42.8	44.6	96.25	3	29
2010	12	28	9	46.4	50	88.75	4	138
2010	12	28	10	50	51.8	94.35	4	15
2010	12	28	11	51.8	53.6	94.71	6	59
2010	12	28	12	51.8	53.6	99.05	6	11

2.4. Solar PV Model

The most current (NREL, 2015) method for modeling solar irradiance was produced and managed by the National Renewable Energy Laboratory (NREL), which also manages the latest processes for modeling the information contained within the National Solar Radiation Database (NSRDB). The evolution of the NSRDB, previously referred to as SOLMET, can be traced back to 1977 with the first models that contained numerical weather data ranging back to 1952 when there were only 248 NOAA weather stations across the United States and only 26 had the capability to measure the type of information needed to derive any type of useful inference regarding solar irradiance. The NSRDB first became a reality in 1994 (NCDC, 2018) under the management of the DOE and the National Oceanic and Atmospheric Association (NOAA) when 239 National Weather Service stations first began to model this type of data and an additional 56 stations were added to the list of those that could measure such data. In 2005, the list grew to almost 1500 National Weather Service (NWS) stations that were capable of modeling solar irradiance. These more current models utilized satellite measurements and ground observations regarding Cloud Index and Clearness Index; Cloud Index is widely used to measure solar irradiance on the Earth's surface (Hammer et al., 2015) and Clearness Index is the ratio of DNI to DHI (Mellit et al., 2008). Current solar PV models utilize numerical weather data and data from the PATMOS-X satellite (Foster & Heidinger, 2013) to predict solar radiation at the Earth's surface; the quantities and relative units of the data are shown in Table 3. This information is then used in radiative transfer models (Barker et al., 2003 that can better ascertain the layer densities and make up of individual clouds and cloud cover. This framework is often referred to as the Physical Solar Model (PSM) framework which is shown in Fig. 3. In this framework, cloud-based data is analyzed by satellites overhead with aerosols and water vapor from other sources, such as forecasts and various surface readings (Cermak et al., 2010). The collection and analysis of cloud related variables conveyed in this framework is relevant to this work as a new collection method for a similar cloud-related feature will be presented here. The 'Cloud Properties' section of the framework contains eight different variables related to clouds, none of which can be obtained from ground-based observations as shown in Fig. 3. In addition to these eight variables, various tests within classification algorithms (Bankert et al., 2009) are used to identify the cloud types (Desbois, Seze, & Szejwach, 1982).

Table 3: Solar Model Variables. (NREL, 2015)



Element	Unit	Description
Clearsky DHI	Watt per square meter	- Modeled solar radiation on a horizontal surface received from the sky excluding the solar disk. **- This is assuming clear sky condition**
Clearsky DNI	Watt per square meter	- Modeled solar radiation obtained from the direction of the sun. **- This is assuming clear sky condition**
Clearsky GHI	Watt per square meter	- Modeled solar radiation on a horizontal surface received from the sky. **- This is assuming clear sky condition**
Cloud Type	Unitless	Obtained from PATMOS-X
Dew Point	Degree C	Calculated from specific humidity
DHI	Watt per square meter	Modeled solar radiation on a horizontal surface received from the sky excluding the solar disk.
DNI	Watt per square meter	Modeled solar radiation obtained from the direction of the sun.
GHI	Watt per square meter	Modeled solar radiation on a horizontal surface received from the sky.
Fill Flag	Unitless	'N/A': 0, 'Missing Image': 1, 'Low Irradiance': 2, 'Exceeds Clearsky': 3, 'Missing Cloud Properties': 4, 'Rayleigh Violation': 5
Snow Depth	meters	Source: MERRA
Solar Zenith Angle	Degrees	Angle between the sun and the zenith
Temperature	Degree C	Source: MERRA
Pressure	Millibar	Source: MERRA
Relative Humidity	Percent	Calculated from specific humidity
Precipitable Water	Millimeter	Source: MERRA
Wind Direction	Degrees	Source: MERRA
Wind Speed	meter per second	Source: MERRA

These tests must be passed for a cloud to be positively identified as being one of ten cloud types. Of the ten categories, one category is assigned to clouds that cannot be identified; these clouds are labeled as unknown clouds. All this information is obtained from various levels of satellite data and process through three different algorithms, all of which are beyond the technical scope of this work, but further relate the need for an improved model that is relevant to observations from the ground and uses numerical weather data and other features that are easy to obtain. An improved model might be less expensive to implement and generalize better to specific locations since the current models are very depended on regional satellite data.

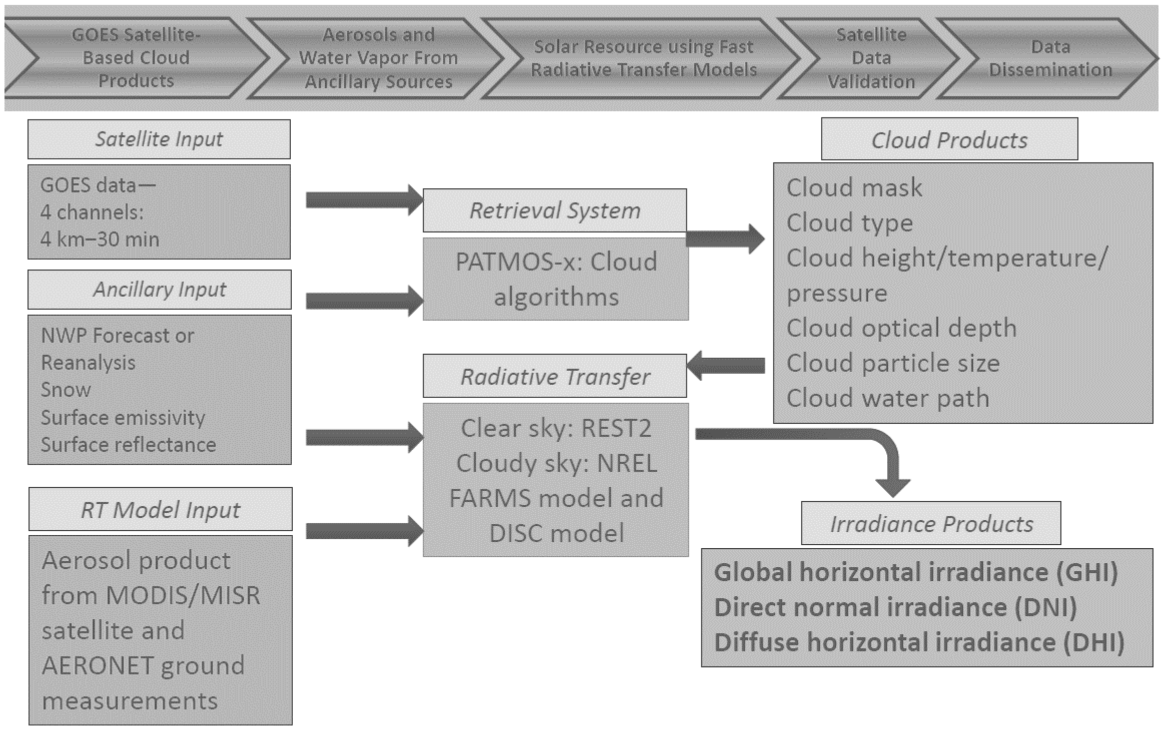

GOES Satellite-Based Cloud Products
Aerosols and Water Vapor From Ancillary Sources
Solar Resource using Fast Radiative Transfer Models
Satellite Data Validation
Data Dissemination
Satellite Input
GOES data— 4 channels: 4 km–30 min
Ancillary Input
NWP Forecast or Reanalysis
Snow
Surface emissivity
Surface reflectance
RT Model Input
Aerosol product from MODIS/MISR satellite and AERONET ground measurements
Retrieval System
PATMOS-x: Cloud algorithms
Radiative Transfer
Clear sky: REST2
Cloudy sky: NREL FARMS model and DISC model
Cloud Products
Cloud mask
Cloud type
Cloud height/temperature/pressure
Cloud optical depth
Cloud particle size
Cloud water path
Irradiance Products
Global horizontal irradiance (GHI)
Direct normal irradiance (DNI)
Diffuse horizontal irradiance (DHI)

Fig. 3: Physical Solar Model. (NREL, 2015)

The Solar PV model also contains four radiative transfer models (Clough, 2005), which are calculate the transfer of solar radiation through the atmosphere. These models are essential to the numeric modeling of solar irradiance; two of the four models are relevant to this work in functionality and the features being used for predicting GHI. The Clear Sky Model, as described in Sengupta & Gotseff (2013), is used specifically to predict irradiance, illuminance and active radiation under cloudless sky conditions. Gueymard & Ruiz-Arias (2015) reviewed 24 radiative models for the application in 1-minute time interval predictions and found a Clear Sky Model refereed to as REST2 to perform well in all weather conditions; an identify that only two other models shared and one of the foundational reasons this work was implemented. In addition to its variation in use, REST2 also was the only model to outperform the Rapid Radiative Transfer Model for General Circulation Models (RRTMG), which uses many climatological physical principals in its Mathematical derivation of GHI amounts; many consider these physical principles used during modeling essential to the model's ability to generalize well to other locations in other climates (Iacono et al., 2008).

Like REST2 and RRTMG, the Cloudy Sky Model from the NREL was created to produce hourly solar radiation forecasts in all sky conditions (Myers, 2006). This method was based on the observation of cloud cover, which is a similar implementation found in this body of work, and aptly described the clouds as 'The Long-Standing Problem' (Myers, 2006), which describes the difficulty clouds introduce into prediction. This model was the first model to use the Octa in the quantification of sky occlusions and compare the measured amount to possible correlations to GHI (Bird & Hulstrom, 1981). This method is still used to classify sky conditions by the National Weather Service. The third model, the Direct Insolation Model, again by Bird & Hulstrom (1981), formed the basis for the prior two models described here and is used in combination with previous methods in the more recent forth model or the Fast All-sky Radiation Model for Solar applications (FARMS) in Xie, Sengupta & Dudhia (2016).

2.5. Machine Learning

GHI forecasting has been based mainly on mainstream statistical methods (Mathiesen & Kleissl, 2011), but machine learning is beginning to change and update the way things have been done. This is mainly because numerous quantitative and qualitative features that were previously unable to be studied can now be purposed and weighted directly in the creation of predictive models (Lauret et al., 2015). In addition, new tools and new methods are now available for experimenting with combining, eliminating or even recreating many of these elements to build a feature space that is better for building predictive models.

Alpaydin (2014) defined machine learning by first addressing the concept of an algorithm as being a sequence of instructions that should be carried out to transform an input to an output.

Restated, a set of rules or instructions relate a set of input variables to a related outcome. This is commonly described as the relationship between X independent variables and Y dependent variable(s), respectively, with the ability to adjust the priority or weighting of all variables as the data and/or observations change (Langley, 1988). It is this ability that makes machine learning appropriate for the use in hybrid-type approaches for predicting global horizontal radiance, but many different problems must also be addressed before the process can be finalized and an overall model produced. In addition, creating a model only addresses a portion of the overall machine learning workflow or process, which is outlined in Fig. 4.

Machine learning begins with a scope that outlines the project details and acceptable project metrics before moving into the largest part of the overall process, which involves working with data. Data is often ‘dirty’ (or contains irrelevant and unusable information) and must be ‘cleaned’ prior to creating an optimal feature space that will be used for model building.

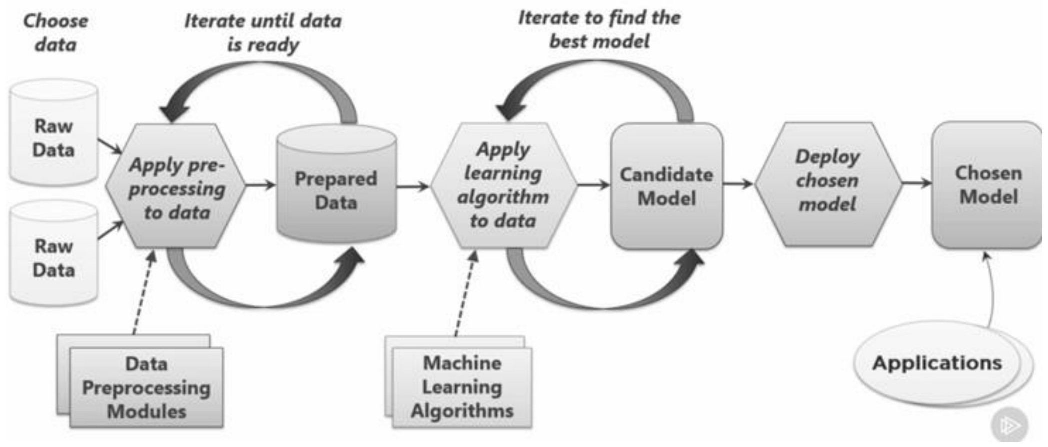

Choose data
Iterate until data is ready
Iterate to find the best model
Raw Data
Raw Data
Apply pre-processing to data
Prepared Data
Apply learning algorithm to data
Candidate Model
Deploy chosen model
Chosen Model
Data Preprocessing Modules
Machine Learning Algorithms
Applications

Fig 4: Machine Learning Process. The Machine Learning Process. (2016)

In addition, a specific type of problem must be matched with a specific machine learning tool(s); both decisions need to be made early in the process. In principle, some machine learning algorithms are sensitive to different types of problems such as problems only containing numeric data or problems containing mixed data consisting of both numbers and categories. Some algorithms, again in principle, function optimally when presented with problems that are in a specific structure or format like comma separated values or test files.

After cleaning the data, a very detailed exploration of the data is performed. This is necessary to make proper decisions about which features to using the feature space and which features to remove, if any. In addition to making these decisions, it is important to understand the relevance of the problem and the nature of the scope as well as the general makeup and type of data that are used in the problem as this information is directly related to the success and optimization of any models built from it. During this process data is often transposed, substituted and even removed to prepare the data for the chosen algorithm that will be utilized during training or when the algorithm learns the rules from the patterns in the data it has been used on. After models are trained, an assessment occurs and either a chosen model is accepted for use in making predictions or the process begins again with another set of algorithms that might be more applicable to the data. A specific model is then chosen and its generalization capability assessed and either then deployed into production, which means using it to make predictions in an external process, or rejected and the retraining begins using different algorithms or using different features (Michalski et al., 2013).

2.6. Data and Problem Types

Machine learning problems are often divided into two categories, both of which are defined by the type of data of the dependent variable. If the dependent variable is the number or numeric, the machine learning problem is called a regression problem (Smola & Schölkopf, 2004). If the dependent variable is a category or class, the machine learning problem is called a classification problem (Bradley, 1997). Machine learning algorithms also can learn patterns in data that is unstructured, like text or multimedia data, and does not fall into either of the previous categories mentioned (Nguyen & Armitage, 2008). Variables for classification or regression problems often fall into one of two definitions: variables that are highly related and can be competently modeled or correlated and/or variables that assume no correlation or predefined relationships (Blum & Langley, 1997). The former case, which is most commonly referred to as parametric data due to its distribution or the measurement of how the data is spread, is often addressed with a specific type or subset of machine learning algorithms that are built to learn direct relationships between features and related outcomes (Silverman, 2018). The latter case addresses data that is most commonly referred to as non-parametric data; nonparametric machine learning algorithms can be used on either nonparametric data or parametric data (Domingos, 2012). While nonparametric machine learning algorithms can be applied to almost any type of data they are often considered by industry practitioners to be very slow and computational inefficient (Brownlee, 2016). In addition, while their output is generally considered to be widely optimized, many different types of nonparametric machine learning algorithms are often prone to overtraining or learning patterns in the historical data ‘too well’ and cannot generalize well when presented with new data that is absent of similar outcomes (Dietterich, 1995). This process is called overfitting and will be described in more detail later in this work.

Langley (1996) also describes the use of specific algorithms in machine learning that improve their performance with experience. For example, machine learning algorithms might use historic data in the form of not only GHI in various geographic locations but also corresponding historical weather-related variables such as temperature, wind speed, etc. Whether describing classification problems that utilize categorical dependent variables or regression problems that utilize numeric dependent variables, problems in machine learning are directly related by the way the models train on the type of data being presented and are broken into two training categories: supervised learning and unsupervised learning.

2.7. Supervised Learning

Kotsiantis, Zaharakis & Pintelas (2007) defined supervised learning as a machine learning process or method that utilizes and describes observations or instances that are provided with **known labels or outcomes** as shown in Fig. 5. Data used in supervised learning problems are broken down into two main sections that encompass all instances or observations. The first section is composed of the features or independent variables. The second section contains the dependent variable. These two sections might or might not be related to each other in some type of way; the main goal of machine learning algorithms is to *learn* how this relationship occurs, so it can be replicated with similar data absent of any outcomes (i.e. make predictions). This is done by training a model. To reduce any errors in the process and increase the accuracy of predictions being made by the models, machine learning algorithms use function approximation algorithms (Rasmussen, 2004). These machine learning algorithms train on historical data with the goal of improving the accuracy of a given function. A good example of such function is the linear separation between two different categories based on their features. Another example of type of scenario is centered

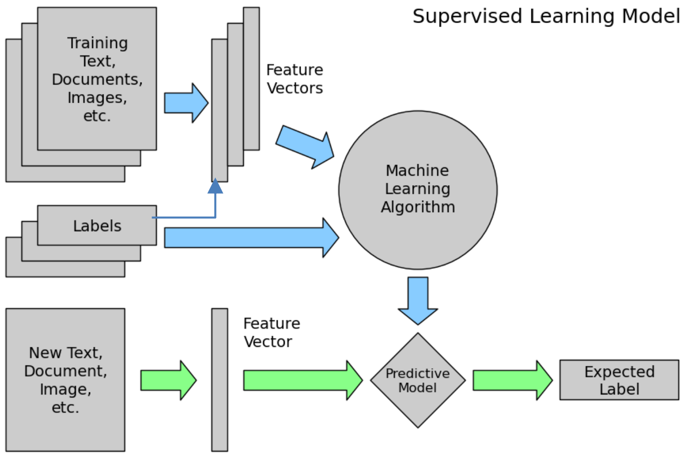

Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label

Fig 5: Supervised Machine Learning Process. (Polotan, 2015)

on minimizing the given function or set of functions within an algorithm as described by Jordan & Mitchell (2015).

In supervised learning, predictive models are built using data that are normally split into training sets and testing sets. During training, algorithms may experiment with different weights or priorities between the independent variables within the feature space and maintain random samples of data and combinations of features during experimentation all while trying to minimize some type of error and maximize some type of metric. After training concludes, any built models are assessed by a predefined or predetermined metric that corresponds directly to the type of problem an algorithm is being used to solve. Regression problems and classification problems have different types of assessment metrics; these will be covered later in the body of this work but regardless of the method, after a specific trained model is trained, it is then used with the test set to make predictions. At the end of this process the predictions made from corresponding rules learned by the trained model are compared to the known outcomes present in the testing set; these outcomes are commonly referred to as "ground truth" and serve as the major assessment point of success or failure for the predictive modeling process. If the predictions made by the trained model fall within a success metric window defined by the project scope when compared to the ground truth, the trained model is successful and can be used to generalize to outcomes with other known data features. In addition, any future predictions made will match the same success criteria that was previously assessed if the structure of the data does not change or the distribution does not deviate from any structure used during model training. Most predictive models do undergo some type of routine maintenance and are often retrained by repeating the process again with new data to account for this change in variation over time.

2.8. Unsupervised Leaning

While supervised learning is widely prevalent in GHI forecasting, a second type of machine learning has been used in cases where the data contains a feature space that does not have labelled or possibly even related outcomes. When data does not have a specific dependent variable, but rather a set of generalized variables that describe a general domain area in a quantitative or qualitative way, unsupervised learning algorithms can be used to ascertain related domain divisions that are relevant within the makeup of the data and component relationships that exist among the individual variables. As described by Fisher, Pazzani & Langley (2014), unsupervised methods must discover 'useful' categories within data using heuristics or predetermined search methods. Unsupervised learning will not be used in this work.

2.9. Overfitting and Underfitting

When predictive models are constructed, the main goal is to ascertain the optimal independent variables that will best predict the occurrence and value or class of a dependent variable. If this is accomplished, the trained predictive model will replicate or predict the dependent variable in the separate test set of data within some probability; this is the optimal scenario. According to Babyak (2004), overfitting yields findings that appear in a model that will not replicate from the trained model to the test data as shown in Fig. 6. Overfitting often occurs in many different types of machine learning models but can be especially prevalent where Euclidean Distance (Nasrabadi, 2007) is implemented in the training process or when elements of collinearity (Doorman, et. al, 2013) exist in regressions models. Overfitting (Hand, Mannila & Smyth, 2001) refers to a machine learning model that can model the data well, but cannot make predictions with the same performance obtained during training.

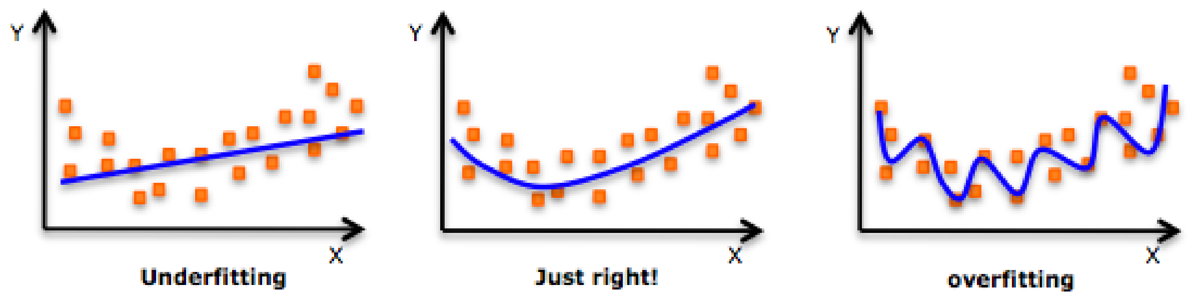


Fig. 6: Overfitting and Underfitting. (Regression, 2018).

This can be avoided by using methods such as Dimensionality Reduction or by reducing the size of the feature space and by using shrinkage, which is reducing the overall variance of the model by adjusting the values of its coefficients as described in Buehlmann (2006). Dimensionality Reduction is used in classification problems when redundant features are often removed to reduce the training size of the feature space and reduce possible bias that might be introduced by the redundant features. Underfitting is a much worse problem and often requires using a different algorithm as a solution (Domingos, 2012).

K-Fold Cross Validation is another method used to prevent overfitting; this method involves an automated process that partitions or folds the training set into various sections or folds wherein data is held out and never trained on (Batista et al., 2004). K represents the size of such equal sample or partition. The process is repeated K-number of times and each of the samples is used only once in the validation or testing data; an example of this process is shown in Fig. 7. The resulting output is then averaged to produce a single metric. Cross validation is also an acceptable way to partition ordinal data or data that follows a predefined order, this type of data present bias opportunities in machine learning algorithms that tend to over train on such types of structured or ordered data (Flach, 2012). This is caused by the folds being removed in different places within the training set thus removing any opportunity for bias caused by ordinal variables or the ordering of the data itself. In addition, since the samples are taken randomly this also increases the likelihood of additional stratification being added to the remaining data.

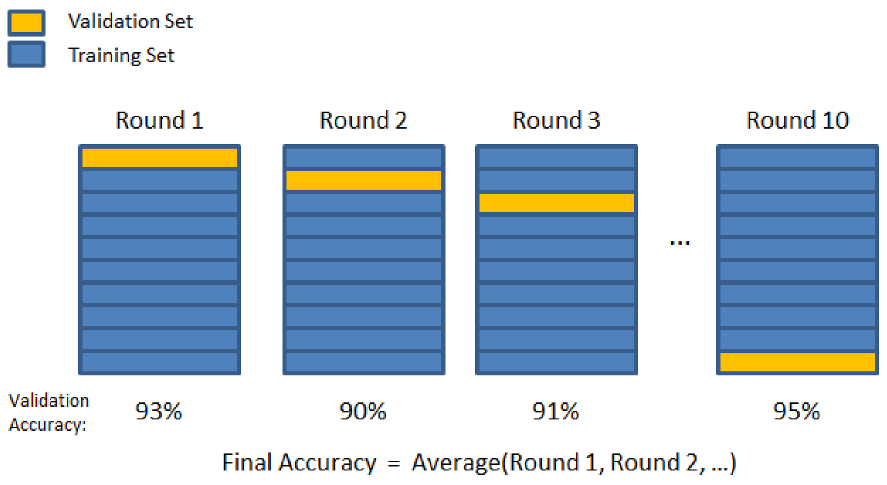


Fig. 7: 10-Fold Cross Validation. (Towards Data Science, 2017)

2.10. Model Assessment

Supervised machine learning models are assessed after training to check not only the performance of the model, but also to ensure the model did not overfit during training. If the trained model passes the assessment metric, it is used to make predictions and a second assessment is performed to gauge the ability of the model to generalize to the data or make predictions. Regression models are assessed with different metrics than classification models are. The degree of error and the ability to understand the amount of variation a model can understand in the data are the two main objectives that are measured in regression models (Mjolsness & DeCoste, 2001).

In regression problems, observations and/or predictions are compared to their central distance from a mean regression line as is shown in Fig 8. All regression problems contain some amount of error, which is assessed as the distance from each individual observation to the regression line.

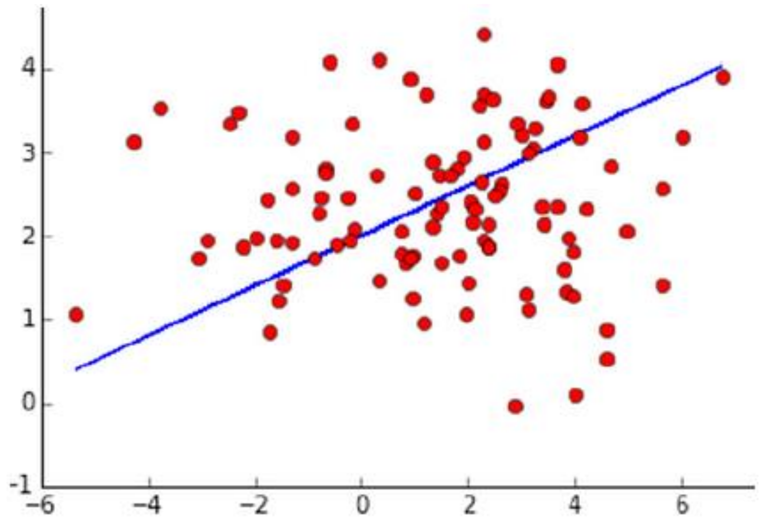


Fig. 8. Mean Regression Assessment. Devasthali (2018)

Mean Square Error (MSE) is the average measure of the squares of all the errors or the squares of all distances between the observations and the regression line (Wang & Bovik, 2009). MSE is calculated as follows where Ŷ represents the predictions and Y represents the ground truth or observed values:

This metric is also considered to be an assessment of the quality of the algorithm being used or its appropriateness to the type of data it is being used on. Mean Square Error, as shown in eq. (2), is used to measure individual models and how well they fit the type of data being used for training. For example: Parametric models are applied to parametric data because the algorithms used for training are sensitive to this type of data, but it is sometimes difficult to ascertain the type of data just from its distribution alone in the absence of domain knowledge and incorrect models are applied to incorrect types of data. High MSE values are seen with this occurs (Wallach & Goffinet, 1989).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{2}$$

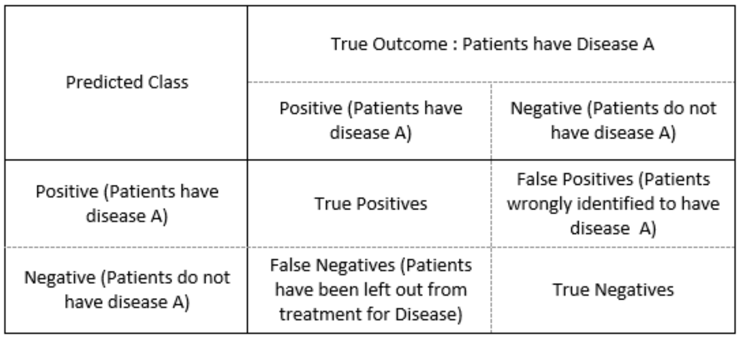
Mean Average Error or MAE is used to assess regression problems where the magnitude and only the magnitude of errors needs to be assessed (Bauer & Kohavi, 1999). This metric measures an average over the test sample of the absolute differences between the predicted values and the known values or ground truth. Where inherent outlier data can be expected, it is often common to see the square root of the average squared error taken; this is referred to as Root Mean Square Error (RMSE) and its value is in the same units as the dependent variable (Chai & Draxler, 2014). An additional metric used to measure error is Mean Absolute Percentage Error or MAPE, which is a measure of the accuracy of the predictions made by a regression models. This is done by comparing actual values to predicted values as a percentage of their difference and is shown in eq. (3) where A_t is the actual value and F_t is the predicted or forecasted value; n is the total number of observations and M represents MAPE:

$$M = \frac{100\%}{n} \sum_{t-1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \quad (3)$$

Lastly, the overall performance of the model and its ability to understand the variation of the data is assessed. This metric is called the Coefficient of Determination (R^2) and is used to assess the proportion of any variance present in the dependent variable that can be predicted from the independent variable. When graphing visualizations of regression models, a higher Coefficient of Determination is easier to see as observations are more tightly wrapped around the regression line. In addition, models that have a higher Coefficient of Determination have a lower error rate regardless of being assessed with MSE or RMSE. This is indicative of a better performing model The Coefficient of Determination can also be used to identify overfit or underfit (poor training) in a model. Coefficient of Determination is assessed in decimal values that range from 0 to 1. As the values increase, the “goodness of fit” of the model also increases, except in instances where the

Coefficient of Determination is 1. In this case, either the model has overfit or a pure correlation or one-to-one correlation exists between an independent variable and the dependent variable. In either case, the process would need to be re-examined to create a model that would generalize well with data that is presented to it later. Pure correlations may represent feature bias (Yu & Liu, 2004) in machine learning algorithms and the purely correlated feature (when related to the dependent variable) or features are sometimes eliminated so that the other features can be prioritized in the model (Kira & Rendell, 1992). Otherwise, a machine learning algorithm will likely treat a pure correlation as a single feature in the model due to the direct relationship it has with the dependent variable.

Classification problems have nominal data as dependent variables, or labels, these are generally assessed with ratios that measure the model's ability to correctly classify different categories against the errors made during the process. Classification model produce four different types of outcomes: true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). Confusion Matrices (Godbole & Sarawagi, 2004) as shown in Fig. 9 are normally utilized to fully understand the model's ability to derive the relationships and adequately build rules between the features; all four categories of outcomes from classification algorithms are shown in confusion matrices.



Predicted Class	True Outcome : Patients have Disease A	
	Positive (Patients have disease A)	Negative (Patients do not have disease A)
Positive (Patients have disease A)	True Positives	False Positives (Patients wrongly identified to have disease A)
Negative (Patients do not have disease A)	False Negatives (Patients have been left out from treatment for Disease)	True Negatives

Fig. 9. Example of a Confusion Matrix. (DNI Institute, 2016).

This method, which is also known as an error matrix, is a tabular format that allows one to ascertain the specific performance of a classification algorithm and the specific classes of the dependent variable the model incorrectly classified compared to the ones the algorithm correctly classified are labeled. This analysis is key to understanding the performance of a machine learning classification algorithm as many classification algorithms are prone to false positives (Kotsiantis et al., 2007) and these might be overlooked if a confusion matrix is not generated and analyzed properly.

Sensitivity and **specificity** (Table 4) are two additional metrics used to assess classification models; both are related to the four outcomes previously defined. Sensitivity measures a proportion of the correctly identified true positives and specificity is a measurement of the proportion of the actual negatives or true negatives that are correctly identified (Huang et al., 2012). Both metrics can assess how well a specific algorithm has performed on a given set of data since both are true assessments of real values and eliminate any confusion stemming from false positives incorrectly identified by the model. Tan & Gilbert (2003) used machine learning on gene expression data for cancer classification and observed cases where heathy genes were false positives or were incorrectly classified as cancerous genes.

Table 4: Sensitivity and Specificity Equations

Sensitivity= true positives / (true positive + false negative)
Specificity=true negatives / (true negative + false positives)

When assessing the overall performance of a classification algorithm, the measurement of Accuracy is used. Accuracy is a simple ratio of the number of correct predictions to the number of total predictions derived by a model (Pang, Lee & Vaithyanathan, 2002). Higher accuracy rates mean the model has a higher number of correct predictions versus the total number of predictions it has made. In cases where the dependent variable does contain two categories, Accuracy is often calculated in terms of positives and negatives and is shown in eq. (4) to be derived as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

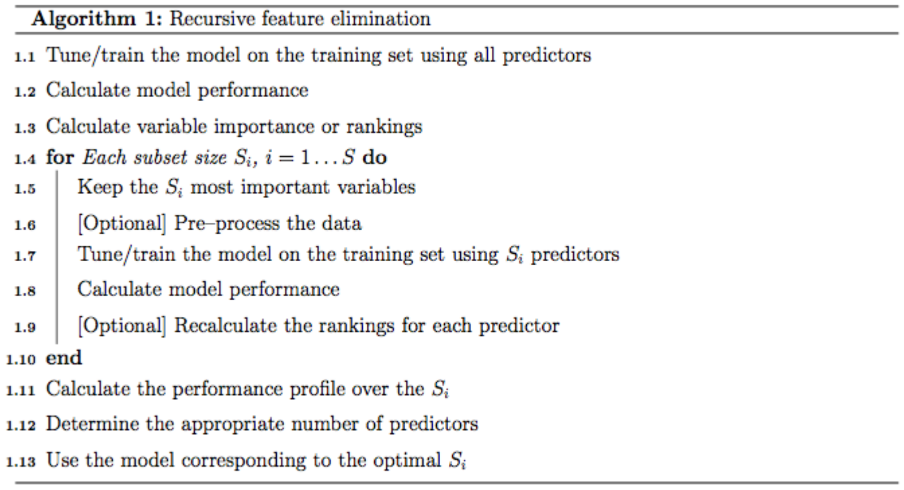
Cohen’s Kappa (García, Fernández, Luengo & Herrera, 2009) is used to assess a model’s ability to generalize well using new data, since it compares observed accuracy with an expected accuracy. This metric is used to evaluate or assess and compare the performance of algorithms used for training (Ben-David, 2008). ROC Curves or Receiving Operator Characteristic Curves (Davis & Goadrich, 2006) are also used to assess classification models and can be used to constrain a classifier that maximizes the true positives, while minimizing the false positives. Restated, the ROC score is a single numeric score that is equivalent to much of the information found in the Confusion matrix; higher ROC score are indicators of better performing models (Sokolova, Japkowicz & Szpakowicz, 2006)

2.11. Feature Selection / Feature Engineering

Regardless of the type of problem that machine learning is used to solve and regardless of the problem being addressed with supervised or unsupervised learning methods, the solution to good models is not found not in the outcome, but rather in the features (Witten et al., 2016). Features are central to the model building process and an understanding of which feature(s) should be in the feature space is key to model optimization and performance, which are both correlated to its ability to produce adequate predictions. Methods in this area of machine learning are in two categories: feature engineering and feature selection (Jade et al., 2003). Feature engineering is performed when specific algorithms are applied to the feature space to reduce its dimensionality, eliminate or alleviate certain features that might be redundant or might not provide any learning advantage. Feature engineering algorithms can also produce a new feature space based on highly correlated features that, when combined, can be utilized in by the model in a better way (Malhi & Gao, 2004). Feature selection is a simple process of deciding which features to keep and which features to eliminate. This is normally accomplished by running simple statistical tests to check for statistical significance or identify redundant features that can be eliminated.

Three definitive areas of feature engineering and feature selection are used in machine learning. Traditional methods, commonly referred to by Hall (2000) as filtering methods, use correlation for studying and better understanding relationships in both parametric and nonparametric data. Specific machine learning algorithms that include feature selection methods during training are called embedded methods and use filtering during the training process. According to Verzijlbergh, Heijnen, de Roode, Los & Jonker (2015), many embedded methods perform iterative experimentation by building numerous sample feature spaces and testing iteratively to assess what performance metric has been met. Wrapper feature selection methods are implemented when an algorithm repeatedly constructs a predictive model and chooses the best feature space by process of elimination (Fan, Xiao & Wang, 2014). Iteratively, a model is constructed with a given feature space of P; the model is then trained and assessed. The process, which is called Recursive Feature Elimination is shown in Fig. 10, then repeats itself with a feature space of P-1 features and reassesses the model's performance. This process is repeated until all the features are eliminated, or a given performance metric is met, for example, as shown by García-Hinde, et al., (2016) where specific features were extracted to solve a relevant GHI prediction problem.

Regardless of the type of features (meteorological related features, cloud imagery or hybrids) studies are beginning to develop and utilize specific algorithmic processes to optimize feature spaces in presented problems. While there are generalized heuristics concerning how the rules of Statistics can be applied to individual independent variables and their related combinations of such, these rules can often be combined and/or slightly manipulated to improve the resolution and sensitivity of a feature space, while not modifying data and/or skewing any related results.



```
Algorithm 1: Recursive feature elimination
1.1  Tune/train the model on the training set using all predictors
1.2  Calculate model performance
1.3  Calculate variable importance or rankings
1.4  for Each subset size $S_i$, $i = 1 \ldots S$ do
1.5      Keep the $S_i$ most important variables
1.6      [Optional] Pre–process the data
1.7      Tune/train the model on the training set using $S_i$ predictors
1.8      Calculate model performance
1.9      [Optional] Recalculate the rankings for each predictor
1.10 end
1.11 Calculate the performance profile over the $S_i$
1.12 Determine the appropriate number of predictors
1.13 Use the model corresponding to the optimal $S_i$
```

Fig. 10. Recursive Feature Elimination. Kuhn (2018)

Feature selection can be extremely powerful, for example, O'Leary & Kubby (2017) used machine learning algorithms in a similar method that included two levels of preprocessing and feature selection before training began. This pre-training feature selection process resulted in a 14.4% increase the accuracy, a 5.37% reduction in MAE and an RMSE reduction of 6.83% and was shown to be highly impactful in this case of regression.

When features are missing observation (Turrado et. al, 2014) or instances altogether, methods must be used to fill-in the missing details, so the data can be processed before training and testing can begin. While this is often considered an element of preprocessing that occurs before feature selection happens, there have been cases (Wang, et al., 2015) where feature extraction methods were derived to create new features to substitute for previously unknown variables that were relevant to the data but missing some type of label or domain description. This methodology resulted in new models that were more accurate (~4-5%) than models that contained the domain specific features that were provided just as they were captured and/or sampled.

While feature selection has been described previously in this work as being the prevalent factor impacting the outcome and performance of a predictive model, the machine learning algorithm is still on the training process, which is also dependent on such features. It is important to understand that there are numerous machine learning algorithms and all have specific use cases and work on all different types of data. In the case of GHI forecasting and prediction there are prevalent algorithms that have been used more than others and those are presented in this work in the next chapter.

CHAPTER THREE

LITERATURE REVIEW

This section will outline recent studies that are widely cited in this area of research; these are being introduced to not only provide more information about current GHI prediction methods, but also establish baseline metrics that will be used later in this study to assess the new hybrid method being presented here. Each study is categorized by the specific machine learning algorithm(s) that were used, all of which will also be used in this work.

3.1. Neural Networks

Neural networks, originally called Artificial Neural Networks (ANNs) (Schalkoff, 1997), refer to a type of algorithm that is modeled to function and process data like how neurons in the human brain process data. According to Haykin & Network (2004) the brain can easily be considered a nonlinear information processing system and thus an algorithm could be built to simulate how inputs are processed into related outputs.

Neural networks are algorithms that receive individual inputs, apply some weighted metric along with a transform to each input, and output the resulting combination for each instance. This process is shown in Fig. 11 and is used in the production of solar photovoltaic systems and in the forecasting of solar irradiance. Neural networks are categorized by their operational methodology or topology. Feedforward neural networks (FNN) are generally considered the simplest type of neural network and most other ANNs are similarly structured. In this ANN, data enters the first layer where the input variables are weighted and provided to the network. Data then moves linearly

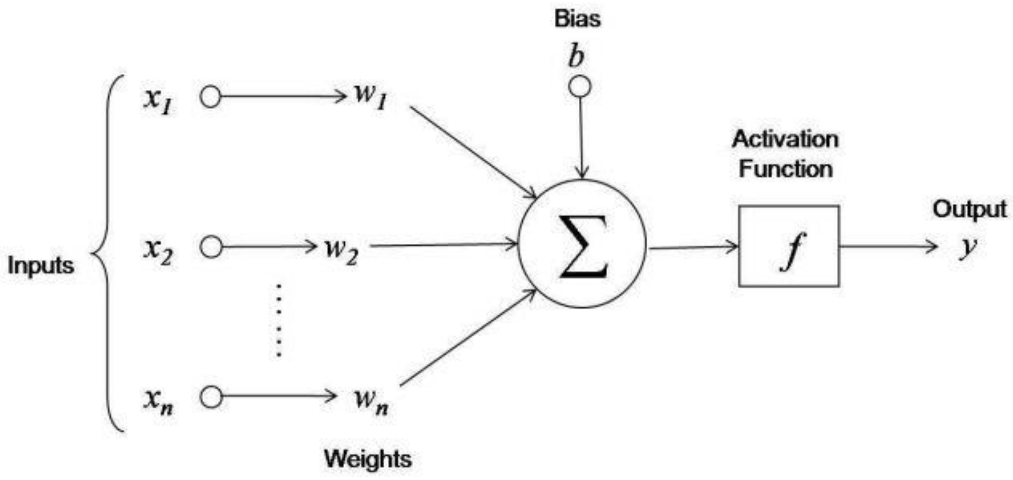


Fig 11. Artificial Neural Network. Castrounis, A. (2018)

through the next layer(s), called the hidden layer(s), where all the work inside a neural network is performed. In the hidden layer(s), weighted inputs are transformed using an activation function (Zurada, 1992). This function defines the output that will be delivered by nodes or sections in the hidden layer to the last layer or the output layer. The output layer contains the outcome of the neural network, which is then compared to the known value(s) or ground truth, in the data. The difference between these two values (predicted and known) is the error of the neural network. This occurs for every observation in a dataset and the goal of the neural network is to minimize the difference between its outputs and the ground truth. The process is measured and controlled by a loss function (Schapire, 2003) and the network will continue to optimize the training process to minimize the loss function. In neural networks that contain more than a single layer, the error can be reduced by sending it back through the network using an algorithm known as backpropagation (Adeli & Hung, 1994) as shown by the arrows in Fig. 12. A recurring theme in much of the academic literature surrounding GHI prediction methods and outlined here is most methods are implemented in chosen or specific weather conditions. The new methods being presented here will challenge this method by using data from all-weather conditions.

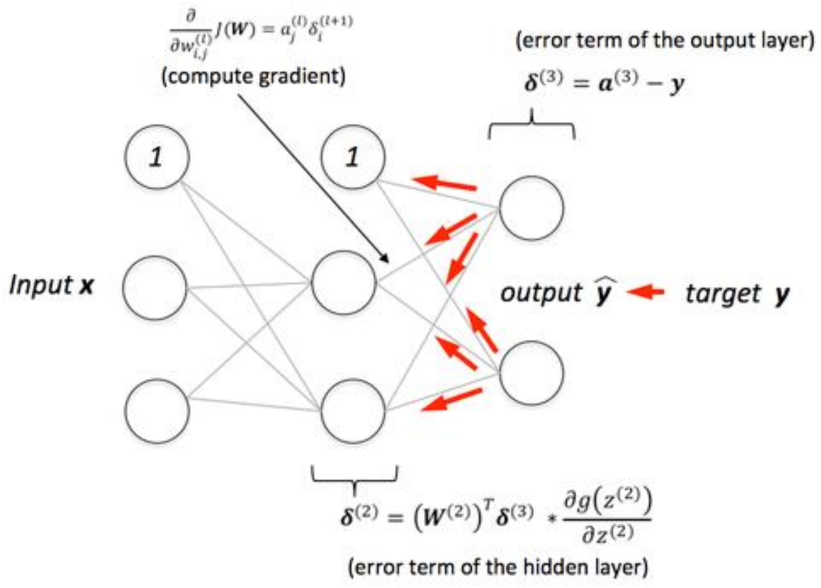

$\frac{\partial}{\partial w_{i,j}^{(l)}} J(W) = a_j^{(l)} \delta_i^{(l+1)}$
(compute gradient)
(error term of the output layer)
$\delta^{(3)} = a^{(3)} - y$
1
1
Input x
output $\hat{y}$
target y
$\delta^{(2)} = (W^{(2)})^T \delta^{(3)} * \frac{\partial g(z^{(2)})}{\partial z^{(2)}}$
(error term of the hidden layer)

Fig. 12. Backpropagation in a Neural Network. (Raschka, 2018)

Ding, Wang & Bi (2011) created an improved backpropagation method that positively impacted the twenty-four-hour prediction capability of an artificial neural network-based approach used in GHI forecasting. This weather data, which served as the inputs to the neural network, utilized historical power obtained from a local 15 kW photovoltaic array. The data used in the study consisted of historical and forecasted weather from a local public weather forecasting website and studied rainy, sunny and snowy days as shown in Fig. 13. The design of the neural network contained one hidden layer, which was designed by trial and error per Bahman (1998). To properly validate the performance of the model and the accuracy of predictions, historical forecasts from one sunny day and one rainy day were used for testing and historical power data was used for validation and comparison. Predictions for the sunny day forecast fell within 10% of the ground truth and predictions for the rainy day fell within 20% of the ground truth. The study considered the sunny day to be more accurate because of the low variation in photovoltaic output on sunny days; the **average error (MAPE) of all predictions was 14.75%.**

Date	Weather Type	High Temperature	Average Temperature	Low Temperature
2010.10.01	Sunny	33	22	11
2010.10.23	Rainy	17	12	7
2010.10.27	Snowy	14	7	0
2010.04.18	Sunny	26	15	4
2010.07.07	Sunny	38	27	14
2010.09.20	Sunny	23	17	11
2010.11.25	Sunny	7	1	-6

Fig. 13 Three Weather Conditions Studied in Ding, Wang & Bi (2011)

Amrouche & Le Pivert (2014) implemented a process using ANNs at two European locations, INES (French National Institute of Solar Energy) and Cadarache, FR, for one-day ahead solar energy so a photovoltaic connected system could generate and use daily weather forecasts to build the feature space used for training the ANN. This was done so the process could be replicated at any location where similarly structured forecasts were available, although a new predictive model specific to each location would need to be constructed. As conveyed in their study, any single model for a location would likely not generalize well using features used from another location. This study divided the model design and construction into three separate phases surrounding the proposed forecasting method shown in Fig. 14.

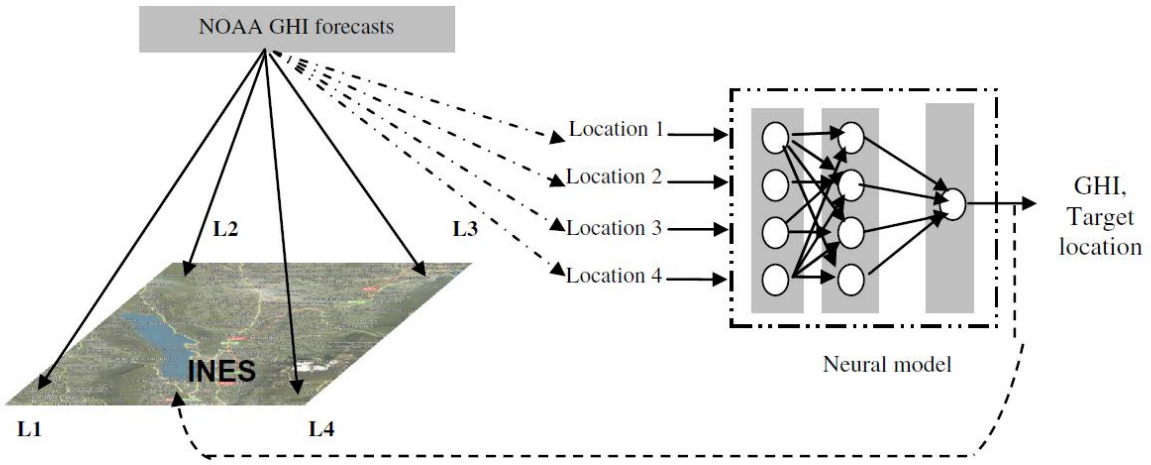
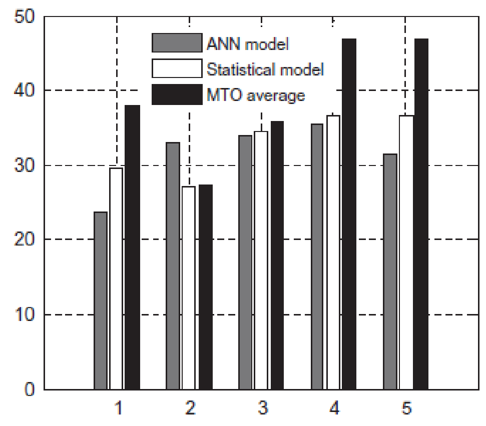


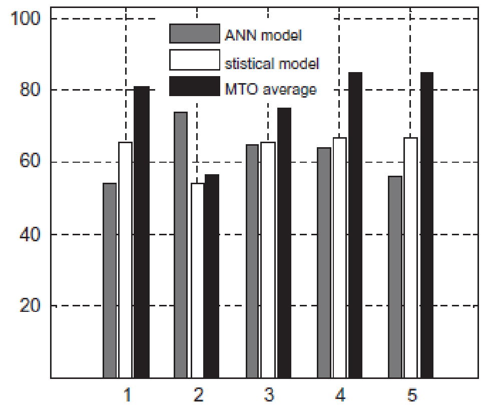
Fig 14. Model Design and Construction Phases. Amrouche & Le Pivert (2014)

The first phase included model choice and topology construction, a second phase for training and the final stage for validation, where the predictions were applied to a new set of data absent of outcomes. During the first phase, GHI only and GHI with ambient temperature forecasts were features used as inputs. The final model design choice was based on the results from models that used either a combination of these two features or either single feature individually in GHI forecasting. The final model design choice was a multilayer neural network with two hidden layers containing 20 and 12 neurons each; these would be used for weighting and backpropagation. This was a progressive approach where the ANN began training on a much weaker dataset and gained more knowledge as the training continued; this method is very similar to Recurrent Neural Networks (RNNs), which can train in both directions and will be described in more detail in a later section. Root Mean Square Error and MAPE were the metrics used to assess the training; these metrics were compared to a prevalent statistical model (Le Pivert, Sicot & Merten, 2009). Fig. 15 illustrates the metrics for each location (INES and Cadarache), which compared the MSE at each validation or testing period of the ANN to the best local statistical model and shows the ANN outperformed other methods, including Geometric Modelling (another statistical method), with an **average error (MAPE) of 22.5% and an RMSE of 51.5 W/m^2** when used for intraday and day-ahead forecasting.

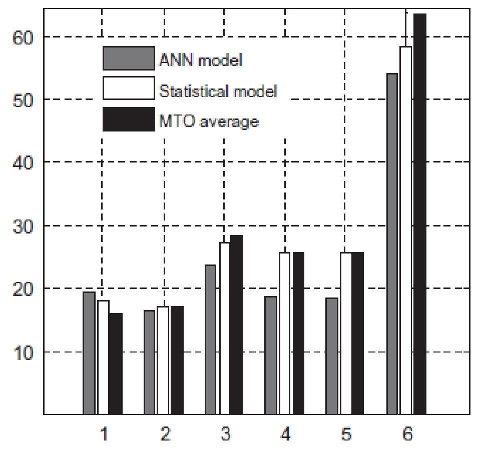
Khatib, Mohamed, Sopian & Mahmoud (2012) presented work where model performance was degraded in poor solar radiation conditions; they derived an approach for using ANNs to predict both hourly solar radiation and diffuse radiation using four different topologies (or designs), but only Elman Backpropagation Neural Networks (ELMNN) Nolfi, Parisi & Elman



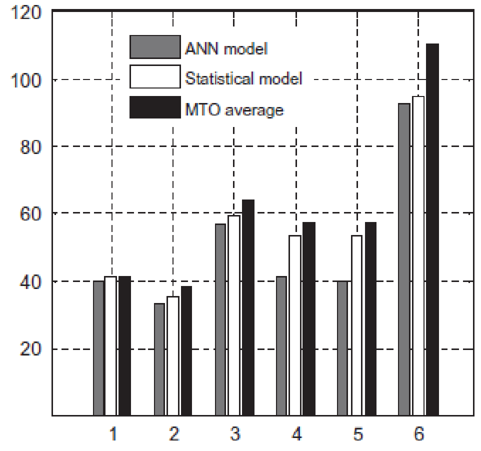
(a) MSE evolution during validation periods for progressive learning scheme (INES)



(b) RMSE evolution during validation periods for progressive learning scheme (INES)



(c) MSE evolution during validation periods for progressive learning scheme (Cadarache)



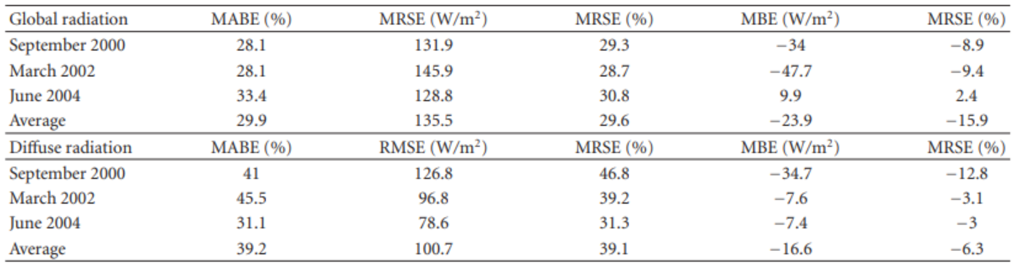
(d) RMSE evolution during validation periods for progressive learning scheme (Cadarache)

Fig 15. MSE per Iteration by Method. (Amrouche & Le Pivert, 2014)

(1994) were specific to the work being done here. ELMNNs are also very similar in makeup to RNNs and have the ability during training to use information recursively from previous iterations. ELMNNs also have a separate layer for gaining learning from previous contexts.

Inputs included eight geographical and climate related variable such as hour, day, longitude, latitude, etc. with two outcomes being estimated: hourly radiation and diffuse radiation. ELMNNs must be designed with a feedback layer (Baharin, et al., 2013) to ensure output from previous learning is transferred to the next iteration of learning, but this context must be present in the data to be of any use to this type of method. This is another similarity these networks have with the RNNs and this will be described in greater later in this section. As shown in Fig. 16, the performance of ELMNNs in this study was assessed using RMSE for DNI **and averaged 135.5 100.7 W/m^2.**

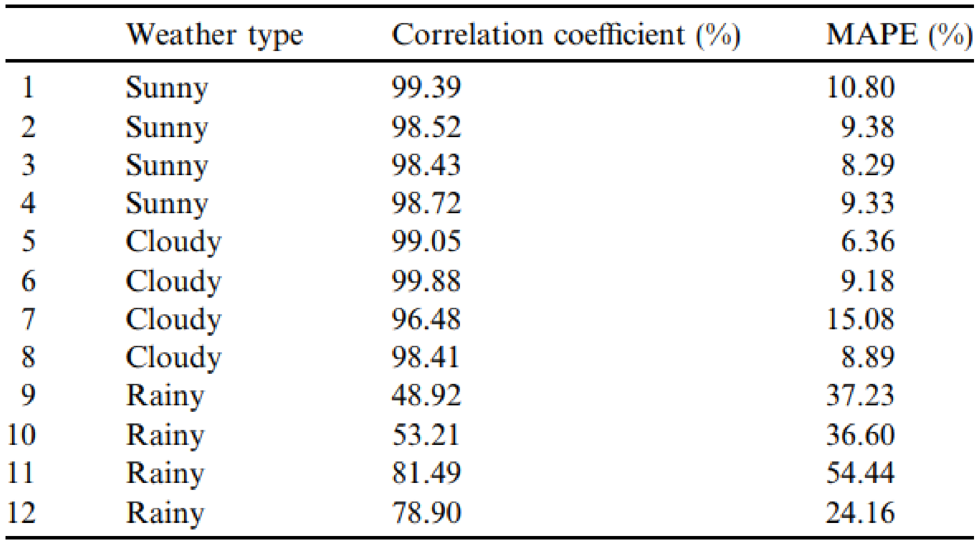
Chen, Duan, Cai & Liu (2011) derived an advanced statistical method for solar power forecasting that utilized past historical power data and historical solar irradiance forecasts, along with relative humidity and temperature, to build a predictive model for 24-hour ahead forecasting. In this study, a neural network was designed with three layers and a feature that utilized unsupervised learning in the hidden layer and supervised learning in the second layer. This ANN utilized a Radial Bias Function (Al-Amoudi & Zhang, 2000) (RBF) for activation, which is like to the Sigmoid transfer functions (Mubiru & Banda, 2008) utilized in many ANNs. This multilayer topology of this ANN is one of the reasons this study was included in the research as similar topologies will be used later in this work. Sunny, cloudy, and rainy days were used to validate the results of the model and create a comparison to ground truth where MAPE (Lewis, 1982) was used to ascertain the difference between the predictions and actual known values. The results are shown in Fig. 17 where rainy days were difficult for the models to predict; the average **MAPE for all days was 19.45%**. This is prevalent in the literature as most models don't perform well in all weather conditions and a solution for such will be addressed in the body of this work.



Global radiation	MABE (%)	MRSE (W/m^2)	MRSE (%)	MBE (W/m^2)	MRSE (%)
September 2000	28.1	131.9	29.3	−34	−8.9
March 2002	28.1	145.9	28.7	−47.7	−9.4
June 2004	33.4	128.8	30.8	9.9	2.4
Average	29.9	135.5	29.6	−23.9	−15.9
Diffuse radiation	**MABE (%)**	**RMSE (W/m^2)**	**MRSE (%)**	**MBE (W/m^2)**	**MRSE (%)**
September 2000	41	126.8	46.8	−34.7	−12.8
March 2002	45.5	96.8	39.2	−7.6	−3.1
June 2004	31.1	78.6	31.3	−7.4	−3
Average	39.2	100.7	39.1	−16.6	−6.3

Fig. 16: ELMNN Performance. (Khatib, Mohamed, Sopian & Mahmoud, 2012)

Yadav, Malik & Chandel (2014) implemented three ANNs (ANN-1, ANN-2 and ANN-3) in a similar manner to predict solar radiation, but also utilized feature selection methods from inputs consisting of geographical locations and meteorological data such as mean pressure, mean temperature, mean vapor pressure. In this manner, the ANNs trained and weighted the previously selected individual features that were related to the specific outcomes of each of observation.



	Weather type	Correlation coefficient (%)	MAPE (%)
1	Sunny	99.39	10.80
2	Sunny	98.52	9.38
3	Sunny	98.43	8.29
4	Sunny	98.72	9.33
5	Cloudy	99.05	6.36
6	Cloudy	99.88	9.18
7	Cloudy	96.48	15.08
8	Cloudy	98.41	8.89
9	Rainy	48.92	37.23
10	Rainy	53.21	36.60
11	Rainy	81.49	54.44
12	Rainy	78.90	24.16

Fig. 17: MAPE from RBF ANN. (Chen, Duan, Cai & Liu, 2011)

Temperature, maximum temperature, minimum temperature, altitude and sunshine hours were selected as the most relevant input variables using filtering methods. The study included data from twenty-six cities across different climate zones, which makes it an excellent candidate for understanding how training that uses vastly different climatological distributions might impact the overall generalization capacity of the predictive models used in different regions. The maximum MAPE for ANN-1, ANN-2 and ANN-3 models was found to be **20.12%, 6.89% and 9.04% respectively.**

ANNs are not just utilized in machine learning problems that are based off numerical features or predictors when forecasting Global Horizontal Irradiance as they are also used in systems that are constructed for cloud detection and measuring the impact different cloud types have on irradiance. The second part of this dissertation presents the design and implementation of a new image collection system and process that will provide the sky conditions needed for a new hybrid approach. This work was based off a smart adapter cloud identification system (Chu, et al., 2014) that and used a combination of sky images and solar irradiance measurements as inputs of an ANN forecasting model to develop and improve forecasting methods. A fish-eye camera captured images every minute during the daylight hours in the winter and spring of 2013 and processed the images with an image processing algorithm that was developed internally. Prior to processing, these images were selected manually and represented the following atmospheric conditions: clear, overcast, and partly cloudy; these would serve and the additional inputs for the ANN. The related model was tested and validated and produced an **RMSE of 77.5 W/m^2** as shown in Fig 18.

In the development of such systems, hybrid approaches are utilized to take advantage of current cloud detection methods and machine learning approaches that utilize quantitative weather features that are then used for training. In these types of systems (Marquez, Pedro & Coimbra, 2013), processed images gathered from satellite imagery and data from numerical weather models combine as features that serve as inputs used in ANNs to further extend the predictive capability of pre-existing GHI forecasting methods.

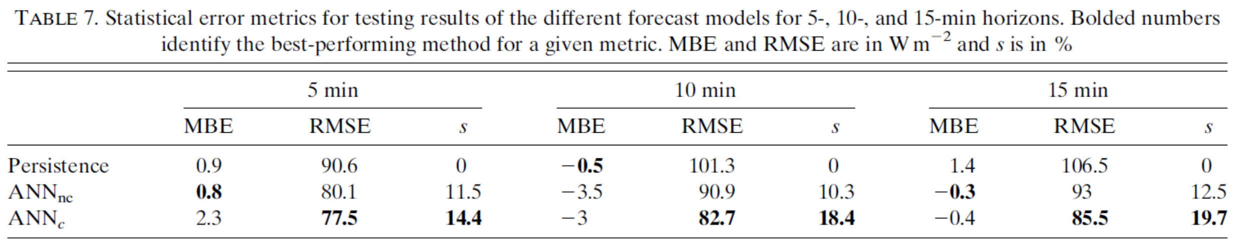


TABLE 7. Statistical error metrics for testing results of the different forecast models for 5-, 10-, and 15-min horizons. Bolded numbers identify the best-performing method for a given metric. MBE and RMSE are in W m^{-2} and s is in %

	5 min			10 min			15 min		
	MBE	RMSE	s	MBE	RMSE	s	MBE	RMSE	s
Persistence	0.9	90.6	0	**−0.5**	101.3	0	1.4	106.5	0
ANN$_{nc}$	**0.8**	80.1	11.5	−3.5	90.9	10.3	**−0.3**	93	12.5
ANN$_c$	2.3	**77.5**	**14.4**	−3	**82.7**	**18.4**	−0.4	**85.5**	**19.7**

Fig. 18: RMSE from Hybrid System. (Chu, et al., 2014)

While hybrid approaches have shown to be relevant and sensitive to both types of input data, there has been less research and study done in this research area, which is another reason this work was completed.

3.2. Support Vector Machines

Support Vector Machines (SVMs) were introduced in the mid-1990s and are considered (Lu, Lin, Jia & Tang, 2014) to be slightly more variable than Artificial Neural Networks when used in this type of application. Support Vector Machines can map data to a higher dimensional input or feature space (Suykens & Vandewalle, 1999) in a process called kernelling. This allows for easier separation of multiclass data or data with numerous categories as the data can be separated by the model so different classes no longer exist on the same plane. SVMs are also considered more accurate (Smola & Schölkopf, 2004) in nonlinear regression problems and problems with mixed data because of this kernelling ability. Most often used in binary classification problems (Hsu, Chang & Lin, 2003) as shown in Fig. 19, SVMs can also be used in regression problems like those found when using numerical weather data in GHI forecasting. Prevalent studies (Zeng & Qiao, 2013) have utilized Support Vector Machines in many different capacities that support GHI forecasting and prediction.

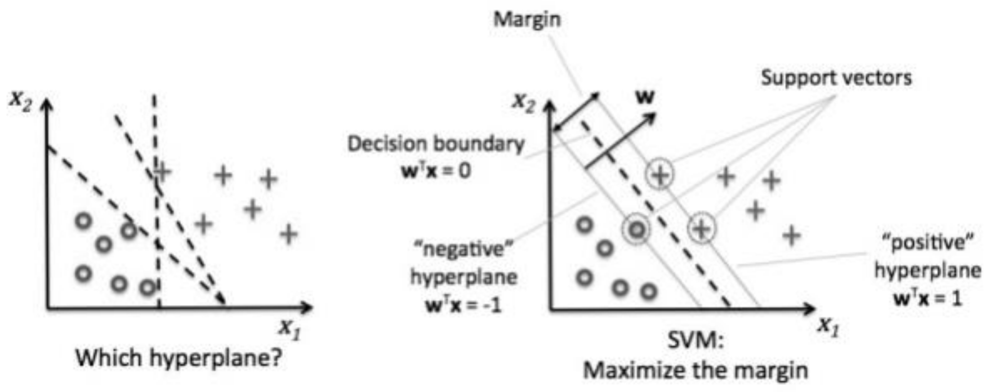
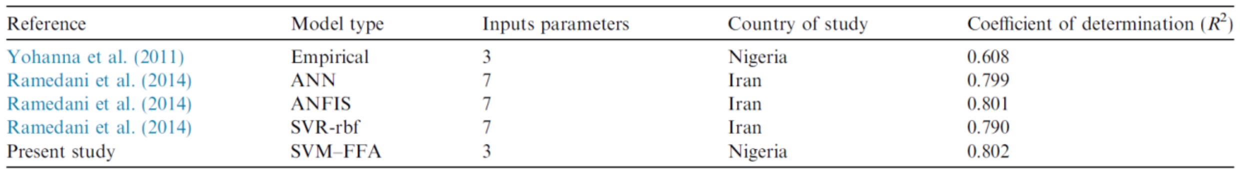


Fig 19: Support Vector Machines. (Hsu, Chang & Lin, 2003)

Some utilize this algorithm to predict different quantitative aspects of atmospheric conditions while other studies have utilized Support Vector Machines to predict the occurrence and amount of specific weather conditions like wind speed (Mohandes, Halawani, Rehman & Hussain, 2004), which has been shown (Borowy & Salameh, 1994) to greatly impact irradiance amounts due to its impact on regional weather conditions. Other hybrid studies (Olatomiwa et al., 2015) have utilized Support Vector Machines in a more hybrid type scenario to predict monthly mean solar radiation when combined with heuristic search methods like Genetic Programming and using sunshine duration, maximum and minimum ambient temperatures. This type of hybrid approach has shown to be very usable in this area when compared with results from ANNs and Genetic Programming (Koza, 1992) with RMSE, R^2 and MAPE values exceeding both the former by as much as 10% during training and testing assessments; training results (R^2) for the outlined model (SVM-FFA) are shown being compared to other cited models in Fig. 20.

Feature selection is also very prevalent in the use of Support Vector Machines. Other prevalent studies such as Piri et al. (2015) and Chen, Liu, Wu & Xie (2011) have shown that traditional and adequate feature selection methods that produce different combinations of experimental feature spaces have shown promise with assessments (Accuracy) as high as 96%. Atmospheric or meteorological values such as atmospheric transmission and/or wind speed are also predicted in this manner (Gill et al., 2006). Later in this dissertation, classification models will be used to optimize GHI prediction methods in a similar manner.



Reference	Model type	Inputs parameters	Country of study	Coefficient of determination (R^2)
Yohanna et al. (2011)	Empirical	3	Nigeria	0.608
Ramedani et al. (2014)	ANN	7	Iran	0.799
Ramedani et al. (2014)	ANFIS	7	Iran	0.801
Ramedani et al. (2014)	SVR-rbf	7	Iran	0.790
Present study	SVM–FFA	3	Nigeria	0.802

Fig 20. Model Training Metric Comparison. (Olatomiwa et al., 2015)

Shi, Lee, Liu, Yang & Wang (2012), which used classification methods to identify specific clouds related data in the feature space. In this study SVMs, were trained on specific types of cloud related images that were divided into categories such as clear sky, cloudy, foggy and rainy; a similar approach using GHI and images related to sky conditions will be outlined later in this dissertation. In other similar classification studies, such as Inman, Pedro & Coimbra (2013) and Martínez-Chico, Batlles & Bosch (2011), numerical features were often discretized into specific labeled categories to provide a less diverse training space for the algorithm and this same method was used here. In the latter study, specific percentages of cloudiness showed good potential for solar energy applications like the work being presented here.

3.3. Decision Trees

Decision trees are a type of machine learning algorithm that uses inductive learning (Quinlan, 1987) to quantify different patterns within data using iterative sets of binary branches as shown in Fig 21. These can also be used to uncover complex relationships and then move the inductive learning forward into a predictive model. Studies that utilize decision trees to forecast GHI often utilize secondary algorithms like K-Nearest Neighbor (McCandless, Haupt & Young, 2015) or even more traditional ANNs and SVMs to assist with feature selection.

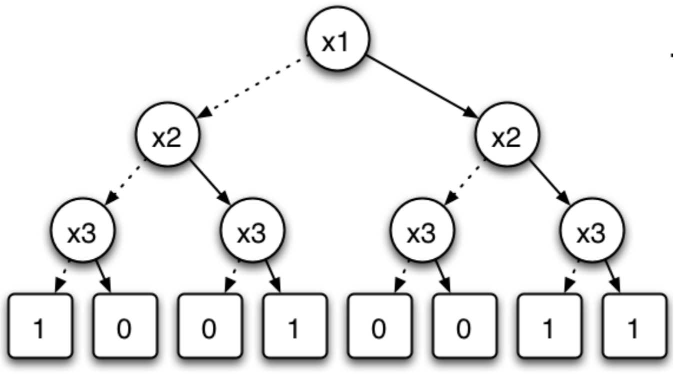


Fig 21. Decision Tree Branching. (Revolvy, 2018)

This type of combination yields a higher result than studies that have been presented that implemented a single algorithm in its basic capacity in shorter or near term hourly forecasting (Tso & Yau, 2007). These supplemental algorithms are used for external processes such as feature selection, which often optimizes the model(s) altogether in cases where this type of method is used Koprinska, Rana & Agelidis, 2015). In regression problems, near-term forecasts producing baseline metrics including RMSE have shown to have been lowered by as much as 31% (Hassan, Khalil, Kaseb & Kassem, 2017). This type of multifaceted approach has also been successful in short-term or next day forecasting where decision trees and additional algorithms have been used to optimize the feature space before training begins (Hong, Pinson & Fan, 2014).

Although approaches that utilize decision trees for cloud identification or prediction are not very prevalent due to limitations of the algorithm in image identification, some studies like Jiménez-Pérez & Mora-López (2016) have implemented decision trees and the use of clouds in a hybrid approach, like the work being presented here, for GHI forecasting. Studies like this are defined in two distinct phases: the first containing some type of definitive discretization, which is a categorized cloud type, and the second phase containing feature selection and model training based on inputs both from the feature selection phase and from the discretized inputs from the first phase (Banda & Angryk, 2009).

3.4. Linear Regression

In its simplest form, methods of linear regression derive almost the same implementation that basic machine learning methods do in that most methods combine a set of specific input values (feature space or predictors in machine learning) and relate these to a general output of a given known observation (i.e. the dependent variable). When a single input and a single output are utilized in the model, simple linear regression is implemented. Problems that require wider feature spaces with numerous input variables often employ multiple linear regression methods (Neter, Kutner, Nachtsheim & Wasserman, 1996). In either case, the basic form of the model resembles Fig. 22.

Linear regression models are limited to numerical features in most cases although nominal features can be used for training machine learning models if the classes are limited to binary levels (Montgomery, Peck & Vining, 2012). Since this data is mainly composed of quantitative values, linear regression models can quickly and easily be trained to build predictive solutions. Sharma, Irwin & Shenoy (2011) used historical data from National Weather Service forecasts and annual GHI readings to build a predictive, site-specific model for solar power generation. In this study, multiple machine learning techniques were derived using linear least-squares regression, which uses the sum of all errors in the problem as the optimization method, and found linear regression to be up to 51% better than similar approaches that only used past data along with existing sky conditions (Toğrul & Onat, 1999).

Edwards, New & Parker (2012) studied seven different machine learning methods including Linear Regression to potentially create better residential modeling methods using power measurements collected every 15 minutes. It is very important to note that the study did not examine GHI prediction as many of the previous studies outlined in this survey have but rather had the objective of predicting next hour residential building consumption. This work is included in this survey because consumption should be included in the overall process of understanding how to better predict the overproduction of energy. Since the problem is two faceted, it is essential to understand both methods: the first being energy prediction and the second being energy consumption. Another aspect of this study, was it included a comparison of sensor-based modeling techniques that monitored and reported data at predefined intervals with more traditional

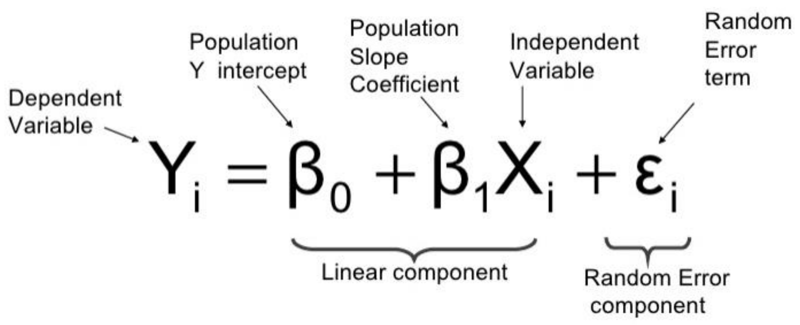


Fig 22. Linear Regression. (Towards Data Science, 2017)

supervised machine learning techniques. This comparison can be correlated to the two different methods of cloud identification that are used for GHI prediction since one method utilizes satellite imagery and the second utilizes actual data from the ground. Sensor data is more directly related to specific buildings in this study and can be more accurate than generalized or historic data that is taken from buildings of a similar design than from those not containing sensors. Model training and testing was done using an experimental set of data taken from a recently constructed residential development and compared to a recent energy production competition that is held annually by ASHRAE or the American Society of Heating, Refrigerating and Air Conditioning Engineers. The results of the study found that ANNs generally perform much better than traditional Linear Regression methods which is consistent other work as well (Li, Su & Chu, 2011). An additional advantage of Linear Regression modeling is its speed. In machine learning applications where it was used in larger scale regression problems (Collobert & Bengio, 2001), it is often considered the preferred option, but it is often limited due to its parametric nature. If correlation among the variables is not high, patterns in the data may be missed by this method.

While machine learning methods are the central focus of this survey, the highlighted usage of traditional Linear Regression statistical methods is being presented here because some have shown optimal results in studying cloud climatology. One study (Badescu & Dumitrescu, 2016) used Linear Regression to predict solar radiation availability at any time by building a comparison between global irradiance, cloud amounts and cloud types. The latter two inputs were provided by an existing facility and the solar irradiance measurements were taken for the same time periods from five different stations on the ground. The output of this comparison would create a better understanding of the potential impact cloud amounts and cloud types have on solar irradiance in different locations, which is same objective being outlined here. Statistical models were constructed for both clear and overcast skies and an existing heuristic outlined in Mueller (2009) was followed for generating solar irradiance values. Models predicting solar irradiance during cloudy sky conditions averaged R^2 values of 0.80, which can be considered much lower than solar irradiance models constructed using Linear Regression methods that utilize features and training (Ibrahim et al., 2012), but these types of models also generally do not include cloud types in the feature space(s).

3.5. Ensembles

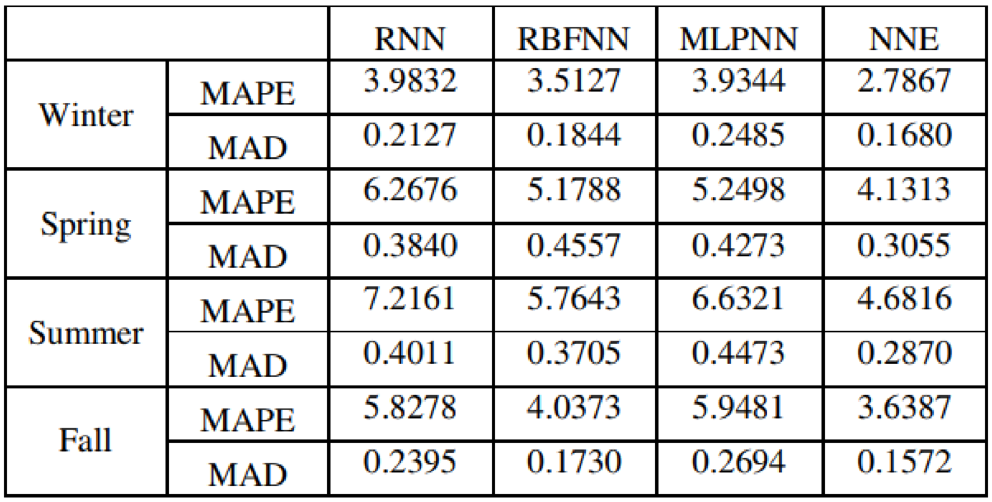
In studies that will be outlined in this section, processes and algorithms were combined in a cumulative approach called ensemble learning (Dietterich, 2000). Dietterich further describes ensemble methods as “learning algorithms that construct a set of classifications and then classify new data points by taking a weighted vote of their predictions.” This construction occurs iteratively and uses different methods that invoke building and comparing the results of numerous models during training. Random Forest (Sun, et al., 2016) is a good example of an ensemble learning method where multiple decisions trees (i.e. the forest) are created during training and the output is derived by taking the mean result of all the produced trees within the forest. Classification and Regression Tree (CART) algorithms perform iterative training like all ensembles, but CART algorithms build linear models that contain processes based on decisions and correlated extensions that contain the related regressive calculations (Lawrence & Wright, 2001). In CART methods, both classification and regression are used in one combined method.

Bootstrap Aggregation or bagging (Brieman, 1996), occurs when a training set of *N* examples is used to created multiple models of different samples of the data contained in the training set. A prediction can then be made using an average of the results of each of the previously created models. Bagging is considered an ensemble method because one of its main goals is to improve the overall model accuracy through iterative training. This is performed with the assumption that an average of errors on different samples provides a higher accuracy of the given learning method.

According to Friedman (2002), Gradient Boosting, constructs additive regression models by "sequentially fitting a simple parameterized function to current residuals by using the least square at each iteration." Restated, the model builds multiple decision trees like Random Forest does, but Gradient Boosting uses an arbitrary differential loss function instead of using the mean to make the resulting prediction.

Chaouachi, Kamel, Ichikawa, Hayashi & Nagasaka (2009) constructed a bagged ANN model for predicting short-term solar irradiance that consisted of a Multilayer Perceptron, an RBF Kernal, an RNN and an ensemble approach (NNE) that provided an average forecasting based on the other three models. The objective of this study was to create a reliable 24-hour-ahead solar power generation forecast. Model performance was assessed using forecasting error, which in this study was the difference in the actual and forecasted values when compared to statistical approaches or Mean Average Deviation; MAPE was also used to assess all three bagged models as shown in Table 5. Both error metrics were low for all three bagged methods and all three methods had the similar degree of error; this occurs when many ensemble approaches are utilized

Table 5: Bagged ANN Error Rates. (Chaouachi et al., 2009)



		RNN	RBFNN	MLPNN	NNE
Winter	MAPE	3.9832	3.5127	3.9344	2.7867
	MAD	0.2127	0.1844	0.2485	0.1680
Spring	MAPE	6.2676	5.1788	5.2498	4.1313
	MAD	0.3840	0.4557	0.4273	0.3055
Summer	MAPE	7.2161	5.7643	6.6321	4.6816
	MAD	0.4011	0.3705	0.4473	0.2870
Fall	MAPE	5.8278	4.0373	5.9481	3.6387
	MAD	0.2395	0.1730	0.2694	0.1572

because the base model topologies (designs) are similar in nature to begin with. This also further explains the use of ensemble methodologies as being used to not only optimize training, but also to further fine-tune smaller averages within the results.

Sun et al., (2016) took a less investigated approach using ensemble methods to predict solar radiation by creating different Random Forest models using meteorological data, solar radiation and air pollution index data, which is not used very often. The study utilized variable importance, which is an internal function used to ascertain the quantitative value of individual features. Some machine learning ensemble algorithms (again, Random Forest) can perform this function, but not all. Investigators uses variable importance here because they wanted to ascertain how the models treated or weighted the individual inputs to better understand any prevalence the related features might have on solar irradiance; the process is shown in Fig. 23. This study demonstrated that Random Forest ensemble models can be used effectively to predict the impact seasonal changes might have on of solar irradiance models and used RMSE as the metric for assessment. Output was measured in megajoule per square meter (MJ m^{-2}) with RMSE values in summer being 2.229

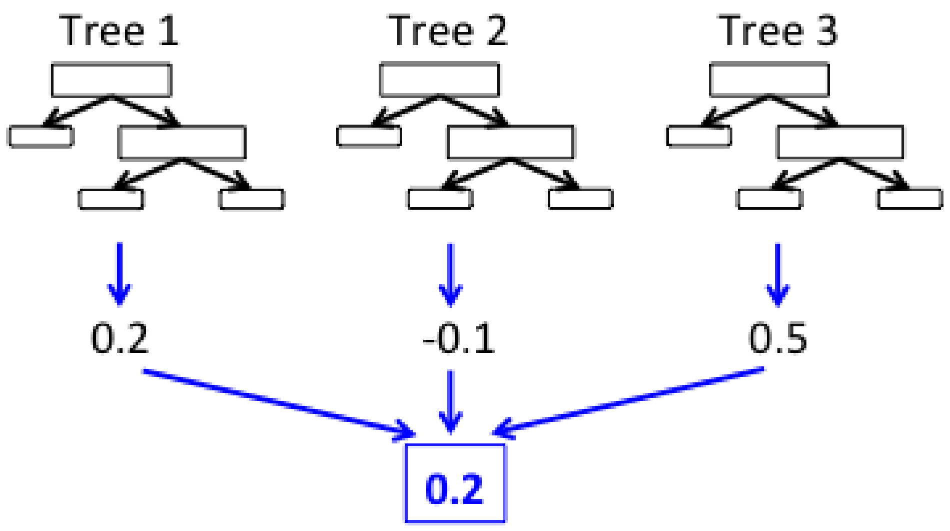
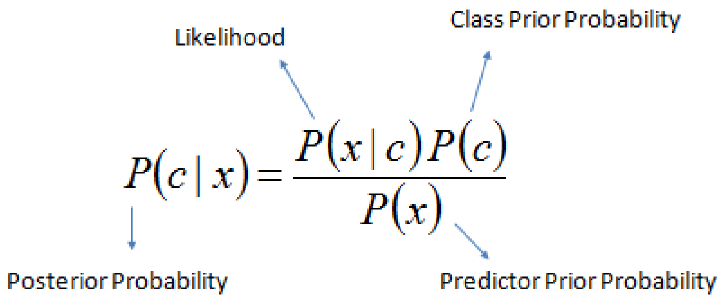


Fig 23. Random Forest Functionality. (Bradley & Made, 2015)

MJ m^{-2} and 1.189 MJ m^{-2} in the winter (around 500-600 w/m^{2}). Since Random Forest has the functionality needed to assess the feature importance, the model found sunshine hours, along with one feature that is prevalent in the pollution index, to be the most relevant features used for modeling. This finding can then be correlated to it being the most relevant features in the process externally when being used for predicting solar radiation. This study is another example of a hybrid approach being created using numerical weather features and externally collected real-time information, which in this case, came from the local pollution index. Later in the body of this work, the investigation of an externally collected feature will also be described and implemented.

3.6. Naïve Bayes

Machine learning algorithms can use the dependencies that exist between features to assign different weights or priorities for each for training, but there is a method that assumes that all the features within a given space are conditionally independent (Murphy, 2006). These types of algorithms are in a category of Bayesian classification and assign the most likely outcome to the most closely related feature Rish (2001). Although this is an unrealistic assumption, it is considered (Lowd & Domingos, 2005) an accurate practice and the implementation of naïve Bayes is prevalent in GHI prediction from the data consisting of prior observations as shown in Fig. 24.



$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

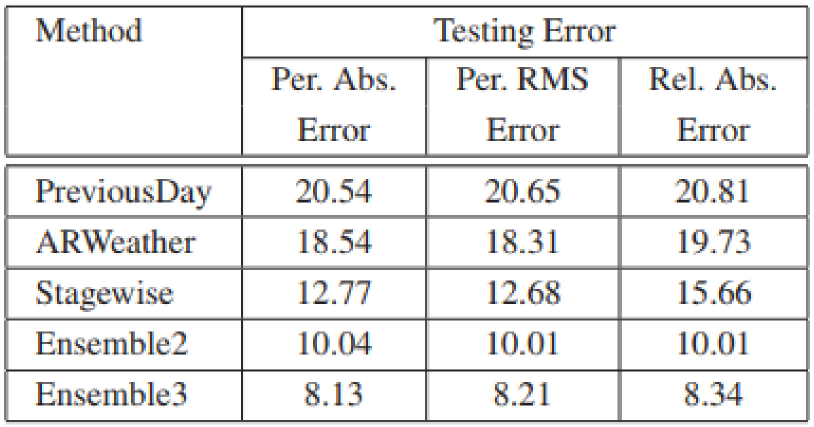
$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Fig 24. Bayesian Classification. (Naive Bayes, 2017)

In most Bayesian classification methods posterior probability is directly linked to some level of probability along with previous observations.

Chakraborty, Marwah, Arlitt & Ramakrishnan (2012) utilized a Bayesian ensemble-based approach for predicting photovoltaic output that included three different methods. The first method utilized a combination of Naïve Bayes and nearest neighbor methods, while the second method utilized weighted inputs that more closely captured variations between local and more global data. The third method used a more statistical approach using a Motif based (Snoek, Larochelle & Adams, 2012) process that took advantage of the sequential nature of solar photovoltaic power generation. Historic photovoltaic power generation data and available weather forecasting data was used for the feature space(s) and related outcomes. Previous day measurements as well as weather models and a Stagewise method (Hocking, 1976) like recursion, were used as baselines for validating the models' results. Training was implemented using predictors with the lowest error and the models were assessed using Percentage Absolute Error, Percentage RMSE and, as shown in Table 6, the Bayesian Ensemble (Ensemble 2) was only outperformed by the method using the Motif ensemble. Percentages are shown in the table to provide an understanding of the comparison between different methods, but non-composite values were not provided in this study. As Bayesian methods continue to progress in this area, more ensembles will continue to be utilized in similar manners and machine learning methods continue to be combined to produce better results than in previous studies and implementations. Heidinger, Evan, Foster & Walther (2012) used Bayesian methods for cloud detection using satellite observations and a high-resolution Radiometer as the inputs and outcomes. The relative purpose here is machine learning being used to handle building predictive models utilizing feature spaces that are derived both from weather-related features and from the output of Bayesian ensemble methods that included cloud detection and classification outcomes serving in the feature space.

Table 6: Training Error Metrics. (Chakraborty, Marwah, Arlitt & Ramakrishnan, 2012)

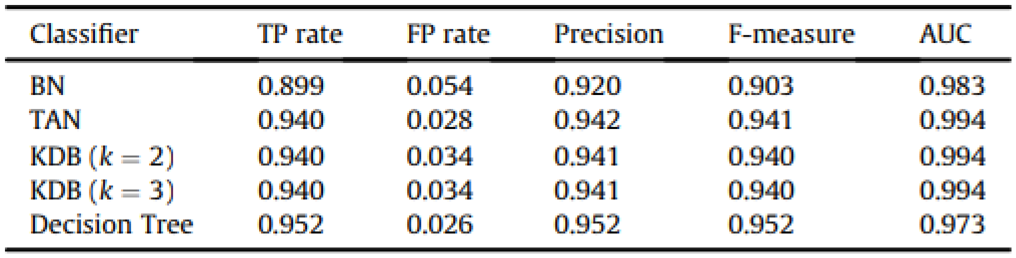


Method	Testing Error		
	Per. Abs. Error	Per. RMS Error	Rel. Abs. Error
PreviousDay	20.54	20.65	20.81
ARWeather	18.54	18.31	19.73
Stagewise	12.77	12.68	15.66
Ensemble2	10.04	10.01	10.01
Ensemble3	8.13	8.21	8.34

The resulting methods in this study provided probability of correct cloud detection metrics ranging from 70% to 90%, which can be generalized to much broader geographical area, unlike other methods previously discussed, which is an additional reason it was included in the survey. As has been previously discussed earlier in this survey most machine learning models in this area are constrained to certain geographical areas where the training data was taken from. Alonso-

Montesinos, Martínez-Durbán, del Sagrado, Águila & Batlles (2016) used Bayesian methods for cloud classification using satellite images to create the feature space used for training and different cloud genera (Rogers & Yau, 1989) as the related outcomes. The feature space utilized satellite channels, solar altitude, DNI and DIF and Accuracy was used assess model performance. Different Bayesian models were created including using Tree Augmented Naïve Bayes, k-dependence estimators (KDB) (Sahami, 1996) and a decision tree-based Bayesian model with the Tree Augmented Naïve Bayes (TAN) slightly outperforming all other models as shown in Table 7. While the previous studies in photovoltaics have utilized naïve Bayes successfully to adequately predict power generation, others like Aguiar et al., (2015) and English, Eyre & Smith (1999) have successfully used this same methodology for cloud classification in images taken from satellite observations.

Table 7: Accuracy rates for Bayesian Models. (Alonso-Montesinos et. al, 2016)



Classifier	TP rate	FP rate	Precision	F-measure	AUC
BN	0.899	0.054	0.920	0.903	0.983
TAN	0.940	0.028	0.942	0.941	0.994
KDB ($k = 2$)	0.940	0.034	0.941	0.940	0.994
KDB ($k = 3$)	0.940	0.034	0.941	0.940	0.994
Decision Tree	0.952	0.026	0.952	0.952	0.973

3.7. Deep Learning

While traditional methods have shown a wide variety of applications in cloud detection and identification, a relatively new area (Deng & Yu, 2014) of machine learning is beginning to show promise in both areas as well. This area of machine learning is commonly referred to as Deep Learning (Goodfellow, Bengio, Courville, & Bengio, 2016) and is based on a hierarchical framework of learning many different representations of data rather than just understanding and ascertaining rules and complex relationships like can be found more traditional methods. Like

ANNs, deep learning algorithms receive input, perform different weighting aspects and transforms and eventually develop an output (Schmidhuber, 2015). In the same linear form of an ANN, the process is linear in the beginning. but becomes different in the hidden layer(s) of the algorithm when the process involves deep learning. In the hidden layer of deep learning ANNs, iterative learning occurs in a hierarchical fashion that uses the information from subsequent hidden layers and trains the model based on the weights and results of the previous layer (LeCun, Bengio & Hinton, 2015). In this recurrent nature, hierarchical learning begins to develop as knowledge and continues to develop the process of passing from one hidden layer to another; a comparison of both methods is shown in Fig. 25. This iterative and recurrent architecture solves existing problems that occur when using ANNs in their typical architectural topology (Arel, Rose & Karnowski, 2010). This is done mainly by making previous learning rules that were constructed and assessed earlier in the network available to layers again later in the network. This type of recursive or recurrent learning makes the learning algorithms very sensitive to structure data that is sequential in form or follows a certain order and is also why deep learning neural networks are often referred to as Recurrent Neural Networks (RNNs).

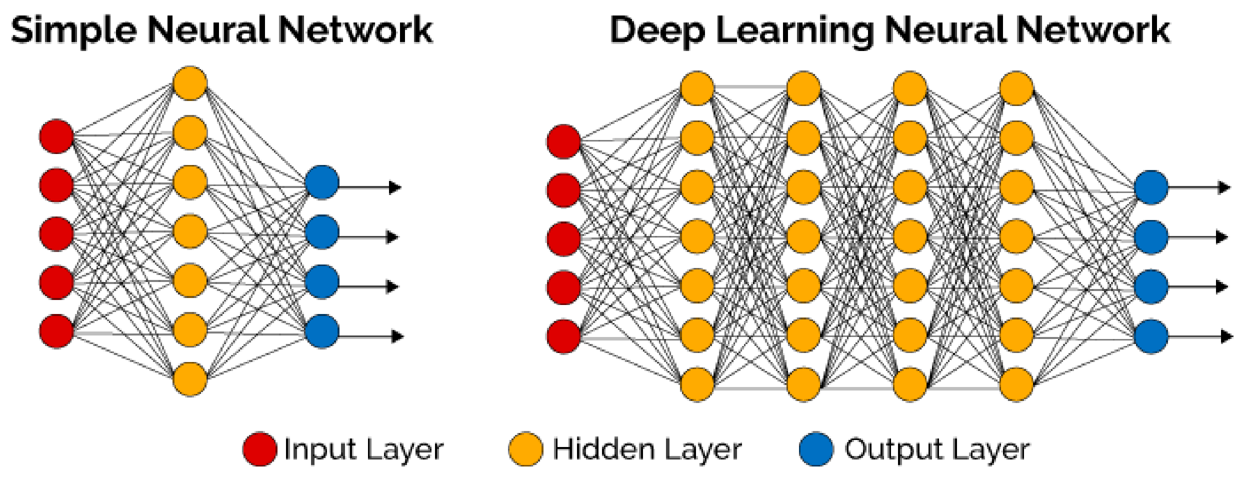


Fig 25: Deep Learning Architecture. (Deep Learning in Digital Pathology, 2018)

This recurrent and hierarchical characteristic also make deep learning methods widely used in image and object recognition and methods can easily be scaled to work on larger problems like those prevalent with feature spaces consisting of satellite imagery (Chu, et al., 2014) and/or imagery taken from the ground by whole sky imagers (Tapakis & Charalambides, 2013). This large-scale size also requires an improved computational backbone that is more capable than most modern CPUs which is why GPUs are most often used in larger image recognition tasks such as was done by Krizhevsky, Sutskever & Hinton (2012).

RNNs (Medsker & Jain, 2001) are sensitive to sequential data and do not only use feedforward functionality, but perform the same task for every element within the sequence. Each layer in the network is not only dependent on the output from the previous layer, but can use previously learned context from earlier in the training like the EMLNNs discussed prior in Khatib, Mohamed, Sopian & Mahmoud (2012). This process of sequential learning from layer to layer is shown in Fig. 26 where *X*$_t$ is the input at time step *t*, *S*$_t$ is the hidden state at time step *t* and *O*$_t$ is the output at step *t*. *W* represents the weight added at each individual node and *V* represents the output of each related node. Sequential information is preserved in the hidden state. As learning occurs, the network shares all its parameters, including all its weights, across all nodes; this ensures every sequential task can access the same depth and breadth of learning occurring elsewhere within the training.

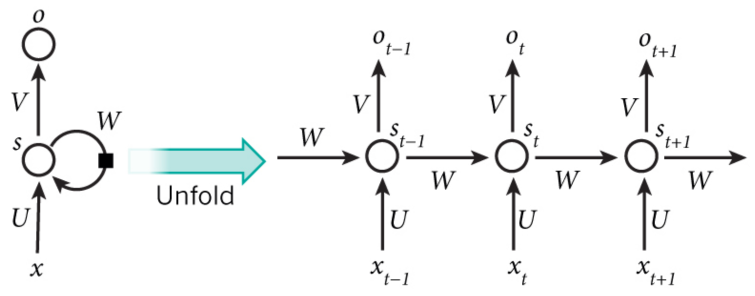


Fig. 26: Dissected Recurrent Neural Network Node. (Gupta, 2017)

As the network progresses, it finds correlations between events separated by frequency. These events are often referred to as long-term dependencies (Hochreiter et al., 2001) because downstream learning can depend on what has happened previously in the network recurrence. RNNs, like ANNs, also use backpropagation, but in RNNs it is often referred to as *backpropagation through time* (De Jeses & Hagan, 2001) for the added sequential functionality present in RNNs. As RNNs begin to train, each node tries to reduce the error of the previous node by applying newly learned or previously learns rules and optimizing any weighting necessary to optimize the function; this change is expressed by a gradient curve and is dependent on such optimization through the process. This is type of optimization is referred to as Gradient Descent (Ruder, 2016); an example is shown in Fig. 27.

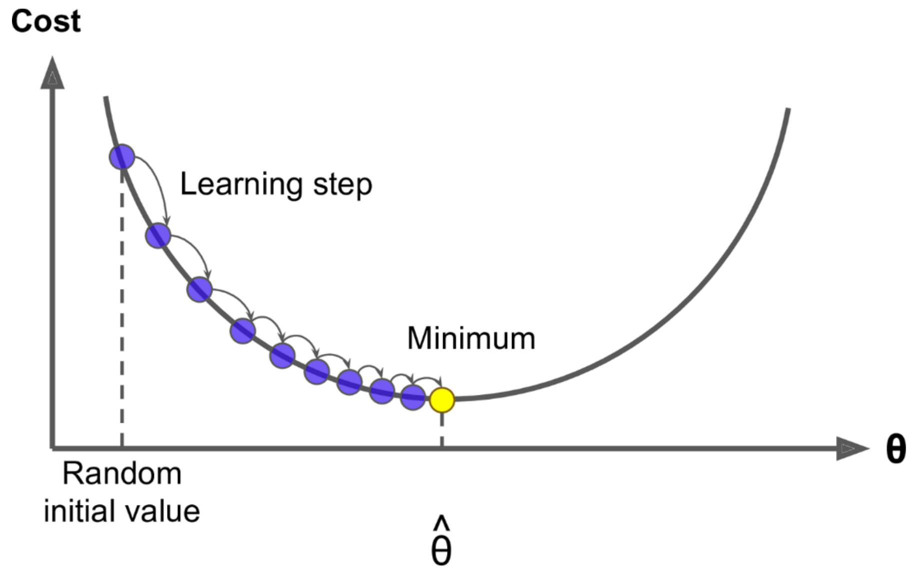


Fig. 27: Gradient Descent. (Hands-On Machine Learning, 2018)

Gradient Descent is widely used in machine learning algorithms and often used with RNNs (Andrychowicz et al., 2016) and continues to drive the error lower throughout each iteration until the function has been optimized and the loss minimized. This optimization method and the recursive architecture of RNNs can negatively impact model performance. During training, weights receive an update in each node that is related to the error reported by backpropagation. This is an iterative process and how the reduction of error occurs in neural networks, including RNNs. As this error reduction occurs, some of the error amounts returned through backpropagation are very small, yet continue as a catalyst for Gradient Descent. In cases where this occurs, the probability of the gradient no longer serving as a guide for the loss function becomes more of a problem until the gradient disappears and any optimization ceases. This is commonly called the 'Vanishing Gradient' problem (Jozefowicz, Zaremba & Sutskever, 2015, June). This issue is like overfitting described earlier, as the problem is directly related to model overtraining and can be prevented with and optimized network topology.

Cao & Lin (2008) combined an RNN and a wavelet neural network, which operates like a standard ANN except with a nonlinear activation function, as the base method for creating a new method they called a diagonal recurrent wavelet neural network (DRWNN). This new method was created for hourly and daily irradiance forecasting and was validated at two locations in China: Shanghai and Macau. The RNN used in the study was a diagonal recurrent neural network, which is a different type of neural network that uses backpropagation, but does not use connections between the nodes as functional RNNs do. This method optimizes training time due to a lower convergence (learning process) of the network topology. Historical climate records, date, time, cloud cover, aerosol and relative humidity were used as the inputs for the network and he study noted that cloud cover presented the strongest influence on GHI over the remaining features. Success measures were derived from two previously created models; the comparisons, along with a common regression line, are shown in Fig. 28.

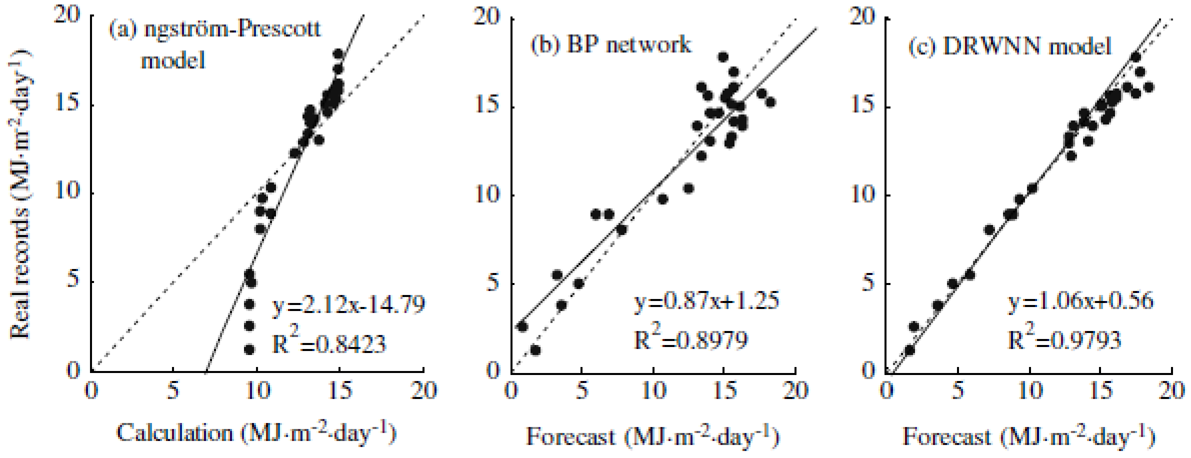


Fig 28. R^2 Values from Cao & Lin (2008)

Coefficient of Determination (R^2) values from the DRWNN were assessed to be .973, but these were validated on a very small sample (100 observations) of ground truth. In comparison to the previously discussed models, R^2 was high and possibly unrealistic for generalization in this type of application. This study was included in the literature review not only for its use of cloud cover in the feature space, but also to convey that some models presented in this area of study could be overly optimistic, but perform very well on smaller validation sets with low variance.

A Long Short-Term Memory Unit or LSTM is an RNN that uses a gated process or cell (Fig. 29) that can store information and make decisions about when to use the information (Poudel, & Jang, 2017). These cells open and close and are controlled during training by the activation functions of the network. As information is passed to these memory gates or cells, the cells decide whether to block the information or pass it along to another node or another cell (Srivastava & Lessmann, 2018). This decision is made based on the strength and priority of the information when it is received and it controls the potential amount of error passing back through the network. This methodology solves the vanishing gradient problem (Chung et. al, 2014) by preventing overtraining due to function ‘over’ optimization. If the function has less error to optimize, the gradient will not be in danger vanishing from attempting to minimize the loss.

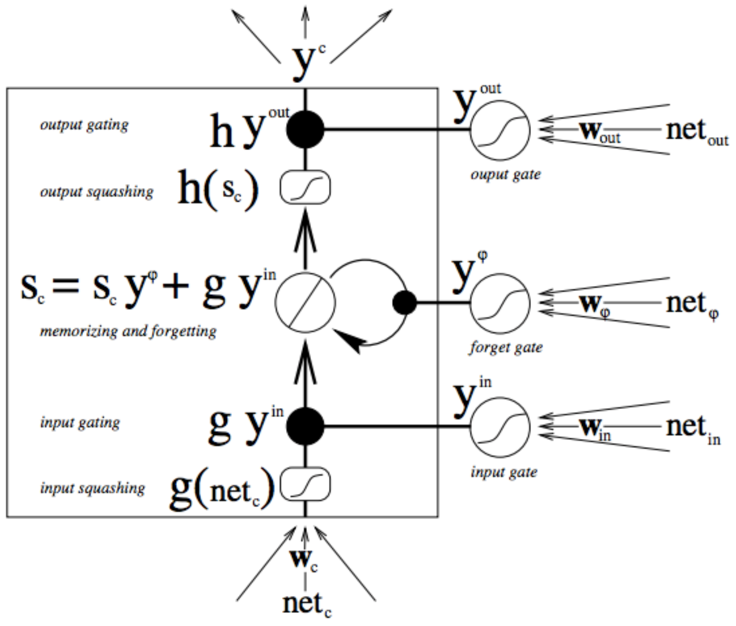
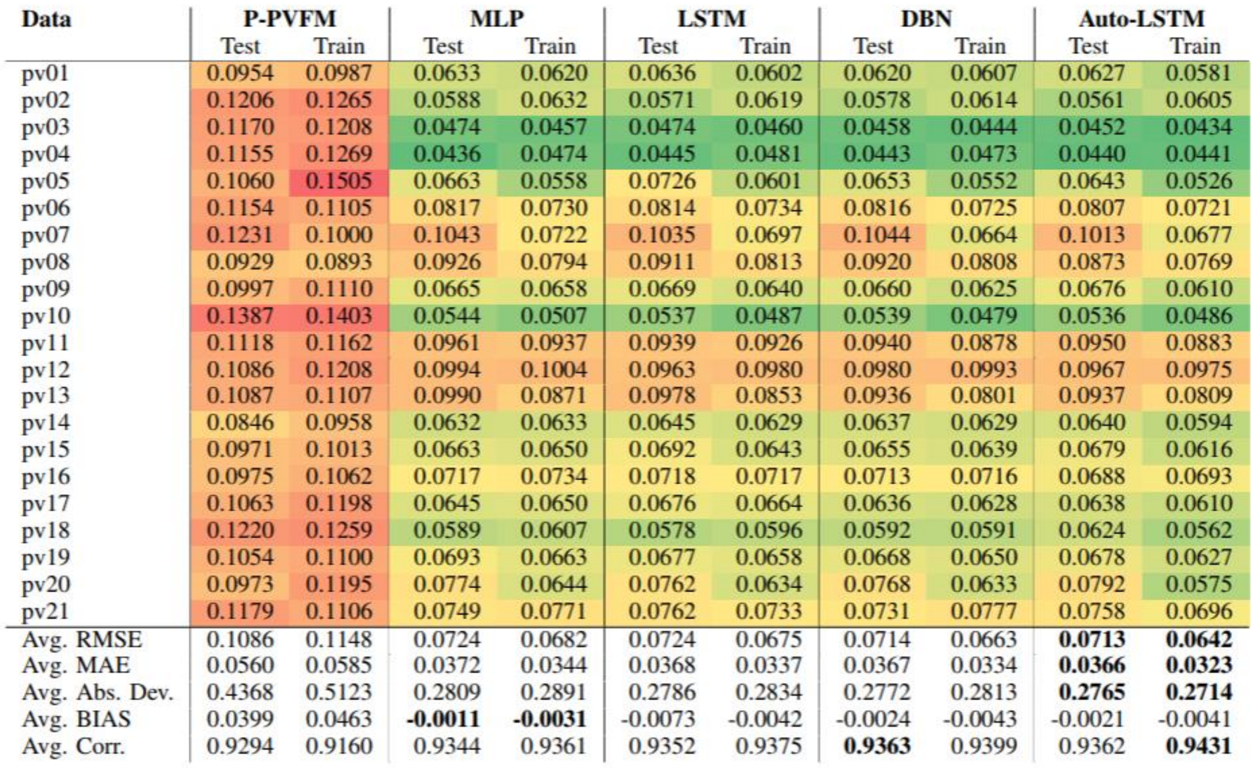


Fig 29. Cells in an LSTM Block. (Chen, 2016)

Gensler, Henze, Sick & Raabe (2016) utilized deep learning algorithms in the form of the Deep Belief Networks (DBN), an AutoEncoder (Witten, Hall & Pal, 2016) and an LTSM for power forecasting of renewable energy plants. This study focused on using numerical weather prediction methods to create a combined forecasting algorithm (Auto-LSTM). A physical photovoltaic forecasting model (P-PVFM) was used as a reference comparison and the data consisted of time series related data taken from twenty-one photovoltaic facilities was captured in a three-hour resolution over three years.



Data	P-PVFM		MLP		LSTM		DBN		Auto-LSTM	
	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train
pv01	0.0954	0.0987	0.0633	0.0620	0.0636	0.0602	0.0620	0.0607	0.0627	0.0581
pv02	0.1206	0.1265	0.0588	0.0632	0.0571	0.0619	0.0578	0.0614	0.0561	0.0605
pv03	0.1170	0.1208	0.0474	0.0457	0.0474	0.0460	0.0458	0.0444	0.0452	0.0434
pv04	0.1155	0.1269	0.0436	0.0474	0.0445	0.0481	0.0443	0.0473	0.0440	0.0441
pv05	0.1060	0.1505	0.0663	0.0558	0.0726	0.0601	0.0653	0.0552	0.0643	0.0526
pv06	0.1154	0.1105	0.0817	0.0730	0.0814	0.0734	0.0816	0.0725	0.0807	0.0721
pv07	0.1231	0.1000	0.1043	0.0722	0.1035	0.0697	0.1044	0.0664	0.1013	0.0677
pv08	0.0929	0.0893	0.0926	0.0794	0.0911	0.0813	0.0920	0.0808	0.0873	0.0769
pv09	0.0997	0.1110	0.0665	0.0658	0.0669	0.0640	0.0660	0.0625	0.0676	0.0610
pv10	0.1387	0.1403	0.0544	0.0507	0.0537	0.0487	0.0539	0.0479	0.0536	0.0486
pv11	0.1118	0.1162	0.0961	0.0937	0.0939	0.0926	0.0940	0.0878	0.0950	0.0883
pv12	0.1086	0.1208	0.0994	0.1004	0.0963	0.0980	0.0980	0.0993	0.0967	0.0975
pv13	0.1087	0.1107	0.0990	0.0871	0.0978	0.0853	0.0936	0.0801	0.0937	0.0809
pv14	0.0846	0.0958	0.0632	0.0633	0.0645	0.0629	0.0637	0.0629	0.0640	0.0594
pv15	0.0971	0.1013	0.0663	0.0650	0.0692	0.0643	0.0655	0.0639	0.0679	0.0616
pv16	0.0975	0.1062	0.0717	0.0734	0.0718	0.0717	0.0713	0.0716	0.0688	0.0693
pv17	0.1063	0.1198	0.0645	0.0650	0.0676	0.0664	0.0636	0.0628	0.0638	0.0610
pv18	0.1220	0.1259	0.0589	0.0607	0.0578	0.0596	0.0592	0.0591	0.0624	0.0562
pv19	0.1054	0.1100	0.0693	0.0663	0.0677	0.0658	0.0668	0.0650	0.0678	0.0627
pv20	0.0973	0.1195	0.0774	0.0644	0.0762	0.0634	0.0768	0.0633	0.0792	0.0575
pv21	0.1179	0.1106	0.0749	0.0771	0.0762	0.0733	0.0731	0.0777	0.0758	0.0696
Avg. RMSE	0.1086	0.1148	0.0724	0.0682	0.0724	0.0675	0.0714	0.0663	**0.0713**	**0.0642**
Avg. MAE	0.0560	0.0585	0.0372	0.0344	0.0368	0.0337	0.0367	0.0334	**0.0366**	**0.0323**
Avg. Abs. Dev.	0.4368	0.5123	0.2809	0.2891	0.2786	0.2834	0.2772	0.2813	**0.2765**	**0.2714**
Avg. BIAS	0.0399	0.0463	**-0.0011**	**-0.0031**	-0.0073	-0.0042	-0.0024	-0.0043	-0.0021	-0.0041
Avg. Corr.	0.9294	0.9160	0.9344	0.9361	0.9352	0.9375	**0.9363**	0.9399	0.9362	**0.9431**

Fig. 30: Error Comparisons for Deep Leaning Methods. (Gensler et. al, 2016)

Time series values were normalized and the dependent variable, which was the measured power output, was normalized using the normal output capacity of the given facility where it was recorded. The study used five different error metrics to assess the performance of the model; RMSE is provided as normalized values of W/m^2 and MAE is shown normalized percentage of prediction error and are shown in Fig. 30. Both metric averages were RMSE: .0713 and MAE: .0366 for the Auto-LSTM models and much lower (-.30) than the standard P-PVFM.

Alzahrani, Shamsi, Dagli & Ferdowsi (2017) experimented with solar irradiance forecasting with an RNN that used climatological and meteorological features as inputs and local solar irradiance measurements as outcomes. Support Vector Regression and a NN were also trained so comparisons could be made with more traditional methods of machine learning. Data was preprocessed using interpolated values and normalized to ensure a low probability of feature bias during training. An LSTM network topology containing two hidden layers and thirty-five neurons produced normalized RMSE and MSE metrics as shown in Table 8.

Table 8: Model Error Metrics. (Alzahrani, Shamsi, Dagli & Ferdowsi, 2017)

Method	**RMSE**	**MBE**
FNN	0.16 W/m^2	0.005
SVR	0.11 W/m^2	0.0042
LSTM	0.086 W/m^2	0.004

An additional sequential neural network used for image classification and will be used for such later in the body of this work is called a Convolutional Neural Network or CNN (Sun, Szűcs & Brandt, 2018). In 1968 two Neurophysiologists published a paper (Hubel & Wiesel, 1968) describing the path information travels from the eyes to the visual cortex. In this work, it was discovered that complex cells in the visual cortex were sensitive to an objects interaction within regions of the receptive field. These regions overlap and cover the entire surface of the visual receptive field and process what is seen by the eyes (Gilbert & Wiesel, 1992), but Hubel and Wiesel discovered that these regions, which are made up of cells, were sensitive to edges and curves. They derived this was the basis for understanding and processing images and context is not gained from content alone, but the visual cortex is processing millions of shapes and curves obtained from the images. The operational aspects of CNNs were patterned after this same process and are shown in Fig. 31.

CNNs are used for image classification to take advantage of the sparse connectivity between the network's neurons and adjacent layers (Yosinski et al., 2015). This spatial connectivity uses the inputs from one layer from the subset of inputs from the previous layer, all of which come from inputs from neurons serving as the receptive field of the network. When

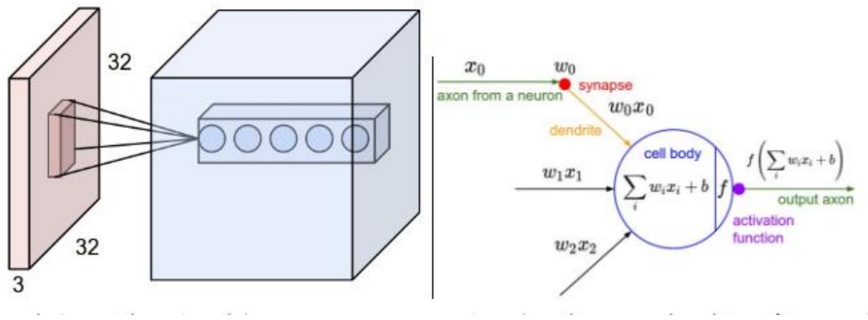


Fig. 31: Visual Cortex / CNN Receptive Field. (CNN From the Ground Up 2018)

CNNs use images as their inputs, the inputs are three dimensional and represent multichannel images as numeric arrays. The ‘Convolution’ portion of the network begins by applying a small matrix filter sequentially over each area of the image and extracting features (curves and edges) from the pixels represented in the image array (Zeiler & Fergus, 2014). As the convolution continues each filtered ‘snapshot’ of the extracted pixels represented by the extracted curves and edges is compared to the actual pixels in the previously scanned region; this process continues as a series until the entire image is represented by a set of layered feature maps (He, Zhang, Ren & Sun, 2016). Shi, Wang, Wang & Xiao (2017) tested using CNNs for ground-based clout classification and utilized to publicly available data sets for training and validation. As outlined in the study, clouds play an important role in GHI prediction but existing ground-based images are obtained by professionally trained personnel. Furthermore, the study stressed the need for “automatic and efficient cloud classification” similar to methods previously used by Buch, Sun & Thorne (1995) and Singh & Glennen, 2005). Five different CNN models were extensively tested on cloud image data to measure the viability of using this type of network for image classification. The results in Fig. 32 show that CNNs outperformed more traditional image classification methods (Heinle, Macke & Srivastav, 2010) where ‘FC’ outlined the number of folds and convolutional layers in the CNN and the highest Accuracy was assessed at 89%.

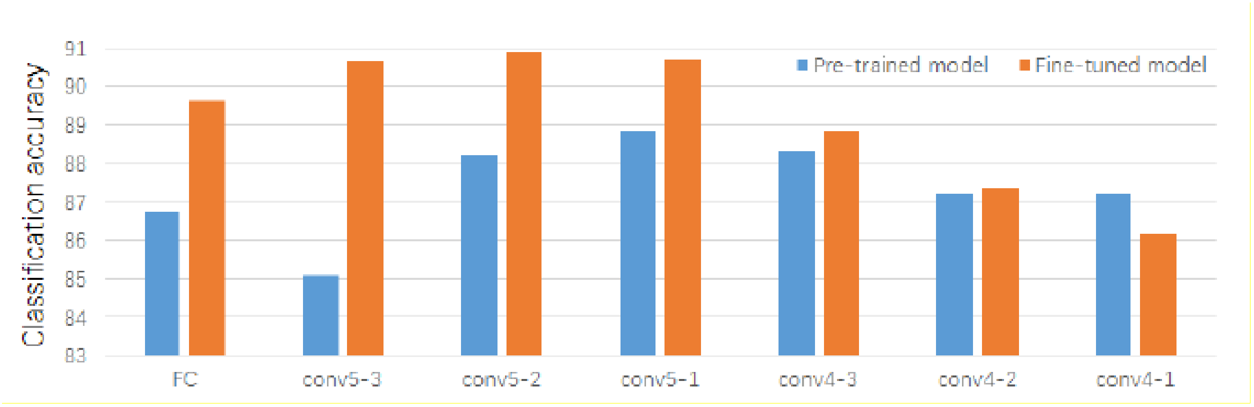


Fig. 32: CNN Accuracy. Shi, Wang, Wang & Xiao (2017)

Like RNNs, CNNs can share weights among all neurons. In image classification problems this in optimal because neurons operate on each feature map during training to learn the structure of each area of each map during convolution (Ciregan, Meier & Schmidhuber, 2012). This local connectivity maximizes efficiency by distributing the learning across the image sequentially rather than linearly like traditional supervised learning functions.

CNNs are composed of three layers: a previously described Convolutional layer, a pooling layer and the final layer creates the final connection of the full network. Pooling layers are often seen in between iterative Convolutional layers and serve to reduce the spatial size of the image region representation; this maximizes the computational process overall (Lee, Gallagher & Tu, 2016). This is done through a process called Max Pooling, which uses filters to reduce the dimensionality of each image region processed by the Convolution process in a process also known as downsampling (Schmidhuber, 2015). The final layer, the fully connected layer, contains all activations to all neurons in the previous layer and completes the network. The CNN process is shown in Fig. 33.

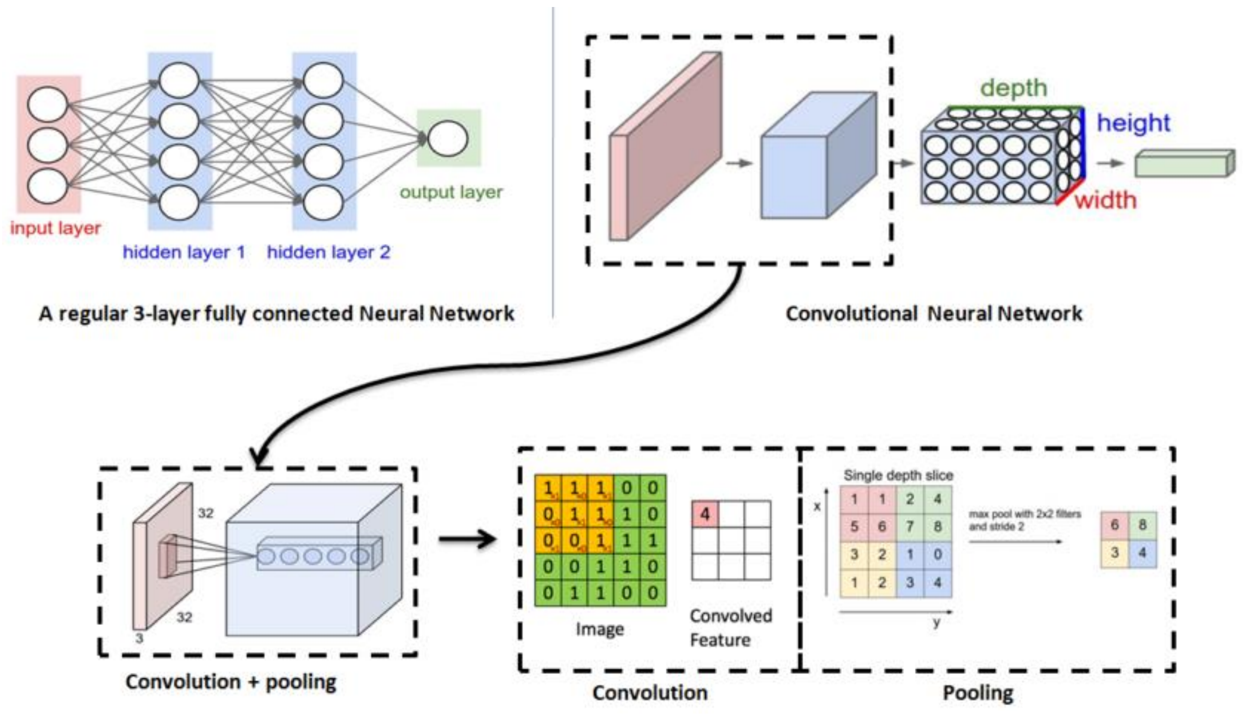


Fig. 33: Convolution Process. (Convolutional Neural Networks, 2018)

Training sequential networks occurs much differently than when using non-sequential neural networks (Oquab et al., 2014). Not only does the entire layer architecture need to be pre-planned (many tools have grid search functions that can help optimize and automate this process), but each model has a specific set of learning processes and options that also need to be configured. After establishing the training and testing sets, the model is defined with a series of one Sequential Class (if an API is being used which is most common) then a series of layers follows. The first layer, which serves as the input layer, takes the number of input dimensions (features), an activation function and a model assessment metric. The activation function in CNNs works just like an NN activation function and calculates the weighted sums of its inputs to determine the relative output. Two of the most common types of activators are Rectifier (reLU) and Sigmoid, but only ReLU (Srivastava et al., 2014) will be used in this work because its sparse activation matches the sparse connectivity in the defined layers in RNNs and CNNs used in this study.

The learning process for sequential models contains two different functions; a third is added to control what metric the model trains or prioritizes during training. The first function the learning process uses is called an optimizer; this function controls what optimization function is being applied to the learning process (Lv, Jiang & Li, 2017). The ‘Adam’ optimizer is often used find a good starting point in the training and learning processes; ‘Adam’ differs from the normal Gradient Descent and has some advantages over other methods as well (Kingma & Ba, 2014). It uses a moving average of the gradient and the squared gradient instead of only adapting the learning on the average mean as in more traditional (and older) methods like traditional Gradient Descent. The next function in the learning process is called the loss function and this defines the magnitude of error of the learning that occurs during training just as is done in ANNs.

During training, there are three parameters that are specified: epoch, batch size and number of iterations. Sequential models, as has been previously discussed, don’t train like ‘regular’ machine learning models; they require many passes through the training data to establish the training baseline and subsequent error metrics going forward (Jaeger, 2002). The number of epochs in a model represents a single forward and a single backward pass through the training observations. Batch size refers to the number of training examples in a single epoch or forward/backward pass, and the size of each batch is proportional to the amount of memory used for training. The larger the batch size is, the more memory that is used, which is one reason smaller batches of one hundred examples are normally used during training (Sutskever, Vinyals & Le, 2014); the number of iterations controls the number of passes made for each batch size.

3.8. Conclusion

Different supervised machine learning methods and deep learning methods used for predicting solar irradiance were outlined in the section. These studies were selected not only because of having been predominantly cited in many GHI prediction studies, but also for their relevance to the work being presented here. The criteria for choosing these studies outside of the two previously mentioned objectives were centered on the implementation of neural networks, ensemble models that utilize linear costs functions as an optimization feature and deep learning methods, of which there are currently very few predominantly cited works. Performance metrics from some of the studies outlined in this section will be compared to the hour-ahead prediction metrics produced from the deep learning models later in chapter 5. The models chosen for comparison in this study are shown Table 9.

Table 9: Comparison Metrics Used in This Study

Studies	**MAPE**	**RMSE**
Ding, Wang & Bi (2011)	14.75%	NA
Amrouche & Le Pivert (2014)	22.50%	51.5 W/m^2
Khatib, Mohamed, Sopian & Mahmoud (2012)	NA	135.5 W/m^2
Chen, Duan, Cai & Liu (2011)	19.45%	NA
Chu et al (2014)	NA	77.5 W/m^2

CHAPTER FOUR

PHASE ONE

Fig. 34 provides a visual understanding of the machine learning workflow as was previously shown in chapter one; each process will be explored further in the first section of this chapter. After collecting data, *preprocessing* is the next part process and involves cleaning and transforming data to prepare it for analysis. *Exploratory Data Analysis* involves obtaining a deeper understanding of all data utilized in the project. Traditional statistical and analytics methods are utilized to gain insight and data is often transformed into a structural format that is more appropriate for machine learning algorithms. This includes scaling, standardizing or normalizing the data not only to ensure that all independent variables are represented by the same scale in the model, but to also decrease the possibility of introducing feature bias, which occurs when a model over prioritizes the relevance of any one given feature due to its greater value of scale, into the model during training causing it to overfit.

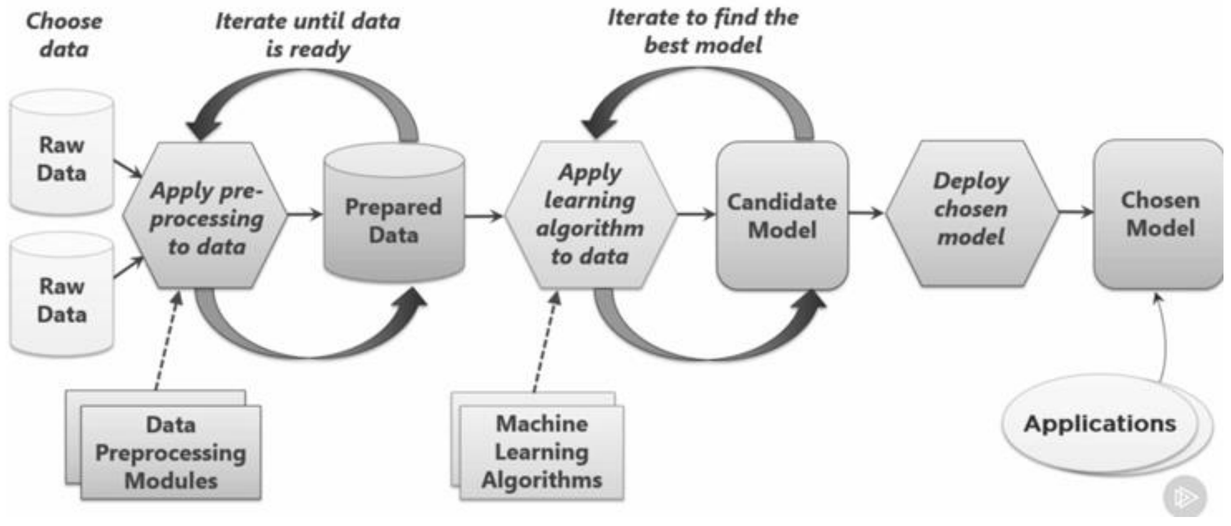

Choose data
Iterate until data is ready
Iterate to find the best model
Raw Data
Raw Data
Apply pre-processing to data
Prepared Data
Apply learning algorithm to data
Candidate Model
Deploy chosen model
Chosen Model
Data Preprocessing Modules
Machine Learning Algorithms
Applications

Fig 34: Machine Learning Process. The Machine Learning Process. (2016)

Feature Selection as was previously described involves not only selecting the specific independent variables that will make up the feature space used for training a model, but also involves making some decisions about how the data will be used for training. *Model Training* or model fitting contains two objectives: iterative model training and model selection. Iterative model training involves repetitively training a model with different machine learning algorithms until an appropriate success metric has been met. Tuning or adjusting the training parameters of each algorithm is done during this iterative process to improve the performance of each model. During this iteration it is sometimes necessary to return to the feature selection process to make modifications and/or improvements to the structure of the data based on the output of the iterative training sessions (Michalski, Carbonell, & Mitchell, 2013). This process continues until an optimal model has been constructed and a predefined assessment like RMSE or Accuracy met. The model is then applied to the testing set and a second set of assessments created. This is to ensure that the trained model generalized optimally to data that it was not trained on. This also ensures that no overfitting occurred during training. If overfitting occurred, the assessment of the predicted values compared to the known values in the test set would be very low (Witten et al., 2016) and model building would need to be repeated after solving the issue. If the testing assessment validates and the trained model generalizes well to values not used for training, the model is then put into production or deployed into a working environment.

4.1. Data Collection and Preprocessing

The data used in this study was collected from the National Solar Radiation Database and represented average observations taken every 30 minutes in the region surrounding Athens, Georgia, USA. The recorded data was delivered as a series of text-based files that contained recorded observations from years 1998-2014. Data during the daylight hours between 8:00 AM and 4:00 PM was used to capture the most optimal times for studying GHI as was done in Yadav, Malik & Chandel (2014). Preprocessing involved imputing any missing values with the related means except in the cloud type variable. This variable contained over 1000 empty observations that represented unknown clouds; these were removed from the data since a suitable replacement was not available. Preprocessing ended after removing any data that was non-usable or in a format that was not acceptable for machine learning models, which represented ~ 2% of the data.

After preprocessing and cleaning the data, exploratory data analysis was performed to gain insight useful for building models. The main goals of this section were centered on visualizing the distribution of each feature, identifying any potential outlying observations and comparing the relevance each feature had to other features and the relevance each feature had to Global Horizontal Irradiance. Correlation was measured between all the variables to study the strength of the relationships and to identify variables that were highly correlated not only to GHI, but also to each other. This is called collinearity (Belsley, 1991) and will be problematic for regression algorithms causing overfit during training. This occurs because regression algorithms often cannot properly ascertain the correct weights to assign to individual independent variables that are highly related to each other. Classification models are not impacted by collinearity, but rather can often be problematic in problems with high dimensionality or containing many features (Friedman, 1997). In classification, correlation is used to identify features that are not relevant to the model; these features can be removed to reduce the number of dimensions in the data and decrease training time.

Pearson’s correlation coefficient was used to measure correlation among all the numeric variables present in the data. Correlation coefficient is measured numerically from -1 to + 1; variables that have correlation coefficients close to -1 are negatively related (Taylor, 1990) and as one numeric variable increases the other numeric variable decreases. When two numeric variables have a high positive correlation, they are positively correlated and when one variable increases the other variable also increases. A matrix of all correlations for the variables in the study is shown in Fig. 35. After investigating the correlation between all variables Temperature was found to be correlated (0.790) with Dew Point as well as GHI (0.520). The only other two variables that were notably correlated were Dew Point and Relative Humidity and were only slightly correlated (0.215).

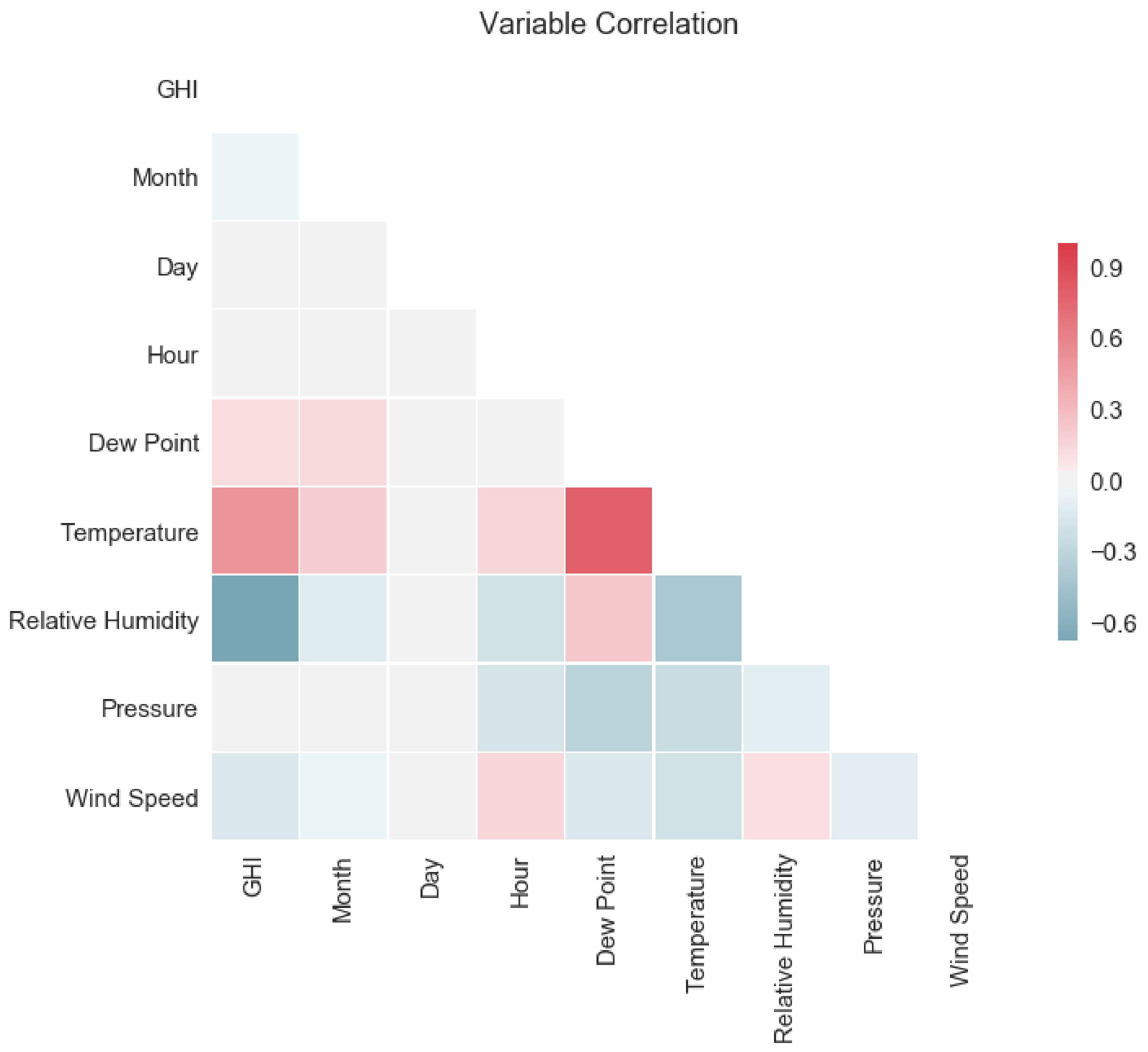


Fig. 35: Variable Correlation

None of these four were removed from the data, although Temperature or Dewpoint might need to be removed from the regression process to prevent overfitting from collinearity; this would be checked later after modelling. No other variables needed to be removed from the feature space as the data was not high dimensional and removing variables because of low correlation may have removed otherwise strong candidates from the machine learning process (Hall, 2000).

Histogram visualizations are used to plot the frequency of occurrence of a given variable; in machine learning it is also useful to understand potential distributions that variables may follow (Rasmussen, 2004). Understanding the distributions of variables can help define the type of algorithm used for training. Histograms do not provide distribution information; this information is derived from density estimation (Silverman, 2018) and this method is often used in machine learning to ensure the correct type of algorithm is being used for training (Robert, 2014). Kernel Density Estimation (KDE) is a type of density estimation method that uses a kernel or type of distribution to specify the shape of the distribution at each point in the data; this allows inferences to be made about the population represented by the variable being studied (Nasrabadi, 2007). KDE is used in the next section of to help visualize any potential nonparametric or parametric distributions that would otherwise not have been shown by histogram binning or partitions alone. Any variables relevant to time were not studied in this phase due to their ordered nature.

Temperature in the region, as is shown in Fig. 36, was skewed to the right of its mean value of 68.08 °F and was widely distributed with a standard deviation of 16.03 °F. These amounts are reflective of the daylight hours covered in the study and do not reflect temperatures throughout an

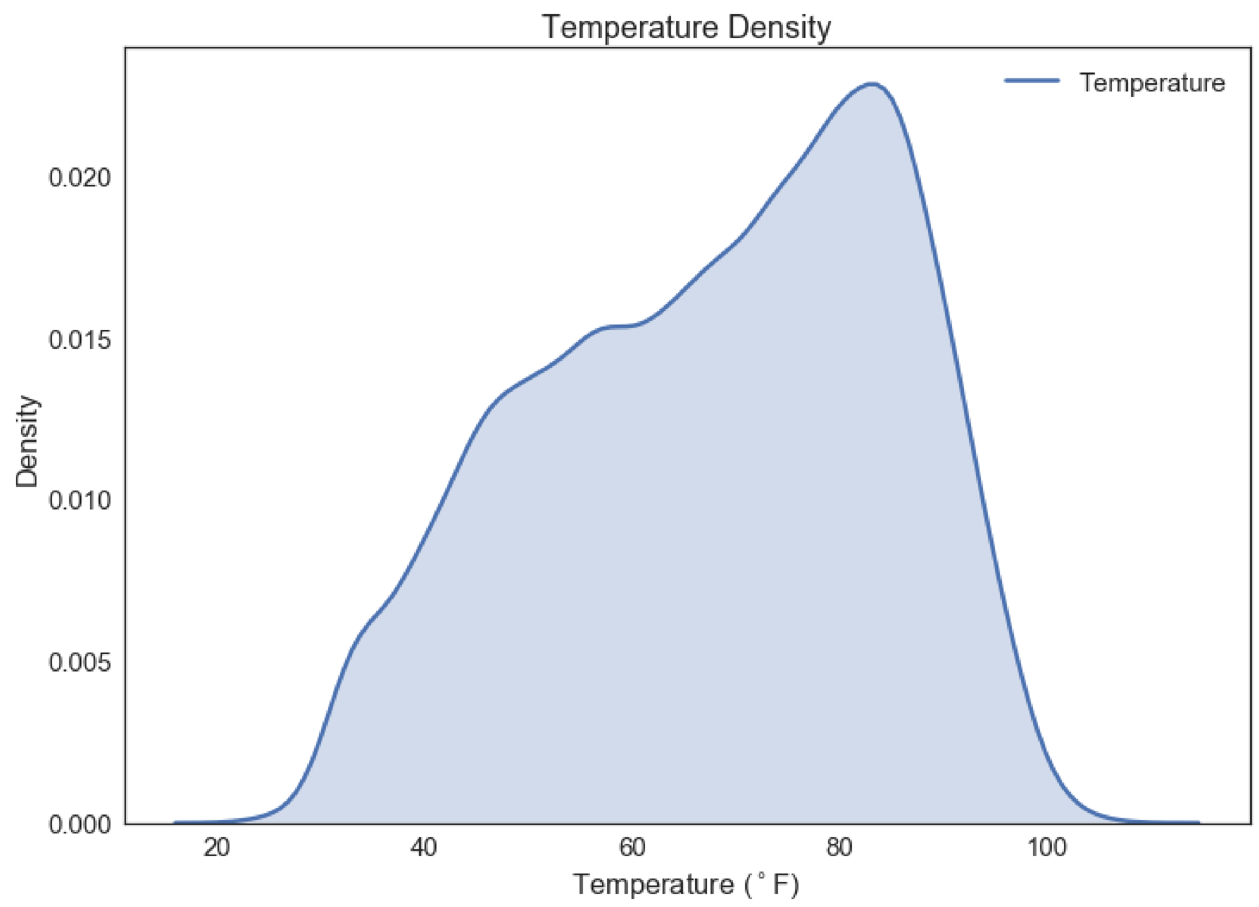


Fig. 36: Temperature Density

entire 24-hour day. Temperature values just above the mean occurred more than values occurring below the mean and most of the observations were in the upper two quartiles.

Barometric Pressure was normally distributed throughout the observations with a mean of 982.07 millibars and standard deviation of 6.04 millibars. Wind Speed was distributed around its mean of 1.33 m/s; Dew Point, like Temperature, was distributed widely to the right of its mean of 51.11 °F with a standard deviation of 14.67. This is relevant to the region where the data was measured as many days are hot and muggy from the high moisture content in the air and the hot days that are prevalent throughout many days of the year.

GHI was the last numeric variable studied in this phase of the analysis. Shown in Fig. 37, GHI had a mean 485.75 of and a standard deviation of 259.47 and a distribution skewed to the left

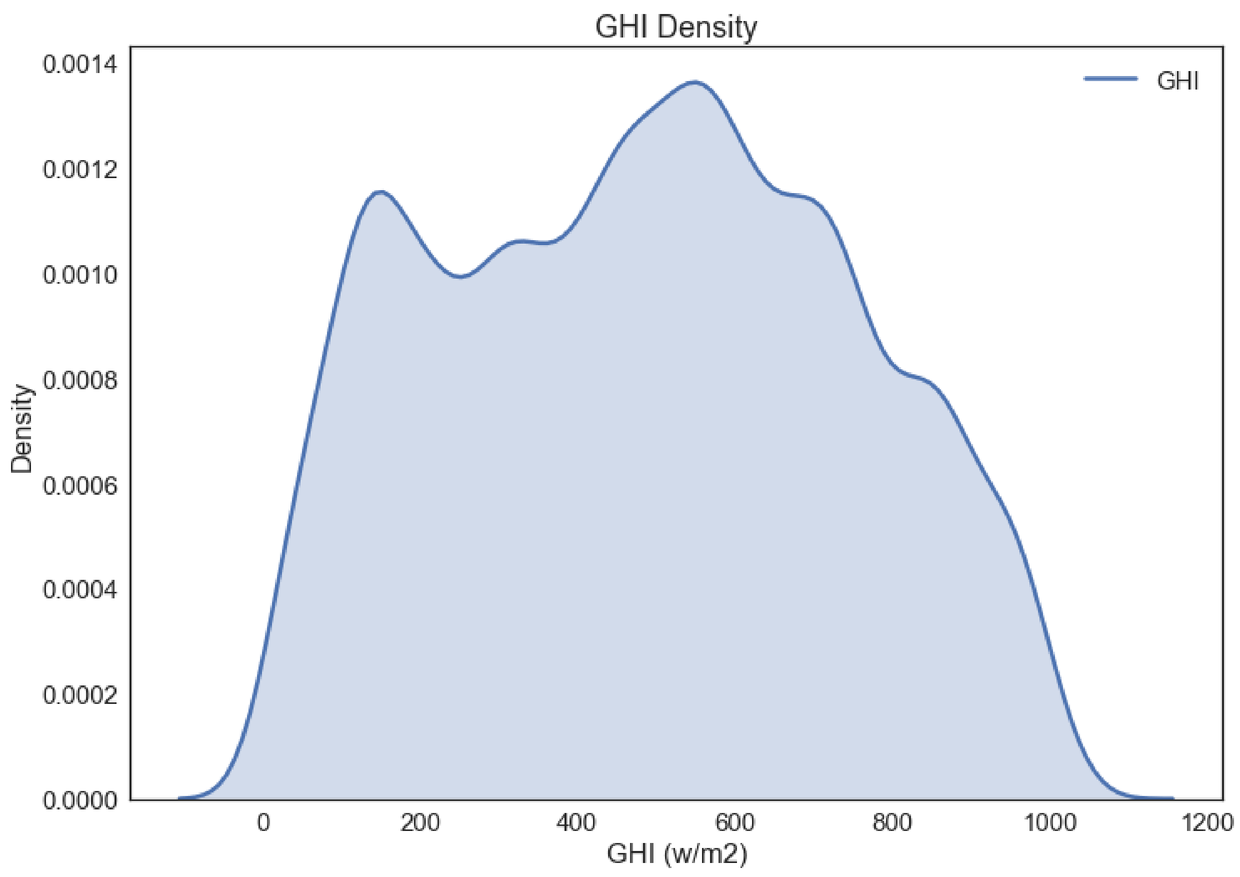


Fig. 37. GHI Density

of the mean. GHI is the dependent variable in all the models and <u>its high variance is one of the reasons classification is also being utilized in this work with GHI amounts being discretized into four different buckets like Jiménez-Pérez & Mora-López (2016) did with cloud classes</u>. These processes will be covered later when the models are described in more detail. The higher amounts of solar irradiance prevalent in the distribution are noteworthy and provided purpose to initiate this work. Cloud Type and Wind Direction are not numeric variables were not included in the density visualizations.

4.2. Exploratory Data Analysis: Pairwise Comparisons

In the next phase of exploratory data analysis, each numeric variable was compared to GHI to ascertain if any of the independent variable distributions were like or intersected with GHI. This would make the independent variable being studied a good candidate for a machine learning model (Rasmussen, 2004). Joint plots were utilized for all the visualizations in this section as these were necessary to show the intersecting densities of two the variables. Darker areas in the visualizations are indicative of intersecting higher densities of both variables; lighter areas in the visualizations are also indicative of intersecting distributions but are indicators of wider distributions that are more widely distrusted across the observations. The relative KDE for each variable is shown in the right and top margins of each plot. In addition to using KDE for this portion of exploratory data analysis, the Pearson correlation coefficient and relevant P-values, which are used to determine possible statistical significance between both variables are discussed here. It is customary (Robert, 2014) to accept P-values of less than 0.05 as an indicator of the relationship between the variable being compared to GHI and GHI as being statistically significant and not a random occurrence. The first comparison was made between ‘Month’ and ‘GHI’. As shown in Fig. 38, there was an overlap in density between the winter months and lower amounts of solar irradiance.

Dew Point was the next independent variable that was compared to the distribution of GHI. As was discussed earlier, this variable was skewed to the right and related to Temperature; a similar assessment was found when comparing it to GHI. These two variables showed a dense intersection throughout almost the entire distribution of GHI. GHI was found to be higher when the Dew Point was also higher, which is a positive relationship and quantified by the .11 correlation coefficient; the relationship was also validated by low P-value (0.02) score between the two variables. The next-to-last variable comparison included in this phase of exploratory data analysis was a comparison between GHI and Hour as is shown in Fig. 40. This comparison resembled the

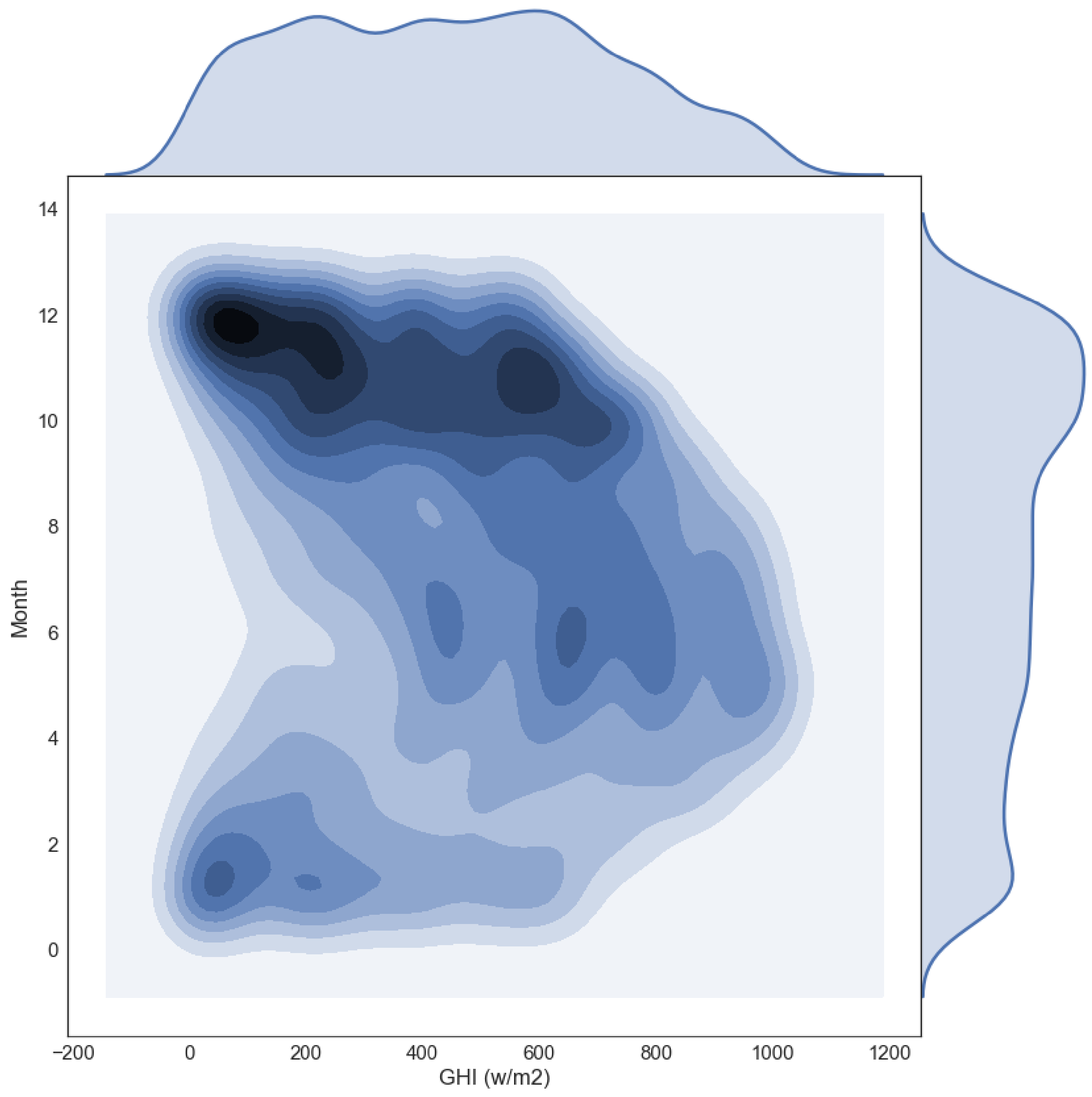


Fig. 38: GHI/Month Density

comparison between GHI and Month and is prevalent in both visualizations as time is related to all three variables. In this comparison, the intersecting densities are lower at the beginning and end of each time that was being studied.

The last pairwise comparison made was between GHI and Hour, which is shown in Fig 39. The relationship between these two variables shows an intersection between their distributions on the lower ends of both variables with intrahour variance also being shown. This can also be seen in the slight negative correlation between the two variables, but low P-value (0.015) measured between these two variables validates the findings the previously covered studies where Wind Speed was relevant to the prediction of GHI. Although high wind speeds are not prevalent in the region being studied, this variable will likely remain in the feature space as most of the previous work also used it as a feature for modeling.

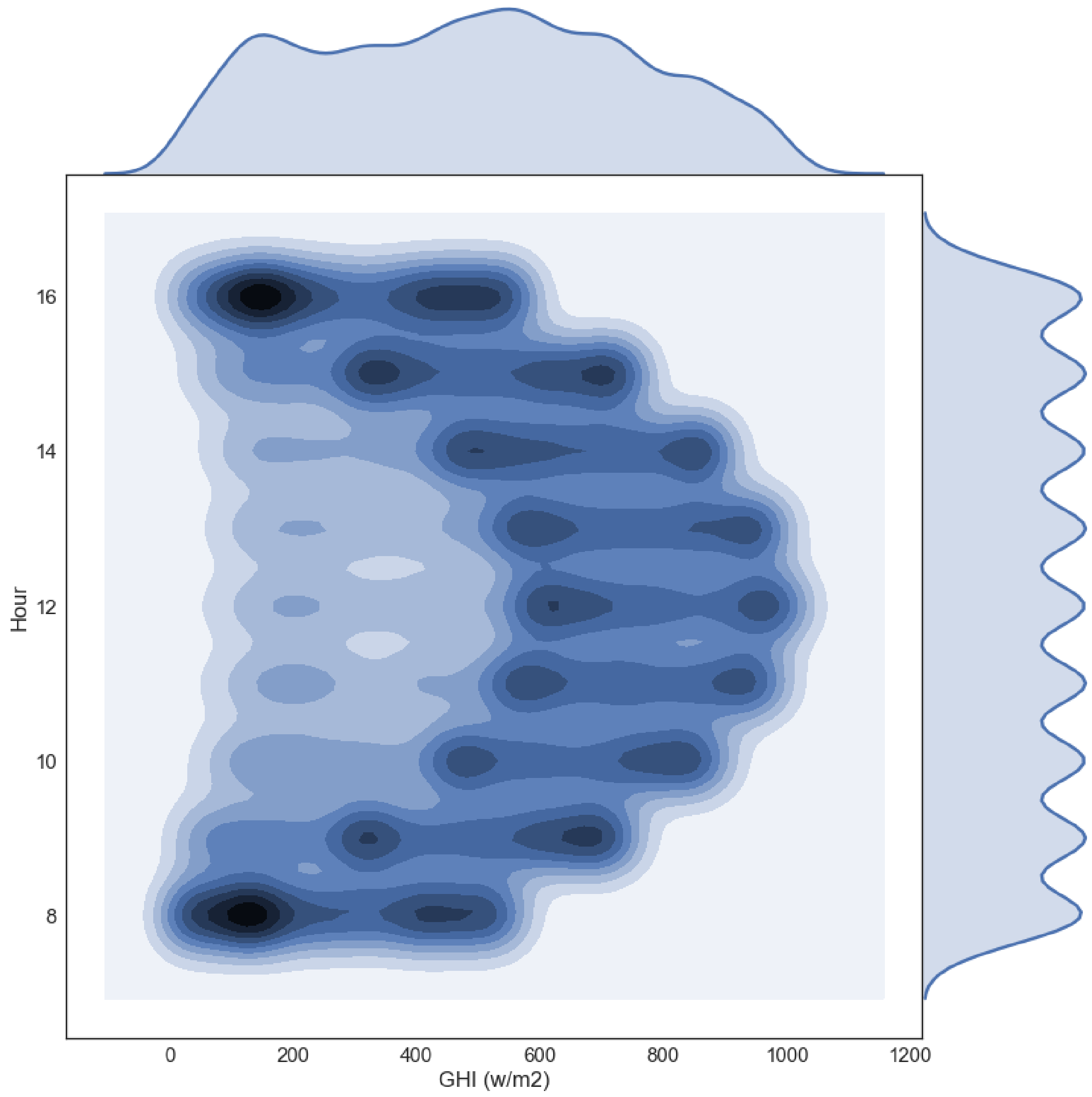


Fig. 39: GHI/Hour Density

While this might not be very relevant in the southern states, it will help any models derived from this work generalize well when/if applied in different. In the current case, values that fall outside of the current distribution and used to validate the predictive models would likely cause a slightly higher error rate. This could be corrected with more data that better explains variance in the outcomes.

While expert level domain knowledge is often not a needed (Witten, Hall & Pal, 2016) when building machine learning models, this analysis expanded the overall understanding of the impact individual numerical weather variables have on the availability of GHI. The findings from this phase of the process will be incorporated into the next phase where the specific feature space for modeling will be created. After studying the correlation of all the independent variables Relative Humidity was found to be related to Temperature and may be removed from the next phase of the process and not be used as a feature in any future modeling. Dew Point and Temperature are also correlated and their distributions intersected similarly with GHI, which might be an indication of collinearity; this relationship will have to be monitored during model training as a potential influencer of overfitting.

4.3. Feature Selection

The next phase of the machine learning process involves making a final selection of the independent variables that will make up the features the models will use for training. Feature engineering methods were not utilized here because the data was not highly dimensional and the many of the independent variables lacked the correlation needed for feature engineering methods to be beneficial. Inversely, independent variables can also be engineered solely to reduce the dimensionality of the feature space so feature selection methods were used. As previously described, there are three methods of feature selection: filtering methods, embedded methods, and wrapper methods. Filtering methods were used previously used doing exploratory data analysis to identify highly correlated features. Recursive Feature Elimination was used on the independent variables to ascertain if it otherwise may have been beneficial to remove more independent variables from the proposed feature space than those few that were identified during exploratory data analysis. This was done to ensure nothing was missed during the previous analysis. This is often done (Das, 2001) in cases of regression and classification for this same goal. While RFE did identify four independent variables that could be removed from the model, these were not identified as being problematic or likely to cause overfit, but were identified because of their low correlation to the dependent variable. As has already been discovered all the independent variable distributions intersected in some type of way and most have very low P-values when measured with GHI.

In the last part of the feature selection process, the independent categorical variable Cloud Type had to be changed from an integer to an object before it could be used in any multiple regression problems. This was because the numerical values were represented by this variable represented each type of cloud that was identified by the current solar PV model, which are outlined previously in this work. Regression problems can use categorical variables if they only contain two different classes; classes represent the number of levels or categories that are contained any categorical variable. Cloud Type contained more than two classes so it had to be converted to a series of additional variables that were each represented by a numerical Boolean state of either ‘0’ for not present or ‘1’ for present. The additional features that represent the single categorical variable were then be added to the final feature space as shown in Table 10.

Table 10: Final Feature Space

Independent Variables	Data Type	Dependent Variable
Month	Integer	
Day	Integer	
Hour	Integer	
Cloud Types (1-9 Boolean)	Integer	
Dew Point	Integer	GHI
Temperature	Integer	
Pressure	Integer	
Wind Direction	Integer	
Wind Speed	Integer	

4.4. Modelling

To establish a baseline to compare the deep learning models against, two processes were designed to create models using the previously identified feature space for training. These processes, one being a multiple regression process and the second being a classification process, used the same machine learning algorithms, the same cross validation methods and the same data for training. The same requirements will also be relevant to the deep learning models. The system of modeling is necessary to support the hypothesis of deep learning improving current GHI models.

4.4.1. Multiple Regression: Supervised Learning

Five supervised machine learning algorithms were chosen to be assessed in the creation of the multiple regression baseline models. The algorithms were chosen based on their ability to be tuned with various hyperparameters and their resiliency to overfitting, except in the case of decision trees. Decision trees were included in the baseline study as they are very adaptable to both regression and classification and were used in previous studies previously discussed. Two ensemble algorithms, Gradient Boosting and Ada Boost, were also chosen to be used. All algorithms and relevant hyperparameters are shown in Table 11. K-fold cross validation was used to stratify the data to remove any ordering and to create a testing set for model assessment.

Table 11: Regression Machine Learning Algorithms and Hyperparameters

Algorithm	**Type**	**Hyperparameters**
Gradient Boosting (GBR)	Ensemble	Learning rate = 0.1
K-Nearest Neighbor (KNN)	Neighbor - Distance	K = 2
Decision Tree (CART)	Criterion/Splits	Criterion = mse, Splitter=Best
Multi-Layer Perceptron (MLP)	Neural Network	Activation = relu, Max Iter = 200
AdaBoost	Ensemble	Estimators = 50

To create a comparison between the baseline regression models, baseline classification models and deep learning methods default hyper-parameters were used during training. Machine learning algorithms, especially ensembles and neural networks, provide numerous tunable parameters and can be tuned specifically to the type of data they are being used on, but this was not the focus of this work. Model training was done using Python and a machine learning library called Sci-Kit Learn. Learning curves were created for each model; these show the progress of the model during training. The Learning curves assessed both the training score, which was the R^2 produced by the models, and a cross-validation score, which is a metric produced by Sci-Kit learn to compare algorithms against one another when used on the same data (Pedregosa et al., 2011). Higher cross-validations scores (Table 12) mean the algorithm being assessed is more sensitive to the same data a lower scoring algorithm was used on. All models were validated using a sperate validation set taken prior from the training and containing 500 random observations.

Table 12: Cross Validation Scores (Regression Models)

Algorithm	**Cross Validation Score**
Gradient Boosting (GBR)	0.92
K-Nearest Neighbor (KNN)	0.87
Decision Tree (CART)	0.87
Multi-Layer Perceptron (MLP)	0.81
AdaBoost	0.74

RMSE was used to assess the validated models and R and RMSE were recorded to monitor for overfit and as assess the performance of the models. The top three models are presented here beginning with Gradient Boosting. Gradient Boosting (GBR) scored higher than other methods and its performance was reflective of it being a good method to use o this data. In addition, the learning curves (Fig. 40) and cross validation curves converged during training, which is an indication of the persistence of the model and it not needing any further data fir training (Dataquest. (2018). GBR understood more the variation in the data than all the other algorithms with an R^2 value of 0.901. The same model also produced a lower error, when comparison to the other models, of 62.68 W/m^2. The Classification and Regression Tree also performed well during training as shown in Fig. 41 with a convergence between the learning curve and the cross validations cross-validation score.

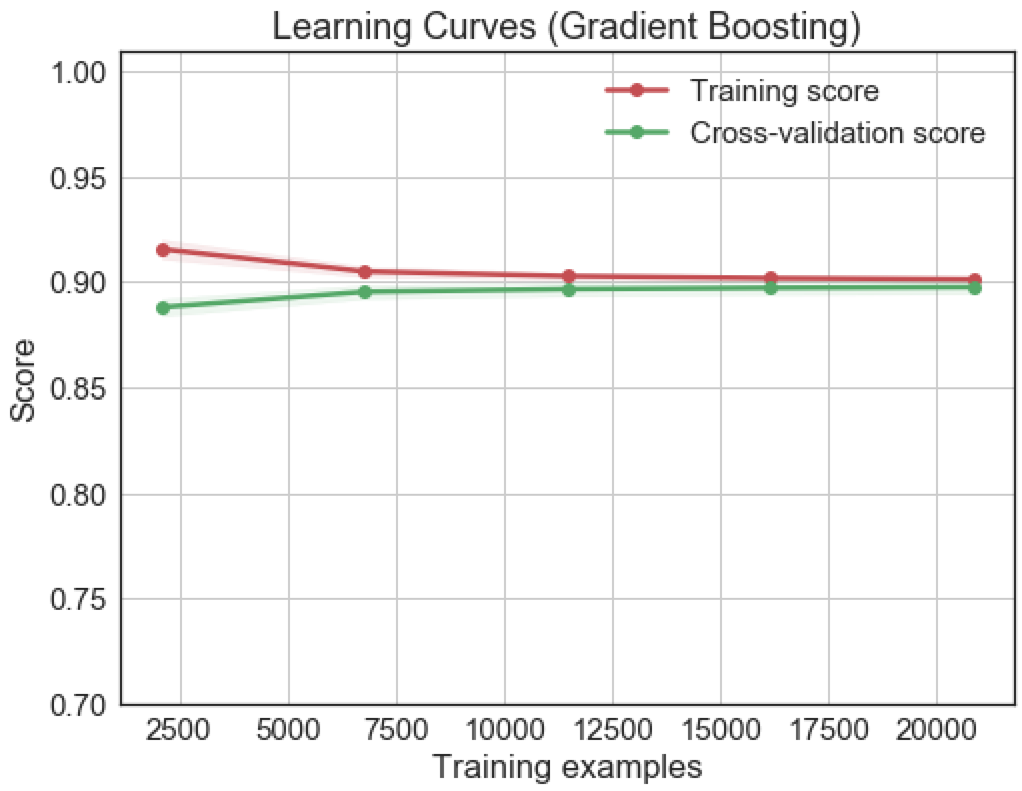


Fig. 40: Learning Curves - Gradient Boosting

Decision Trees and Gradient Boosting ensembles share some of the same regression functionality (Elith, Leathwick & Hastie, 2008); this is likely why both were so similar during training. From the produced metrics the Decision Tree would have likely outperformed GBR with if more data were available.

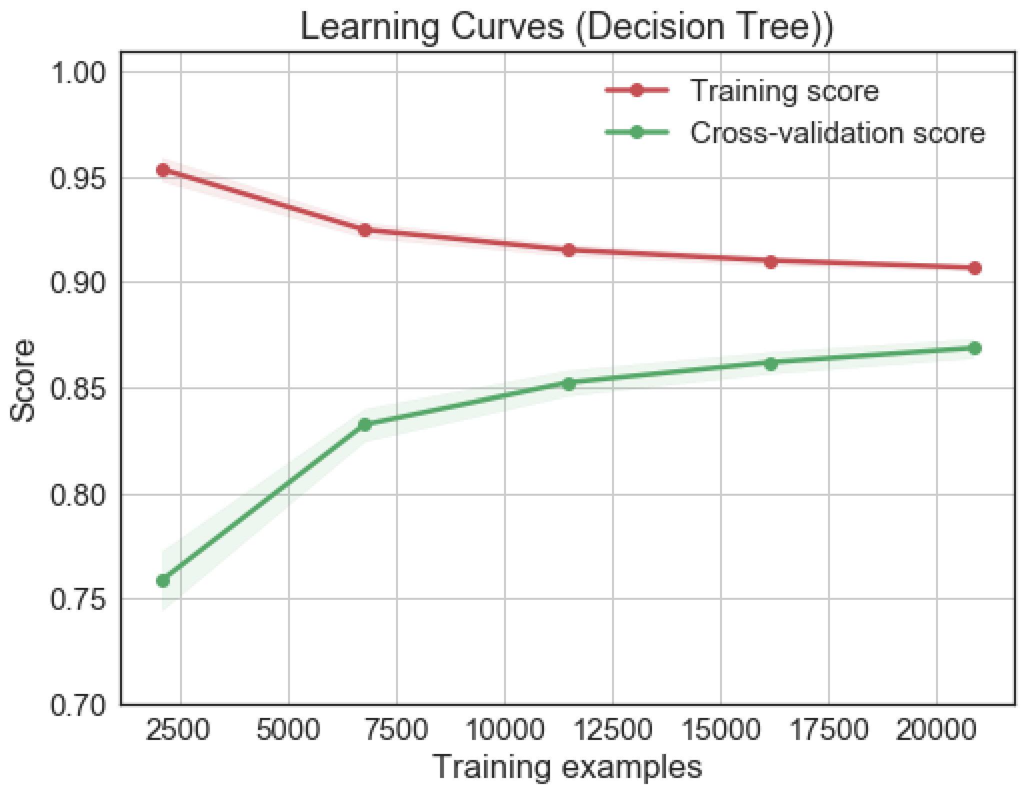


Fig. 41: Learning Curves – Decision Tree

KNN was the final model outlined in this work; it scored 0.87 for a cross-validation score, but this model produced a high RMSE of 109.33 W/m^2 on the validation data. This is indicative of KNN not being able to generalize well to the outcome in the validation data. KNN has shown (Rajagopalan & Lall, 1999) to be sensitive to numerical weather data so this is likely due to the validation set not having the same variance or less than the training data.

While 0.901 is a very acceptable metric for regression models, the GBR ensemble that produced it did also produce a larger error rate than the Decision Tree model as shown in Table 13. This might be indicative of the size of the data or from variance in any one of the given features; GBR will not be used in this work. For the purposes of a baseline comparison used in the remainder of this section, the Decision Tree will be used.

Table 13: RMSE values for Baseline Regression Models

Algorithm	**RMSE**
Gradient Boosting (GBR)	44.4 W/m^2
Decision Tree (CART)	39.06 W/m^2
Other Models in the Study	
K-Nearest Neighbor (KNN)	146.99 W/m^2
Multi-Layer Perceptron (MLP)	99.51 W/m^2
AdaBoost	95.01 W/m^2

4.4.2. Classification: Supervised Learning

The next step in the process involved building classification models to establish a comparison set of models. The classification process used the same algorithms that were used in the multiple regression baseline study and the same cross-validation process, parameters and model building processes. The main differences that were how one independent variable was transformed and how the dependent variable was partitioned. As was previously mentioned in the multiple regression process ‘Cloud Type’ had to be converted to a series of binary independent variables to be utilized for multiple regression. Classification models can utilize numeric data that represents categories, so this independent variable was converted back to such. For the dependent variable, GHI, it was a continuous numeric variable and classification problems only work on dependent variables that are nominal or categorical. To convert GHI from a continuous numeric variable to a categorical variable, a process known as discretization (Kerber, 1992) was used.

Discretization involves equally dividing a continuous numeric variable into partitions or what is called bins. Each of these bins is then given an appropriate label and the variable is converted from a continuous numeric variable to a categorical variable with classes. The distributions of GHI in this data could be equally partitioned into four separate bins and still represented a positive and useful distribution that could be utilized for model building. GHI was binned into four groups that were labeled as: Low, Good, Better and Best. Each of these bins represented the amount of GHI that was a relative to its availability in each predicted class. 'Low' GHI could be considered not relevant enough to be productively gathered by most solar PV arrays.

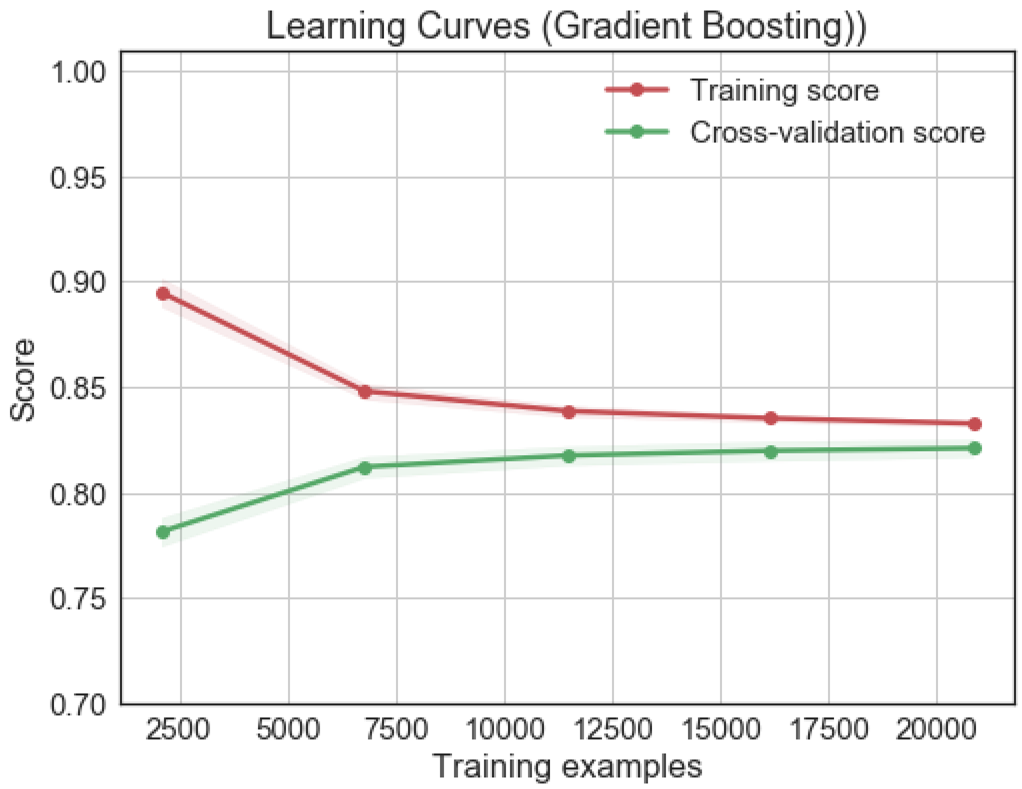


Fig. 42: Learning Curves - Gradient Boosting

'Better' GHI could be useful for everyday use in most photovoltaic arrays. 'Better' and 'Best' GHI amounts are the top two in the predictive classes and represent the most optimal states (Ram, Babu & Rajasekar, 2017) that could be utilized by solar photovoltaic installations. Higher amounts of discretization could easily add additional categories or classes to GHI thereby creating more precision, if needed. All five classification models were trained and again, the GBR ensemble scored an Accuracy of 84%. The learning curves for this model are shown in Fig. 42 with both curves converging during training. In the classification modelling, the decision tree (Fig. 43) also performed well with an Accuracy of 84%, which is further evidence of the influence of the similarities it shares with GBR. KNN, MLP and the other ensemble all performed lower than these two models in the classification process as is shown in Table 14.

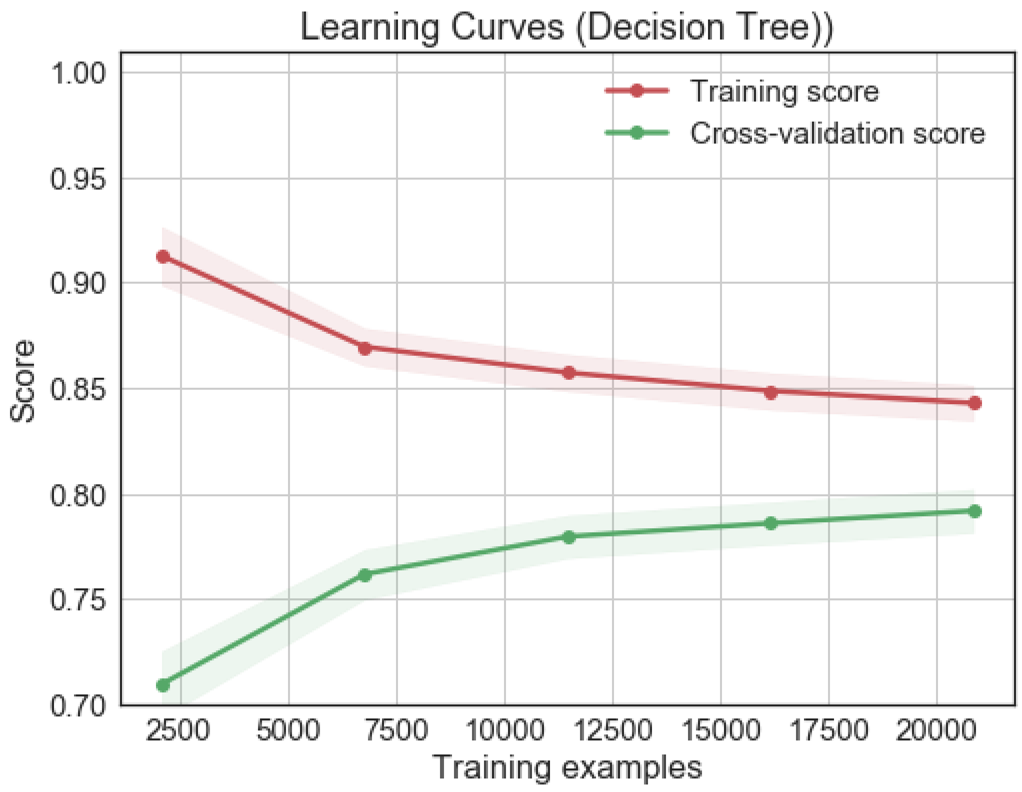


Fig. 43. Learning Curves – Decision Tree

The baseline results of supervised learning methods show that ensembles represent a possible avenue in the use of GHI prediction as the Gradient Boosting algorithm performed highest of all supervised learning methods tested. While the Gradient Boosting algorithm did very well on the Coefficient of Determination it also presented higher error, which was unacceptable. As previously explained the error metric (RMSE) used to assess the regression models is in the same unit as the dependent variable. This would mean the ensemble method that performed so well during baseline regression models could miss generalizing by as high as 45 W/M^2. Deploying the model built from the Gradient Boosting regression algorithm would present the risk of having a 10% error in its baseline predictions when related to the total W/M^2 in a single individual observation.

Table 14 – Accuracy values for Classification Models

Algorithm	**Accuracy**
Gradient Boosting (GBR)	83%
Decision Tree (CART)	84%
Other Models in the Study	
K-Nearest Neighbor (KNN)	80%
Multi-Layer Perceptron (MLP)	76%
AdaBoost	62%

4.4.3. Regression Modelling: Deep Learning

The regression portion of modelling using Deep Learning used the same feature space as was used in the supervised learning process and used a Recurrent Neural Network and the same validation method that was used for the supervised learning models. The network was designed to use five layers during training, including the final fully connected layer as shown in Fig 44. A custom function was built so RMSE could be used as the loss function and the 'Adam' optimizer was used as an optimization function. The model prioritized accuracy during its training so the loss (RMSE) would be minimized during; a batch size 50 of was processed by 400 epochs. The network design and training parameters produced a model with an RMSE of 25.23 W/m^2 and an R^2 of 0.926, which is higher than what the ensemble model scored in the supervised learning regression process, but lower in error.

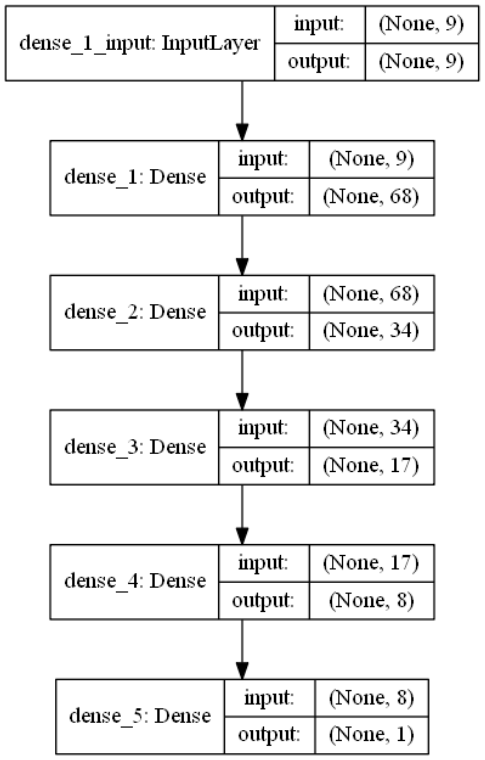


Fig 44. Recurrent Neural Network Design: Regression

When the error in regression models decreases, R^2 values also increase unless models are underfiftting. The same learning curves used for the supervised learning methods are not available for deep learning models. However, the training history can be plotted to show the RMSE gained for each epoch used for training. A summary of the RMSE during training, per epoch, is shown in Fig. 45.

4.4.4. Classification Modelling: Deep Learning

The classification portion of modeling used all the same features that were used in regression and followed the process that was used during the supervised learning classification process using a discretized dependent variable that was binned into four classes.

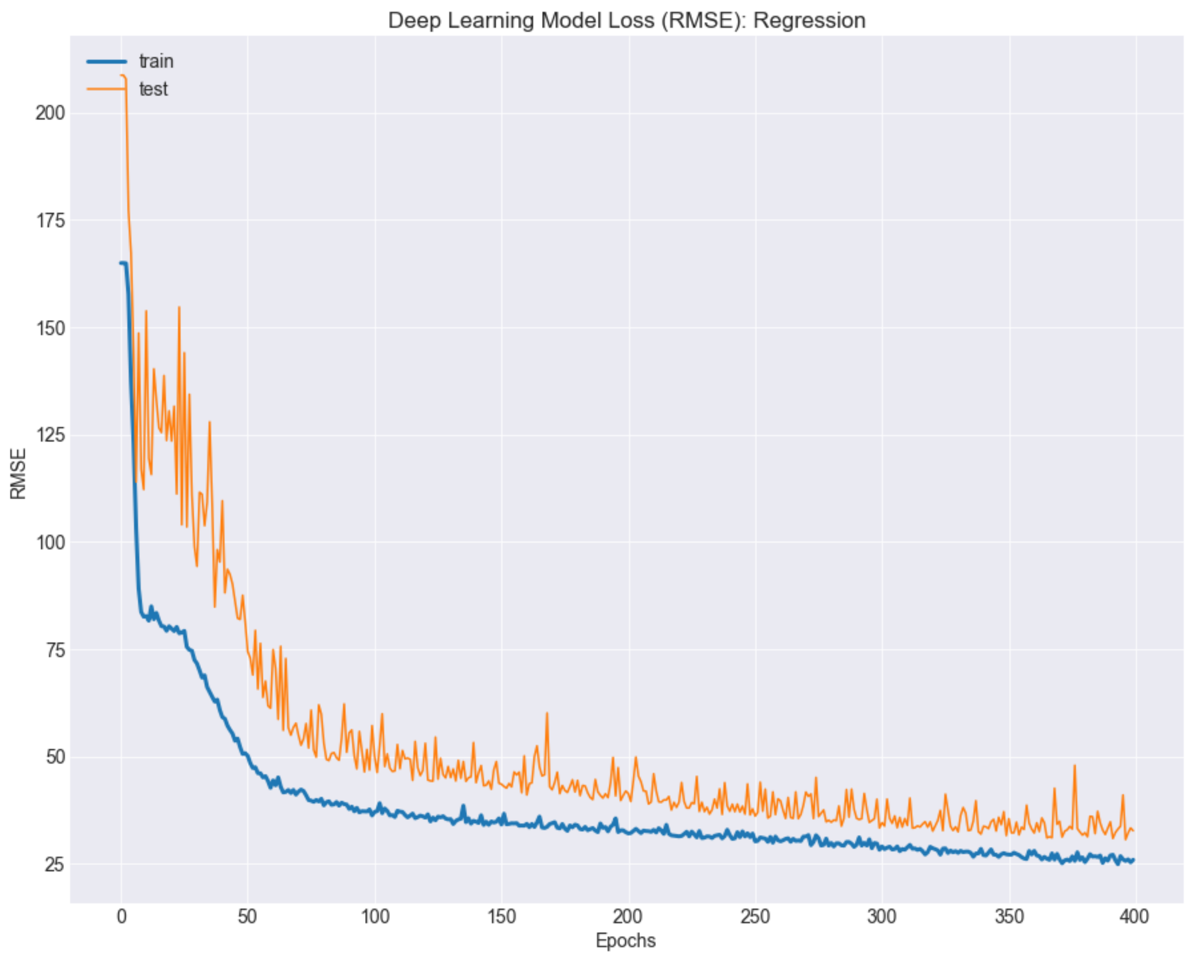


Fig 45: Recurrent Neural Network: RMSE History

The network design for the classification training was different than what was used in regression because of this discretization. The classification model used four layers, including the final connected layer and used on reLU activation to train batch sizes of 10 through 400 epochs of training; the model design is shown in Fig. 46. Model training was optimized using the 'Adam' optimizer just as was done in the regression processes. A function was built so the Mean Prediction value would control how model minimized error during training. Accuracy was used as the output metric. The classification model optimized the mean predictions well enough to produce an Accuracy of 94.35% on the training set and just under 90% on the testing set. Both values

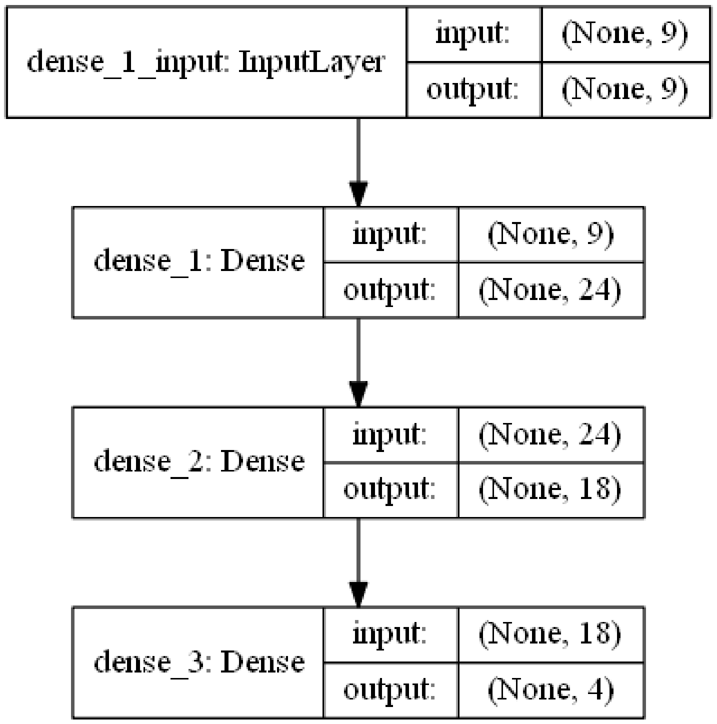


Fig 46: Recurrent Neural Network Design: Classification

(including those found in the regression method) will likely increase with more data being added to the model as the learning histories were trending upward at the end of training; the results of the classification training are show in Fig. 47. The model began a steep training climb and training optimized after 300 epochs of training. The distance between the training line in the testing line is a very strong indicator that the model does not overfit, as the two lines continue separation throughout the training cycles and throughout the epochs. KNN, MLP and AdaBoost produced Accuracy metrics of (80%, 76% and 62%).

4.4. Conclusion

The Recurrent Neural Network trained during the deep learning process retained a very similar Coefficient of Determination as the supervised learning model, but a lower RMSE value as can be seen in the comparisons and Tab. The lower error obtained by this model is reflective of the type of model that a RNN can produce using the optimization function that is applied during

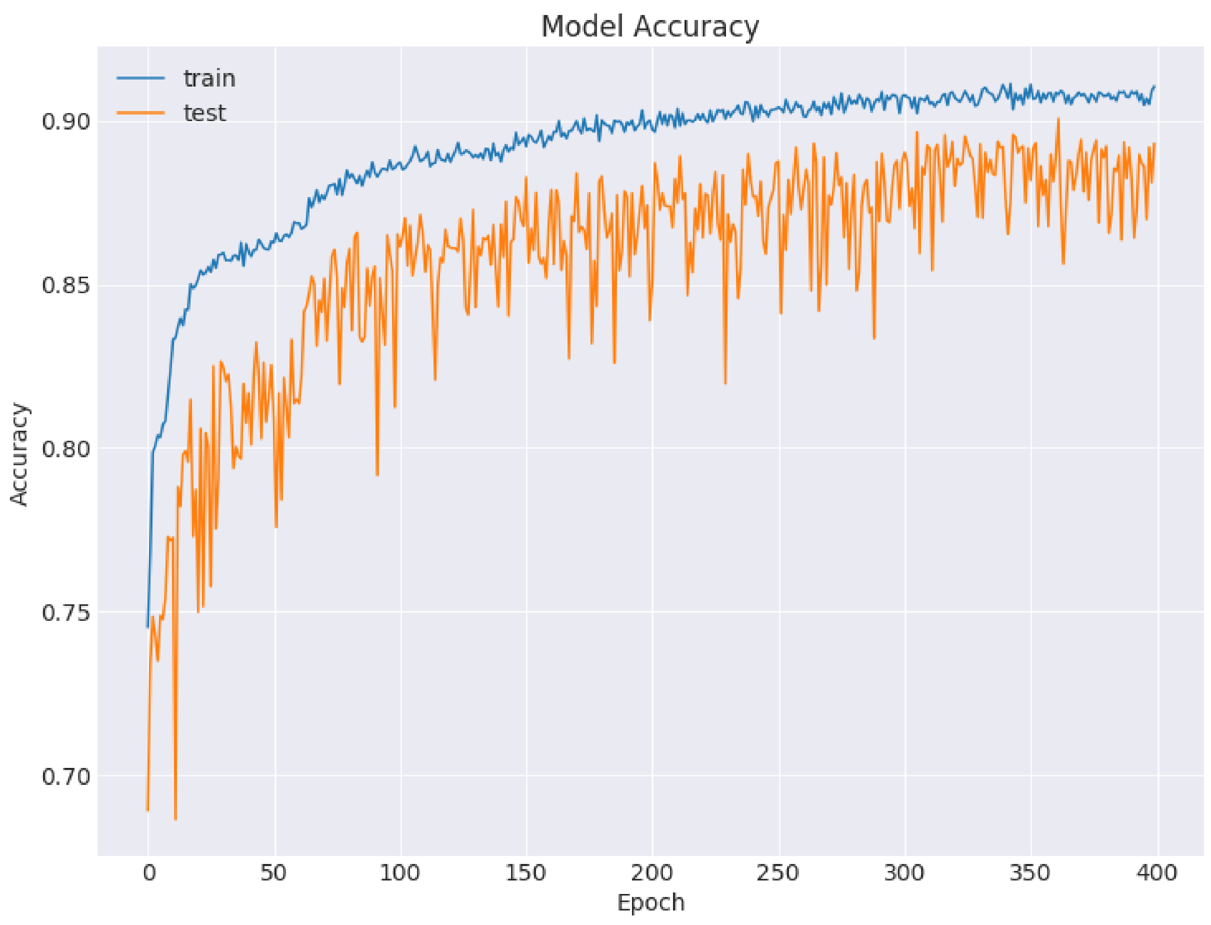


Fig 47. Recurrent Neural Network: Accuracy History

training and from the sequential nature of the layering continuously reducing the error lower during training. This deep learning regression method performed better than all five supervised learning regression methods and three previously described studies as outlined in Table 15; RMSE and R^2 values were both exceeded when used for hour-ahead predictions made between 8:00 AM and 4:00 PM over one week at a single location. This validates that deep learning methods, specifically Recurrent Neural Networks, can outperform traditional supervised learning methods in multiple regression problems used in GHI prediction.

Table 15: Hour Ahead Averages - Comparison

Model/Study	**RMSE**	**R^2**
Chen et. al. (2015)	77.51 W/m^2	NA
Amrouche & Le Pivert	51.52 W/m^2	NA
GBR	44.411 W/m^2	0.92
CART	39.062 W/m^2	0.871
RNN	25.237 W/m^2	0.92

The performance of deep learning in the classification methods was also higher than the five methods representing supervised learning. The Recurrent Neural Network scored over 10% higher accuracy (Table 16) than ensemble methods and decision trees used in supervised learning and exceeded Accuracy in previously outlined study (Alonso-Montesinos et. al., 2016) where similar supervised learning classification methods were utilized.

Table 16: Classification Averages

Classification Model/Study	**Accuracy**
Alonso-Montesinos (2016)	94.1%
GBR	83%
CART	84%
RNN	94.35%

The only disadvantage found for using deep learning in this type of prediction are modeling was centered on the technology itself. Numerous GPU faults occurred during training and the system had to be reset for model training to begin again. The training process is much more stable in cases of supervised machine learning where larger models can train for hours or days without interruption. This could have been a limitation of the system that was being used for this study. In either case, the interruptions were minimal. In future studies of this sort it might be more appropriate to use a cloud-based system that might not interrupt in the process. In addition, the technology behind GPU modeling is becoming more advanced and more optimal as it gets closer to mass adoption in the academic and business settings; all of this is may improve the usability.

CHAPTER FIVE

SKY TYPES

The next section of this research outlines whether a substitution can be made for the feature serving as Cloud Type in current GHI prediction models. As has previously been discussed, this feature is derived from satellite imagery using a complex set of algorithms. These can not only identify the type of cloud, but can also dissect clouds and ascertain the chemical makeup, physical properties, height and distance, none of which can currently be derived from ground observations. This limitation makes finding a substitute variable difficult, but studies previously covered earlier in this text have tried with moderate success. These used specific equipment for collecting whole sky images from ground observation points by capturing a ‘fisheye’ view of the entire sky throughout a day, with the most successful occurring in mostly in clear conditions. To create a more generalized model that can be utilized anywhere, sky-based images need to represent all sky conditions.

Finding a substitute feature will also make prediction less expensive since the use of satellite data would then be optional. In addition to being less expensive, finding a substitute variable might be more optimal. This is because one of the categories used to identify cloud types is used when there is not enough data to classify clouds. This creates an ‘unknown’ category and creates in the observations in the cloud type feature in the data. As was described earlier, this was relevant in the data used for this study and had to be solved during preprocessing. If a substitute variable can be found that has no unknown label category, this gap can be filled and possibly improve the GHI prediction models altogether by providing a more cohesive data set that is more representative of the variables captured and any relative relationships that might be present among them. In addition, without using regional satellite data and focusing on local observations, GHI prediction at an individual location can occur. When examined on a larger scale and across a distributed network, this type of prediction might be very accurate at individual locations because these locations can use predictions from other locations that are nearby where predictions are also being made. This concept goes beyond the scope of this work, but it is a concept that can be explored further if a substitute variable can be found and validated to be useful. For example: a regional, real-time weather forecast is seemingly ever-changing, but the accuracy and response time is increasing with nowcasting methods that utilize this same process (Xingjian et al., 2015). A similar approach can be implemented locally with GHI prediction if a model can be built without the use of cloud types, but with a new feature that is specific to exact locations and could be captured at any time.

5.1. Sky Conditions "Sky Types"

The closest thing relative to the clouds is the sky itself and, while the exact composition of clouds or their height or their distance cannot be ascertained from ground-based observations alone, there is another element that <u>might</u> be relevant to GHI prediction. One variable that can be satisfied from ground observations is the percentage of the sky that is covered by the clouds. The National Weather Service (2018) defines this amount as sky conditions and these are recorded for every weather observation that is reported by the National Weather Service. Their definition is uses an equal division of octants (eights) of the sky and a percentage of the number of octants covered at any given time by clouds in a single observation (Cazorla, Olmo & Alados-Arboledas, 2008). In addition, sky conditions are also captured from space-based imagers, assessed from surface-based instrumentation or collected by trained human observers (Nrel, 2015). All three of these methods contain issues that a new method might solve. Satellite images that are used in this method are often obscured by higher clouds and only capture one moment in time. Lower-level clouds are often closer to the surface of the Earth and are very difficult to obtain from satellite images or are often missed altogether (Arking & Childs, 1985). Lastly, these observations are regional and do not include individual locations other than from the central offices where they are collected. The specific calculation for each sky condition is shown in Table 17, including a sky condition describing a nighttime condition named ‘fair’. For the purposes of this study, this condition was not utilized as GHI is not optimally predicted at night as it is during the day. These sky conditions will be referred to as ‘sky types’ in the remainder of this work and will be the name of the possible substitute variable that will be tested as a replacement for cloud types. This introduces a new question that will be addressed in this section: If the NWS already reports sky conditions and this information is available why not just use the labeled categories that were used during training for building the new process? While this method is viable, it isn’t applicable for local areas. In addition, these types of observations have been recorded for over 50 years. This creates a viable historical data set that can be used to train a deep learning model to identify sky conditions, not from any of the previously mentioned methods, but rather from images taken at the same time. All of this depends on its relevance to GHI.

If it can be validated that the new variable does impact GHI prediction, gathering and classifying local images matching the specific cloud type class that is related to the NWS classification would make a new process scalable and specific to the location the image is collected from. For example: If local images could be paired with the NWS labelled classes provided at the same observation time, a correlation could be made between the NWS text-based label and what the sky really looks like when given that label. The only other data that would be needed to make a local prediction would be the weather data that relevant the same individual location and at the same time. If this process were viable, a user could point their mobile devices to the sky, capture an image with the camera and know the GHI for their exact location at that moment. Expand this process into a networked and distributed system concept and time relevant predictions can be 'pushed' downstream or upstream of the user's location; predictions representing irradiance amounts in the future can then be obtained. This will be discussed in the 'future work' section as using sky types is the relevance of the remainder of this part of the work. Designs and solutions for both portions of the process, image collection and classification, will be presented as well as implemented along with the results being validated.

Table 17: Sky Conditions and Cloud Coverage

Sky Condition	**Cloud Coverage**
Clear / Sunny	0/8
Mostly Clear / Mostly Sunny	1/8 to 2/8
Partly Cloudy / Partly Sunny	3/8 to 4/8
Cloudy	8/8
Fair (mainly for night)	Less than 4/10 opaque clouds, no precipitation, no extremes of visibility/temperature/wind

5.2. Data Collection

Before building a new process, it was only pertinent to ensure that this type of feature was relevant in the prediction of GHI. Just as was done in the previous part of this work, regression and classification models were constructed, but this time only deep learning methods were used. To bridge the gap between numerical weather data containing relevant information for the prediction of GHI and relevant sky types, two different datasets were combined to build the testing and training sets for this experiment. The first data set was collected from the National Solar Radiation Database (NSRB) and was the same data used earlier in this work. The second data set was collected from the NOAA Climate Data Record and contained numerical weather data just as the data set from the NSRDB contained with one major addition; this data also contained hourly observations regarding the type and amount of cloud cover present at every hourly recording. Seven years from each data set were combined to create a single data set that contained numerical weather data and the sky conditions for every observation that was taken. The success criteria for this experiment was established as 90% R^2 and an RMSE below 50 W/m^2 for regression and 90% Accuracy for the classification method; these metrics were taken from previous studies presented earlier. 80% of the data was used for training the model and the remaining 20% was used as a separate testing set.

5.3. Data Preprocessing and Feature Selection

Preprocessing consisted of inspecting the data, imputing any missing values with their means and removing any data that was non-usable or in a format that was not acceptable for machine learning models. Exploratory data analysis was performed to ascertain the relative correlation and distribution of the potential new feature. Pearson Correlation Coefficient was used to ascertain the correlation between the new sky type feature and GHI. The two features were slightly negatively correlated with a value of -.037 with a P-value of .0002. As shown in Fig. 48, there are more days with clear skies, mostly clear and partly cloudy skies than the other sky types. From this basic exploratory data analysis, sky types might be relevant to the prediction of GHI and may be able to serve as a substitution for the cloud type feature that was used in previous models and current solar modeling methods. The low correlation and low P-value are promising signs that machine learning, especially deep learning along with time elements, should be able to use the feature, but model training and testing would provide the validation needed to make the substitution.

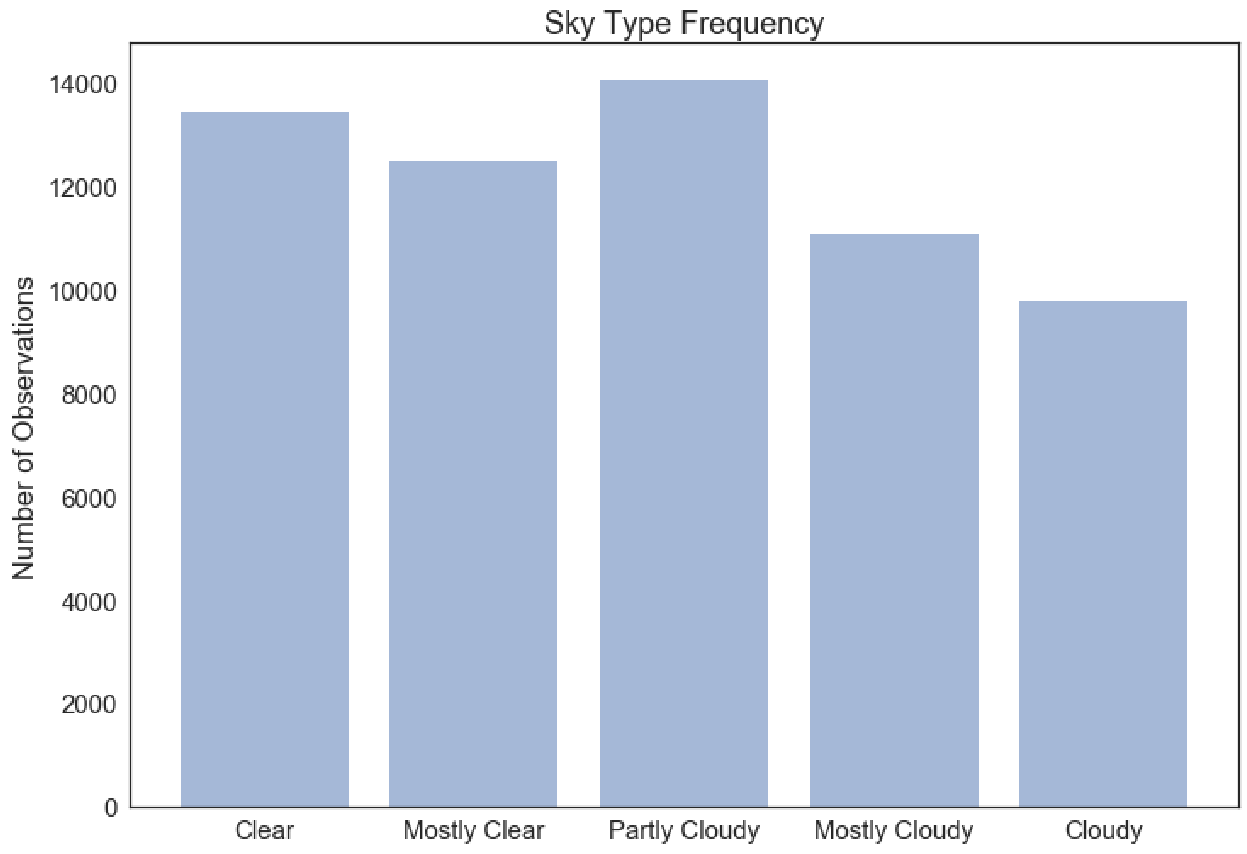


Fig 48: Sky Types Frequency

5.4: Regression Modelling

The Deep Learning regression portion of modelling used the same feature space as was used in the supervised learning process with the exception being sky types was substituted for cloud types. A Recurrent Neural Network trained on the combined dataset and, since it had already been established that deep learning is a better method for predicting GHI, the network was optimized for training for this experiment. The network was designed to use four layers for training, including the final fully connected layer, as shown in Fig 49. The same custom function that was built for RMSE in part one was used as the Loss function and the ‘Adam’ optimizer was used as the optimization function. Just as in the previous process in part one, the model prioritized

Accuracy during its training so the loss (RMSE) would be minimized during training. A batch size 50 of was processed by a lower number (300) of epochs and the smaller network design and training parameters produced a model with a low **RMSE of 23.46** W/m^2 and an **R^2 of .914**. Because of this experiment it was discovered that a smaller network with less epochs would produce at least the same results as the previous methods that utilized a larger network with 100 more epochs. Scaling this problem to a larger amount of data would likely result in a large reduction of training time and a model that is much easier to maintain over time. This is likely related to the smaller number of sky types compared to the wider distributed cloud types feature it replaced. The low RMSE did meet the established metric for using sky types as a substitute feature for regression, but more study would be needed to validate that other features weren't compensating for the absence of 'conditions.

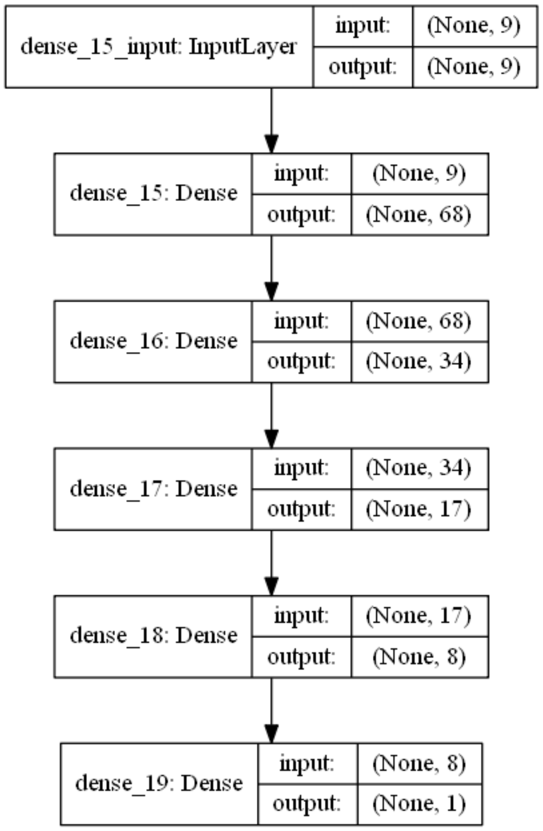


Fig 49: Recurrent Neural Network Topology

In the case of regression, sky types do seem to be an optimal replacement feature for as these results are almost identical to the previous deep learning process that used cloud type and higher than what the ensemble model scored in the supervised learning regression process, but much lower in RMSE. This evidence is shown by the increased R^2 value produced by the model; a summary of the RMSE during training per epoch is shown in Fig 50.

5.5. Validation of Findings

To validate these findings and ensure sky types could serve as a substitute for cloud types, a second regression model was created using an RNN without any features relating to cloud conditions. For this experiment Cloud Types (nor 'sky types') was not present in the feature space and the model was built using the same network topology and number of epochs that was used for the model that used sky types as its replacement. This was done to ensure the impact of removing cloud types was measurable, and the model was not relying on its presence or using any other features more heavily. This second model produced a higher RMSE (88.23 W/m^2) and lower R^2 value of .864 than either previous experiment with cloud types or with sky types. This is a valid indicator that not only can sky types serve as a substitute for cloud types, but it may also be a better performing feature, at least in the case of regression. The next experiment involved substituting sky types for cloud types and building a classification model to ascertain if any improvements in Accuracy might be made over the previous models.

5.6. Classification Modelling

The classification modeling process produced somewhat different results. After optimizing the network topology and increasing the number of epochs the Accuracy (.861) produced by model did not match or exceed the model produced in the deep learning process using cloud types. This was noteworthy given the previous higher Accuracy produced by discretizing the dependent variable. However, the model did produce slightly a lower mean predicted value of .273, which is indicative of a model that might generalize well to new data but wasn't as sensitive to this new feature space as the previous regression model was. This model might be acceptable for consumer or residential usage, especially if given more time to evolve, but it will not be used the remainder of this work as the variance between the predicted values and ground truth is too high when compared to the low RMSE produced by the sister regression method that used 'sky types' as the substitute feature. Based on these findings and conclusions there was no need to validate this process.

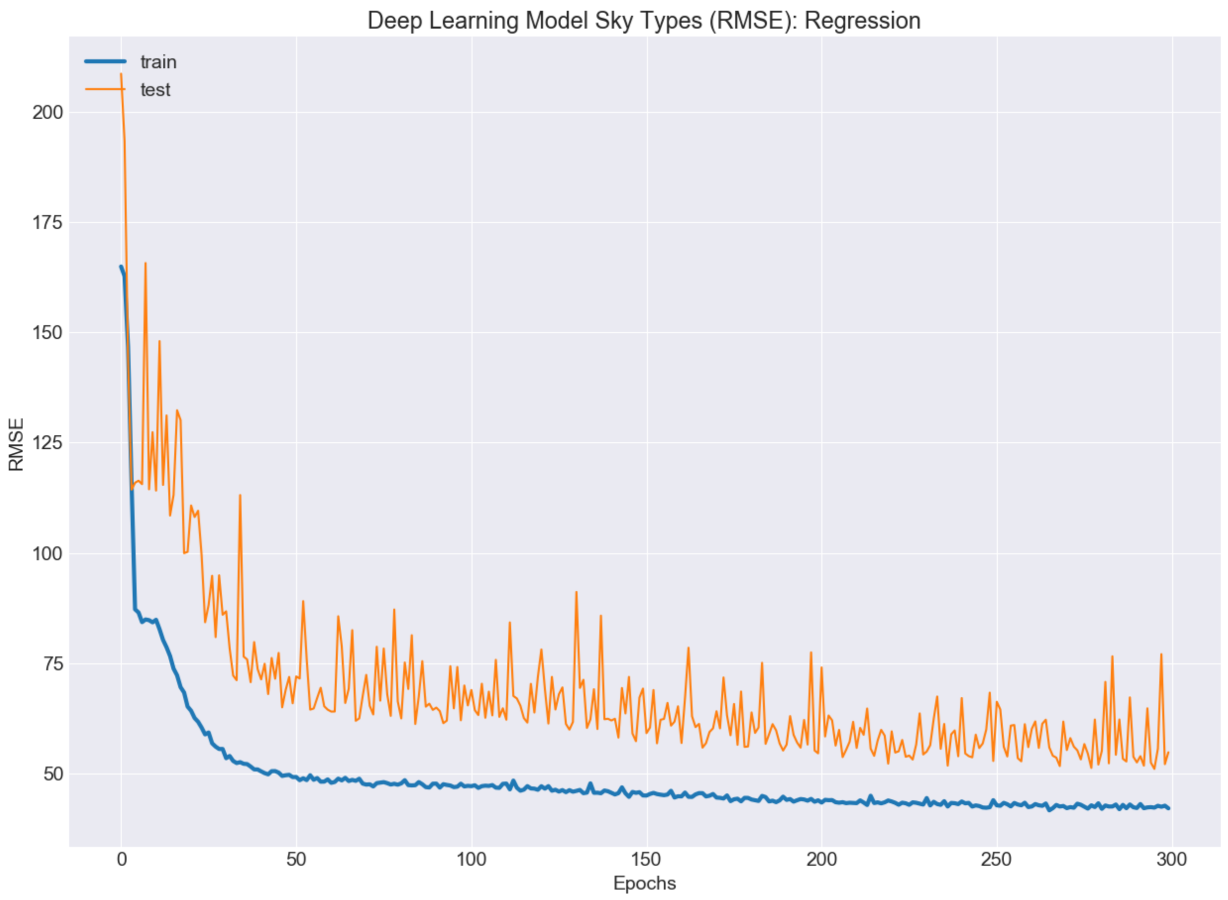


Fig 50: Recurrent Neural Network: RMSE History

5.7. Hour-Ahead Validation

After validating that sky types could serve as a valid substitution for cloud types, the multiple regression deep learning model using sky types and weather conditions in the feature space for its training was validated by the process shown in Fig. 51. The National Weather Service updates their hour-ahead forecasts every hour; these forecasts use the same weather variables as the historical weather data used for training the model, except for GHI. The model predicted what GHI was in one hour based on what the forecasted weather conditions would be in the next hour.

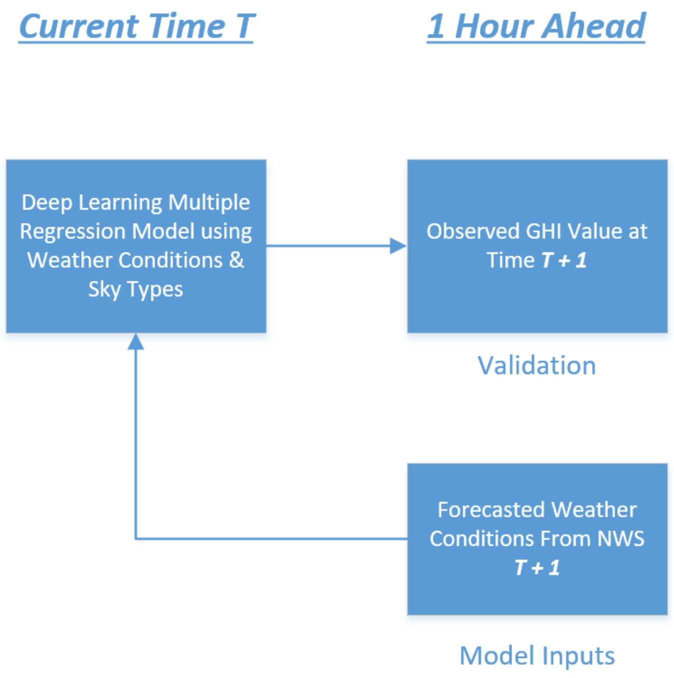

Current Time T
1 Hour Ahead
Deep Learning Multiple Regression Model using Weather Conditions & Sky Types
Observed GHI Value at Time T + 1
Validation
Forecasted Weather Conditions From NWS T + 1
Model Inputs

Fig 51. Hour Ahead Model Validation Process

Model results were validated using a single hour ahead of GFS forecasted weather variables for seven days. To ensure the predictive model could generalize well to any hour-ahead interval, predictions were made at a different time on each day and the predicted GHI values were validated with ground truth GHI values an hour later with RMSE averages shown in Table 18. Results from similar studies are shown in Table 19 for comparison.

Table 18: Average Metrics (Predictions made daily at 9:00 AM)

T = Prediction Time	**RMSE**
T + 1	26.36 W/m^2
T + 2	27.62 W/m^2
T + 3	25.89 W/m^2
T + 4	23.78 W/m^2
T + 5	26.41 W/m^2
T + 6	31.40 W/m^2
T + 7	26.00 W/m^2

Table 19: Comparison Metrics Used in This Study

Studies	**MAPE**	**RMSE**
Ding, Wang & Bi (2011)	14.75%	NA
Amrouche & Le Pivert (2014)	22.50%	51.5 W/m^2
Khatib, Mohamed, Sopian & Mahmoud (2012)	NA	135.5 W/m^2
Chen, Duan, Cai & Liu (2011)	19.45%	NA
Chu et al (2014)	NA	77.5 W/m^2

5.8. Conclusion

The finding that sky types is a valid replacement for cloud types in existing solar irradiance prediction methods satisfies one of the objectives of this study. In addition, and very relevant to the production of GHI, these findings were derived from a data set that included all weather conditions and five of the six defined sky conditions. This is relevant because most of the prevalent GHI prediction methods were focused mainly on clear sky days and often don't include any data that represented changes in the weather nor a high amount of variation in the GHI from day-to-day. In addition, many of these methods will not generalize well to other locations where the weather differs other than from that of the observed and studied area.

Further evidence will be presented later in this study that not only can sky condition serve as a substitution for clouds that are identified and analyzed by satellites, but sky conditions might also provide the needed element to ensure that the deep learning models will generalize well to many areas within a region. Sky conditions do vary from region to region including how they are distributed across the sky. It is likely that the method being proposed here would generalize well across any given region regardless of the specificity of it being designed to be relevant to precise locations. To validate this inference, more data that is relevant to sky conditions from other regions would need to be added to the models before such generalization would be applicable to other areas of the country. The relevance of using deep learning in this type of application is somewhat of a new area and many studies have yet to utilize sky types it in this manner. However, the studies that were presented in the deep learning section of the literature review did address deep learning being implemented in hybrid type approaches and approaches where only numerical weather data was utilized.

The lower error metrics assessed in the regression models built with Recurrent Neural Networks were a good sign that this new method is applicable and can be built upon. Perhaps there are other features that can be substituted for as well? Observing that a regression model trained in this fashion can also be trained with a lower network topology is also a valid finding. Lastly, accepting that the already proven deep learning classification methods don’t generalize well when substituting sky types for cloud types, creates a method for others to challenge or not utilize altogether. These findings can establish a baseline for other studies to challenge as there has not been a study created to data that has used sky types in this manner.

Sky types and this new deep learning regression model method were validated through an applied process, but relevant images would also need to be collected to train a network that could classify images as such. Since the body of this work is focused entirely on prediction, it is necessary that a method for collecting images that correlate to the individual classes of ‘sky types’ be proposed and implemented. The next chapter will outline the creation and implementation of a system that utilized ‘sky types’ for hour-ahead GHI prediction. In addition, a new image collection system and decision support system will also be discussed and presented.

CHAPTER SIX

SKY TYPES: APPLICATION

This chapter will describe the outline and implementation of a new system that is designed to collect, process and classify ‘sky types’ from any location for delivery to the new multiple regression deep learning model for the prediction of GHI and used as a substitute feature for cloud types as was previously discussed. During the proposed process, numerical weather data will also need to be collected at the same time and location images are collected. This data will be used in a predictive model that will be used to make local predictions, which is one of the objectives this study addresses as regional predictions are more prevalent. In addition, this new collection system functions from the ground and does not rely on any satellite data. Planning for the system occurred over a one-year period with different design and scope concepts being conceived. A Systems Engineering process was followed for planning the system; the steps contained in this process will be outlined and described further in this section and throughout the remainder of this chapter; the steps are as follows:

Step 1: Requirements Analysis
Step 2: System Analysis Control
Step 3: Functional Analysis/Allocation
Step 4: Design Synthesis. Process Input, Requirements Loop, Design Loop, Process Output & Verify

The process began with a thorough requirements analysis wherein functional requirements, baseline performance metrics and design constraint requirements were examined and outlined. System Analysis Control outlines the control structure and describes each component of the system. In addition to the individual processes that occur during each step, where applicable, components will also be outlined and described. The functional analysis portion of the section will discuss specific interface elements between each step and component roles in the system. Design synthesis elements will be discussed in detail during the discussion of the previous steps.

6.1. Requirements Analysis

Requirements analysis was divided into two parts that addressed both the end-user requirements and the system requirements. Users often require (Norman, 1999) very specific types of information to be displayed and very specific types of decisions to be made so it was paramount to understand the viability of how this process would map to these types of requirements. This was addressed during planning and implemented throughout the process. System requirements were derived from previous works that were described earlier in this work and conceptualizing new a method and processes that have yet to be utilized in this type of application. The system requirements were collected and a high-level project scope created. The concept, which was briefly described earlier, involved designing and building the following hardware and software components and processes:

1. An image collection system
2. A process that controls and functions the relative image processing steps
3. A system to deliver processed images to a deep learning image classification algorithm that properly classifies the image as being one of five sky types
4. A system to deliver the relevant and classified sky types to the previously trained deep learning multiple regression model
5. A process for collecting numerical weather data specific to a location
6. A user-friendly Decision Support System

User requirements required all the steps be fully functional and self-controlled, leaving the user with very little to do except monitor the system from a dashboard and Decision-Support System (DSS). Usability and maintenance objectives in the project scope required the DSS must have been easy to use and followed all relevant interface and usability guidelines (Rauch, 2011). Any hardware must have also been easy to maintain, if not maintenance free, and must have been easily deployed in an outdoor environment that might be exposed to weather-related elements and conditions. The system also needed some level of security from theft and or alteration. In addition to the hardware components, all software and scripting that drove the system must have also been easy to maintain, relevant to current methods and somewhat future proof, as numerous updates were expected. In addition, the project was likely to be utilized by other students in the future so relevant and current coding standards (Python, 2018) were specified as part of the project scope to be used.

6.2. System Analysis Control

The system, which is shown in Fig. 52, was controlled by a series of scripted functions that Control the flow of data as well as the validation of data and the processing of data from each individual step in the process. Predefined image collection times were also specified as part of the system control functionality; these were implemented to guarantee the relevancy of the images that were being collected. Hardware and software needed to be monitored and a system of notifications implemented in the event of system failure. The last element of system control was centered on maintenance. As was previously proposed, the image collection portion of the system was required to be implemented in an environment that could be exposed to high variations in weather, extended periods of soiling and even theft or modification. Precautionary steps must have been taken to prevent moisture being introduced to the system from exterior elements; these involved creating a completely closed system that were designed with materials relevant to exterior use.

Soiling occurs when equipment has been exposed to outside elements over time (Mejia & Kleissl, 2013). Particles from dust, dirt and wind-driven rain often accumulate on equipment and when the moisture dries, it is left behind until it is cleaned. A system of control had to be put in place to maintain a constant level of cleanliness to ensure that any images captured by the device were clear and error-free. This decision presented a problem as the planned orientation of the unit was to be vertical and almost normal with the sky. This orientation, while optimal for collecting images, also represented the perfect surface for soiling. Numerous ideas were conceived with various designs ranging from a simple powered brush that swept across the lens of the collection device to the design of a circular globe that housed the image collection device and rotated on an axis that was parallel to the ground. This rotation, which would happen at various pre-determined times, would ensure that any soiling collected on the top of the surface of the collection device would be revolved away from the camera itself. A collection plan that included a stationary brush would be housed in the bottom of the unit to wipe away and collect any soiling on a periodic basis. Regardless of the concepts conceived during the design and planning this experiment, the best solution that could be deployed optimally was simply requiring the unit to be cleaned periodically. This is also done in many solar photovoltaic installations where soiling has been determined to be an issue and was an acceptable method for this experiment (Appels et al., 2013). In addition, solving the soiling issue wasn’t a main objective of this study, although future work will certainly be focused in this area.

6.3. Functional Analysis and Design Synthesis: Hardware; Image Collection

The first role in the system begins with capturing an image from the sky. This was accomplished by using a Raspberry Pi 3 that utilized a Raspberry Pi Module V2 - 8 Megapixel camera. The camera had a fixed focus lens and could capture video as well as images and was connected to the Raspberry Pi that was housed inside the collection unit. To protect the small computer, it was placed in an acrylic Smraza case with a cooling fan and heatsinks to control temperature. Heat dissipation was a major concern during this experiment since the unit was housed in a completely sealed structure so it was vital to ensure the heatsinks were installed properly. A standard 5V/2.5A power supply and a microSD card using the default operating system included controlled operation and concluded the of technology hardware for the image collection unit. The unit was mounted on a plastic bracket inside a 4-inch diameter PVC pipe that was purchased from the local hardware store along with PVC end-caps and sealant to help protect the image collector from moisture. The end of the PVC pipe that was to be pointed at the sky was sealed with a 4” hemispheric clear acrylic dome that served to not only protect the collection device,

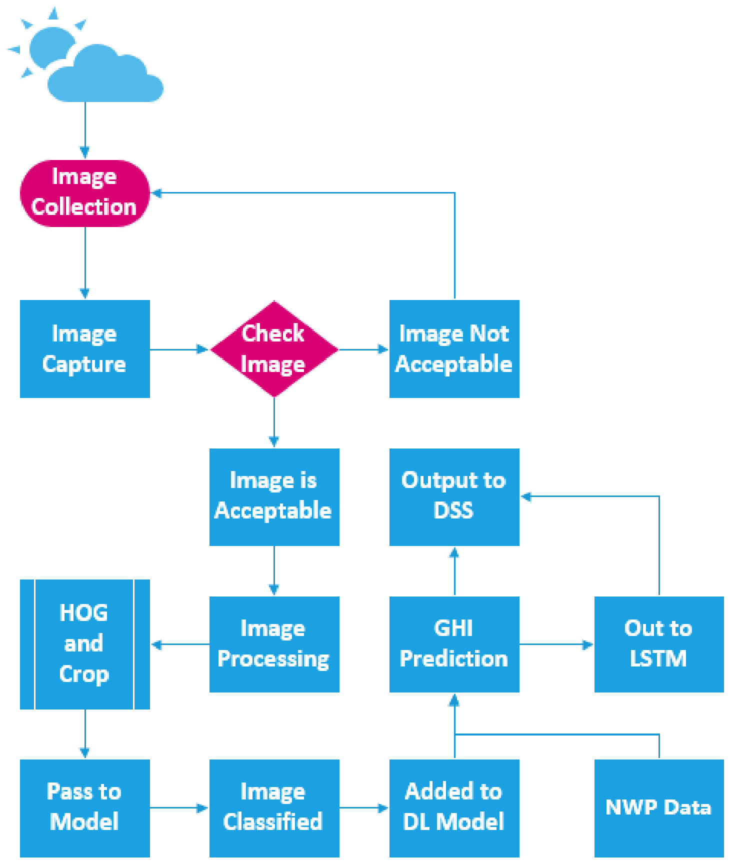


Fig 52: System Design

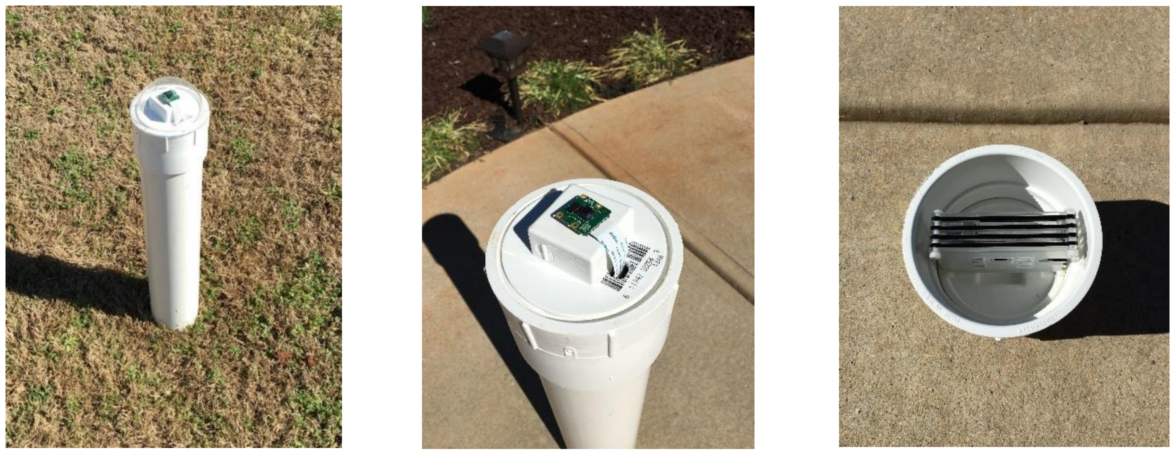


Fig 53: Imager Prototype

but also allowed for images to be collected by the camera and processed by the unit. A picture of the image collection device and its storage container is provided in Fig. 53, which was taken during a dry-fit of the system. The unit was anchored to the ground with a weatherproof ground stake, so it would remain vertical and stable even in stormy conditions and it was placed in an open area free of any overhanging obstructions that might be captured by the camera and in the images. This would be problematic because additional image processing would need to be designed to extract or remove any irrelevant imagery from the regions being processed and so erroneous information didn’t go into the training process later in the system.

First iterations of the unit presented issues. Heat was an issue in the installation and the small computer did not perform well in the completely sealed enclosure, so ventilation holes were added to the top and bottom portions but were drilled at a steep upward angle to prevent water from entering the enclosure. The holes were then covered by a dense mesh wire fabric to not only help with moisture intrusion, but also prevent insects from climbing into the enclosure. The hemisphere clear acrylic dome also presented some issues during the first few days after the unit was installed. Most of these were mainly related to moisture intruding into the system and fogging up the hemisphere dome when the sun began to evaporate the moisture during the day. To ensure that any excessive moisture that funneled up through the unit during the evaporation process was repelled from the interior of the dome, a commercial window treatment solution was applied to the dome. This eliminated the moisture and condensation issue and, although some of the earlier morning images still contained small amounts of the water droplets, image is collected after mid-morning were absent of any moisture issues

6.4. Functional Analysis and Design Synthesis: Software and Numeric Weather Data

This section will provide a systems level understanding of some of the high-level processes and components contained in the software side of the system and how weather data will be collected. A Python script was written to control the frequency of operation of the camera in the system; images were programmed to be taken every 15 minutes throughout the day beginning at 7 AM in the morning and ending at 7 AM in the evening. In addition to controlling and operating the camera in the system, the script also provided a small system check on images that were captured by the camera. This involved ensuring a relevant file size was present indicating that the operation did indeed complete as needed. If a relevant file-size wasn't found by the system an additional image was taken; if a relevant file-size was found by the system, the image was transmitted via network to a local server where the images were stored. A wired network was used to ensure any possible downtime was negated, although a wireless process will be described in detail later this work.

The next step in the process involved processing the images. A script was written that would regularly retrieve recently captured images from the folder where they were being stored and processed the images using the Histogram of Oriented Gradients (HOG) algorithm, which will be described in greater detail in the next section. Images were cropped to provide a focused region of interest (ROI) in the upper-center of the image and to remove any non-relevant information from the image(s). After images were processed, a script delivered the images as inputs to a previously trained deep learning model. The model classified the images as one of five relevant sky types and provided this information to the existing multiple regression deep learning model that has previously been described in this work. The deep learning regression model used the new feature, along with numerical weather data that was provided from an Ambient Weather WS-1001 weather station (Fig. 54) that was installed close to the image processing system. *This combination created the new hybrid GHI prediction system.*

At this point in the system the model made the GHI prediction, which were then delivered to the Decision-Support System to not only provide relevant information to the end-user, but also assist with decision making. This software Displays the information and if applicable, utilized it in a decision-making capacity surrounding whether electricity should be generated or not. After the prediction was delivered to the decision-support system, the prediction was also passed along to a local database that stored every prediction made by the system, which again was done every 60 minutes. After eight hours of predictions the system used the stored predictions to drive a time series process. This information was also passed back to the decision-support system so time relevant ‘future’ predictions could be assessed and used by the system. This is relevant to the location and use of the system over time and, once the timeseries regression model became functional, it would continue to do so, and the relevant assessment metrics continued to improve as more predictions were stored in the system.

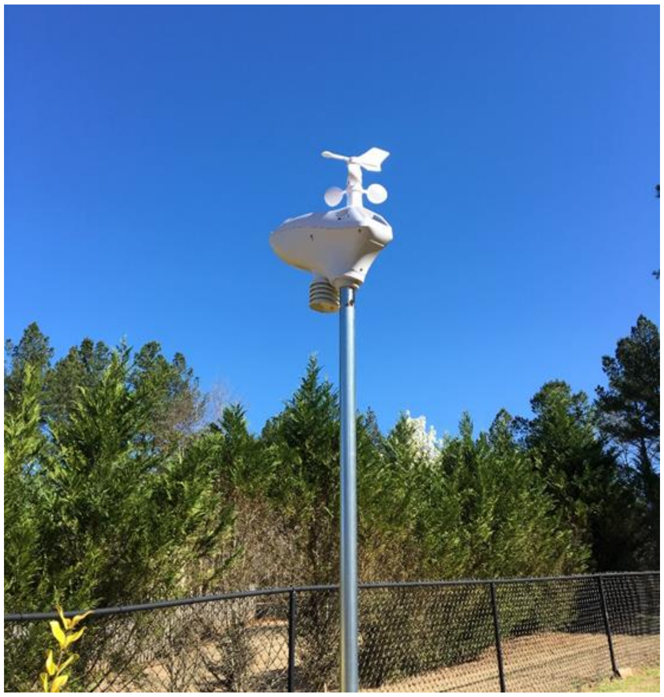


Fig 54: Ambient Weather WS-1001

6.5: Image Capture: Trial One

After numerous iterations of testing the image collection camera, a final orientation was chosen that provided the least chance for obstructions represented by objects in the sky. Two weeks of testing the image collection camera and the network storage process were riddled with numerous images that were either filled with easily identified flying objects, insects, rain and anything else that would otherwise Render the images unusable; an additional problem also existed. The convex nature of the hemispheric plastic dome that was placed on top of the image collection camera created an extreme glare from the sun intersecting with its hemispheric shape at various times throughout the day. This intersection anomaly created lens flares that were captured in almost all images that were processed on a sunny day. Even after processing, these lens flares represented an additional feature that was showing up in all images and this would have been captured by the deep learning classification model that was going to be used later in the system for classifying the images. The main problem this presented was images that no longer halve such lens flares but also represented images that were taken on a clear day when the sun was shining high, would likely not have been correctly classified as being an image from a clear day. Not only did the lens flares create an issue with the images, all created a but the hemispheric dome shape itself created a problem that would also easily be found by the deep learning process. The external rim of the hemispheric dome stretched the images as the sun became parallel with the lens of the camera. This widening of the image would surely not generalize well to the images that come from other sources where the same hemispheric dome was not used to protect a camera from the elements. An example of one of the collected images showing both problems is shown in Fig. 55. To validate if these images could be utilized by the CNN, a trial model was built using 800 images that were collected over the trial time. These images were used for training and an additional 200 images that represented various sky conditions were utilized for testing. As feared the CNN model trained well on the data it was presented with but did not make predictions adequately enough to proceed with this collection method.

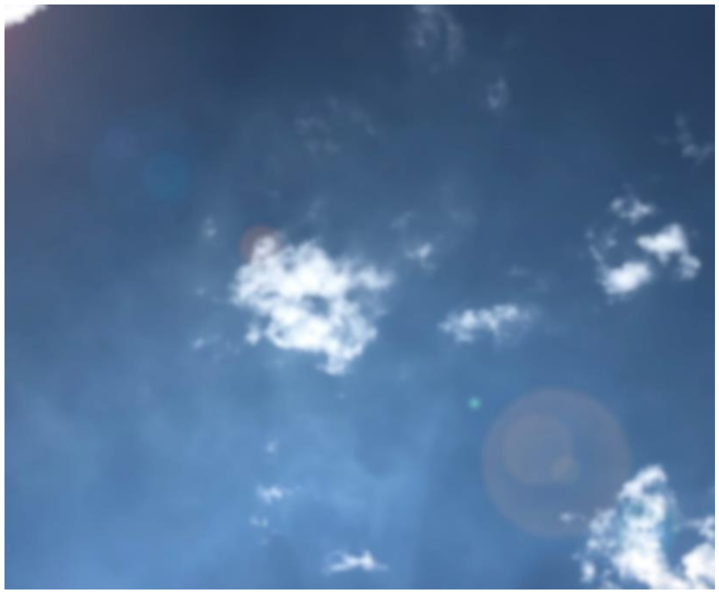


Fig 55: Anomalies Caused by Hemispheric Globe

The best accuracy that was obtained during testing was 40% and that was with a high network topology and additional image processing to try to remove the analogies. In addition, many of the other images that did not contain random lens flares, but did represent the distortion caused by the hemispheric shape of the globe were also incorrectly classified. This validation was enough evidence to reject this image collection method and research another solution.

6.6. Revisiting the Project Scope: Image Capture: Trial Two

The project scope requirements were revisited and a second experiment was planned using images collected from an outdoor webcam. A Reolink 4MP Super HD PoE Outdoor Camera was installed at the same location as the previous image collector was installed but installed parallel to the ground and at a higher elevation to try to eliminate any interference caused from the Sun's glare throughout the day. The camera was oriented toward an open 'window' in the nearby horizon line to lessen the amount of extraneous and unneeded information that might be captured in the images. An example of a captured image is shown in Fig. 56; as shown, there is no glare from the Sun nor any interference from the lens protection that was already on the camera. In addition, there was very little interference being injected into the image and it was already very close to being optimal for use for training, except for the trees that were in the bottom of the image.

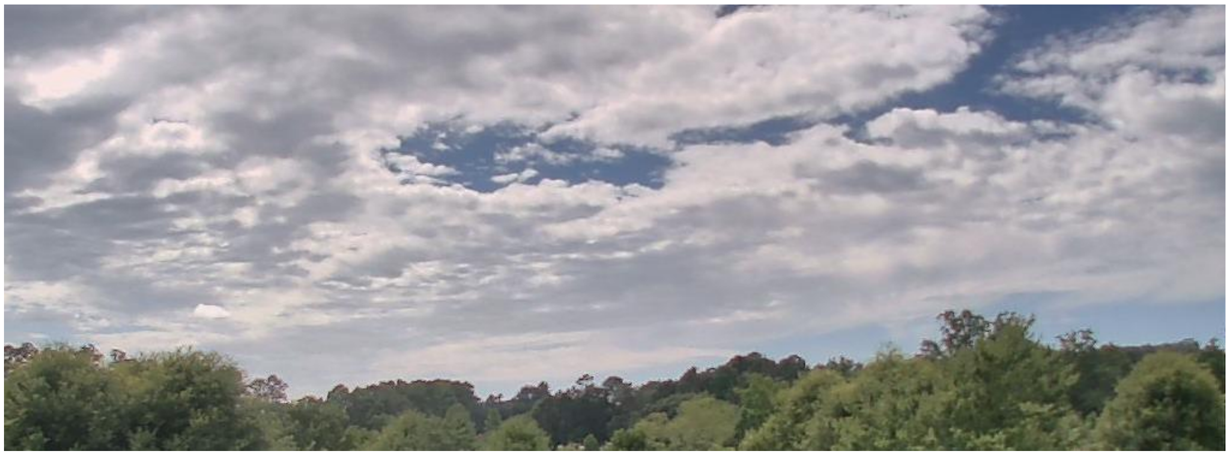


Fig 56. Example of Parallel Image and Horizon

To validate if this method would be useful for model training, another two weeks of captured images were cropped manually and 80% used for training, 10% used for testing and the remaining 10% used to validate the model. Except for the manual cropping, which was done in the same manner for all images to remove the bottom 500 pixels, the images were used ‘as is’ for training, testing and validation. Using a small, four layer designed network topology for a CNN the, Accuracy of the trained model increased to 81.34% and, while the Accuracy of the predictions was lower at 72.41%, this was likely due to the small testing set being used and the large variations present in the skies. The second trial was accepted as being a valid method for collecting images and the image collection allowed to continue, but a process for improving the training accuracy would need to be developed and a second classification test performed after such a process was designed and tested while more images were collected by the newly validated collection process.

6.7. Image Processing

The next problem to solve in the image collection process was centered on image processing. To be increase the Accuracy of the CNN used for training, the collected images needed to be simplified in a way that would be easier for the models to ‘find’ the features of the sky types (Krizhevsky, Sutskever & Hinton, 2012). These characteristics are relevant to each image belonging to each type or class of sky type. The main success metric for this problem was to build a process that could identify the largest occurring features present in the images; this could be done either by color separation or even using processes like texturizing (He, Zhang, Ren & Sun, 2016). The overlapping clouds in the sky contained ever-changing variations that would have likely be too much for any type of machine learning problem to capture adequately enough to be used for training and generalization. Any images that were to be used would have to be simplified wherein the most highly relevant features in each of the images would need to be identified, highlighted in a way to create more focus and separated from the rest of the image. This is reflective of the same definitive process that establishes their very definition; the sky being separated from the clouds. Numerous methods were tested for validity and impact on providing this type of needed solution and a method that is commonly used in object detection in computer vision was found to be the most relevant and impactful.

Object detection in computer vision is often separated into two individual methods or approaches (Torralba, Murphy & Freeman, 2004). The first approach, which is referred to as a global approach, contains methods for distinguishing a single feature in an image as an object while the second approach, which is referred to as a part-based approach, contains methods for extracting individual features from objects within an image. Global approaches are often considered to be more simpler and work well on smaller resolutions or in problems that only contain binary classes, which is also why it is considered global (Papageorgiou & Poggio, 2000). The process is only extracting one object from an image; this is where the binary definition is derived. Part-based approaches are much better for isolating small portions of objects within images and can detect and work around typical occurrences that often occur in images like overlapping objects (Felzenszwalb et al., 2010). Part-based approaches are often considered to be much more memory intensive than global approaches from the larger amount of processing that is required to isolate small portions of an object and handle interactions at that level. Both approaches often apply various types of Gradient Histograms, which measure the orientation and power of objects or image ‘gradients’ with a section or image of a region, to images to help separate the object that is to be extracted or identified from the surrounding and extraneous content (Tsai, 2010). This type of implementation would provide some relevance for the CNN to build upon. There are three main algorithms and computer vision that have this ability and are widely used but the most appropriate one for this work is called the Histogram of Orientated Gradients (Dalal & Triggs, 2005). This algorithm, which is a global approach, was chosen because of its ability to capture edge or gradient structures that can easily correlate the local shapes, which are often seen in different compositions of skies. In addition to this ability, the algorithm also can be adjusted so that the orientation and spatial sampling of the density of the histograms within each cell can also be adjusted (Peng et al., 2016). From a machine learning viewpoint this is important because it increases the likelihood of relevant and important features being identified by the network during training. HOG also functions from the same processes that CNNs do and this convolution approach was expected to work very well with such a network.

The basic concept behind the operation and functionality of the Histogram of Oriented Gradients (HOG) is centered around redefining an image as a series of gradients (Lalonde et al., 2012) with varying intensities and/or edge directions; this happens in five linear phases:

1. Image normalization
2. X and Y gradient computation
3. Histogram gradient creation
4. Cell normalization
5. Flattening to create the feature vector

Most images include some degree of impact caused by direct or indirect illumination from light sources (Van der Walt et al, 014). This is especially relevant to images taken outside where the natural ambient light intersects with many of the objects captured in the images. Before any histogram image processing is done, the degree of this impact is often normalized to provide a foundation for the image processing algorithm to do its work. The first step HOG performs is applying a global normalization function designed to reduce this impact caused by illumination (Huang et al., 2012). Since the algorithm would be processing the entire image, this normalization is accomplished by computing the square root of each color channel. After normalization, the gradient computation process begins with the algorithm dividing the image into small portions that are often referred to as cells (Huang et al., 2012). The center horizontal and vertical gradients are calculated along with the related magnitude and orientation. During this process, silhouette and basic texture information is also captured by the representation of the added gradients; if RBG images (3 channel) are used, the process will choose the color channel with the highest gradient magnitude. Next, the algorithm builds a system of binned histogram orientations or descriptors within the cells and, in this final step of the process, collates all these descriptors, approximates and combines them into a 'feature dimension' described by using the combined edges as a vector and describing the object within the cell, which is shown in (b) of Fig. 57 (Huang et al., 2012).

After the images were collected during the second test period of the external collection process, a small process for collecting images from one folder, processing them with HOG and moving them to another folder, was created for testing. The HOG parameters used during this process specified using eight orientations for the gradients in 16x16 pixel cells and was adjusted to utilized multichannel images since the ~800 images being processed were in color.

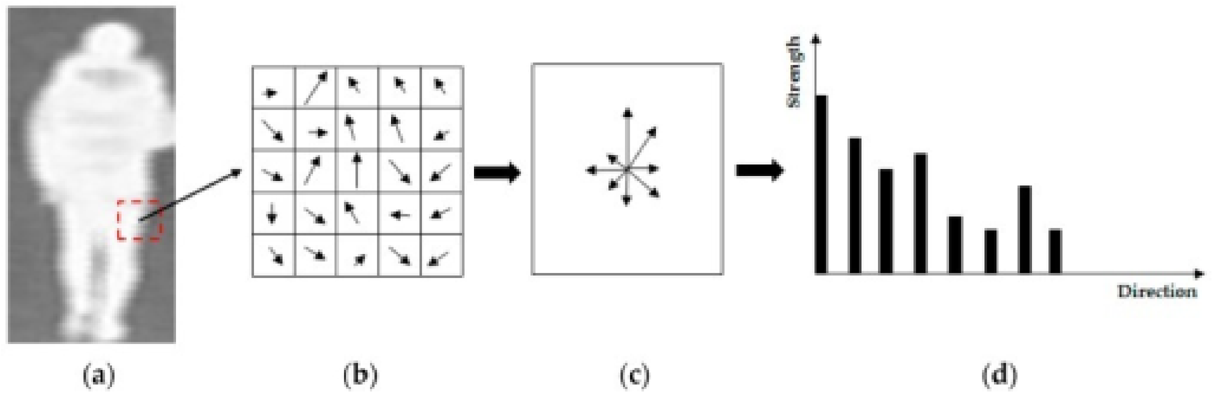


Fig 57: Functionality of HOG. A Demonstration of the HOG Feature Extraction Method (2018)

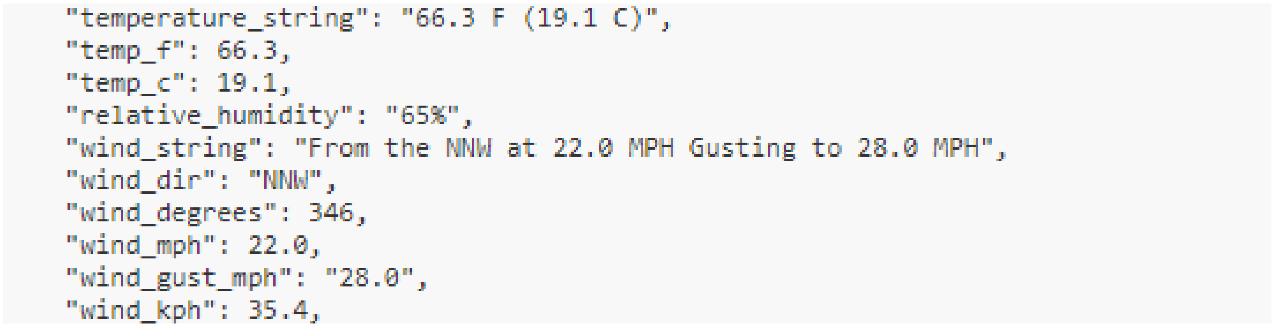
The second CNN experiment was conducted using the same network topology and training/testing processes as in the previous experiment before the images were processed, but used the processed images as features for the CNN; this was done to assess the impact the image processing had on model Accuracy. As is shown in Fig 50, the Accuracy improved to ~95%, which validated the CNN model performing well on the images processed by this method.

6.8. Final Process Design

The next step in the process was to revisit the image collection method and derive a process for scaling and collecting more images. Not only did the single collection point from the local deployment of the external camera not provide enough variable images over time, but also the images that it collected were collected from a single collection point. This practice did not provide the variations needed to create a GHI predictive model that would generalize as well to images that were collected from other locations. This problem would need to be solved before any further modeling was tested and major objectives would need to be met for this process to be considered successful. The first objective was obtaining a method for collecting sky conditions recorded from the National Weather Service at specific times and at specific locations. The second objective was to obtain images from those same time observations and from those same locations. If a process could be derived to bring these two objectives together in a single method, that process could be validated. These two objectives created both items needed to train a CNN: the ground truth sky conditions obtained for each location as the inputs for training the network and the related outcomes would be the sky types that correlated to each observation.

6.9. Obtaining Ground Truth

Numerical weather data is easy to obtain from various organizations and services including the National Weather Service, which provides an application programming interface or API that allows developers easy access to any weather data obtained from any National Weather Service station at any time. To access this type of information from most organizations, an API key is required to validate that the proper credentials have been secured to access the information. These credentials are normally an email address and a password, so the organizations know whom is using the data and for what purpose, which is most commonly for research and is free to use for such. The researcher or developer accesses the API from within the software, which then makes a request to a server for the information being granted access to. The information is returned in various types of formats that can be parsed by the developer or system and individual pieces of the information separated and used accordingly. An example of what this information looks like when it is returned from the server can be seen in Fig. 58.



```
    "temperature_string": "66.3 F (19.1 C)",
    "temp_f": 66.3,
    "temp_c": 19.1,
    "relative_humidity": "65%",
    "wind_string": "From the NNW at 22.0 MPH Gusting to 28.0 MPH",
    "wind_dir": "NNW",
    "wind_degrees": 346,
    "wind_mph": 22.0,
    "wind_gust_mph": "28.0",
    "wind_kph": 35.4,
```

Fig 58: Numerical Weather Data API Response

The ground truth needed for sky types were collected by obtaining real images of sky conditions that were needed to satisfy the second half of this process; these images were not as easy to obtain as the numerical weather data relating to sky conditions. Two approaches were investigated. The first approach involved deploying more external web cameras at various locations across the region and capturing images over the course of several months. The second approach involved using existing public external web cameras that were already deployed across the United States. The second method would be much easier to scale and much less expensive to implement if such a system of cameras could be located and utilized and permission to use such obtained. After researching and investigating user terms, a publicly accessible system of web cams was found that could be utilized for research efforts since any images obtained from such would not be published, but rather were utilized to train the CNN that was used in the system.

There are thousands of these types of cameras located across the United States (Breiholz et al., 2018). Many belong to universities, many belong to public institutions and schools and many belong to government entities; cameras that were specifically available to the public at any time were chosen for this work. Many of these cameras took time-lapse photos in a series of video streams and were specifically located in areas that were often free of surrounding buildings and obstructions, so the sky and weather conditions could be easily observed. In addition, most publicly available cameras such as these often timestamp the images collected. This would prove to be very beneficial for matching the images to the ground truth sky conditions obtained from the numerical weather data.

This network of cameras created the collection source for this study, although a process would still need to be created for bringing the two portions (images and weather data) of the system together. The location of the cameras, the height of their installation and their orientation were very important aspects related to the study. If camera locations were oriented in a direct normal relationship with the horizon, images captured could more easily be divided into two separate pieces: the top half representing the sky conditions and the bottom half representing the extraneous material that wasn't needed for this study. An example of this type of orientation is shown in Fig. 59 where the upper half of the images free of obstructions in the lower half of the image contains the relevancy of what's being captured in the image, which was not going to be used for training. If the installations found the cameras being oriented parallel to the ground and high enough to be free of obstructions, the images could be utilized in a manner needed for this study to be successful. A semi-automated process was created to retrieve over 300 images daily from ten different locations across the United States:

- Atlanta, GA: Dekalb EMA – pointing East
- Tucker, GA: Stone Mountain – pointing West
- Miami, FL: Port of Miami – facing West
- Phoenix, AZ: City of Phoenix – facing North
- San Francisco, CA: Port of San Francisco

- New Orleans, LA: Louis Armstrong International Airport
- Panama City Beach, FL: City Traffic Cam – facing South
- Ocean City, MD: Ocean City Boardwalk
- Oak Harbor, WA: Traffic Cam – facing East
- Tulsa, OK: Oklahoma DOT Cam (I-244) – facing West

When visitors visit the webpages that contain these web cameras the users' browser automatically stored references to every image within its internal network queue. At the end of every day this network queue was analyzed for each camera at each location and any relevant time-lapse imagery captured in the form of the URLs that linked to the images. A process was then created that collected and stored all the URLs at the end of every day. Because all the public web cameras were time-lapse cameras, all images were captured as a series of still frames. The system stored the URL of all collected image sources and separated the image names from the relevant URLs. As expected, the naming conventions of every image captured by the system also contained the relevant timestamp related to when they were captured as still images from the video feeds.

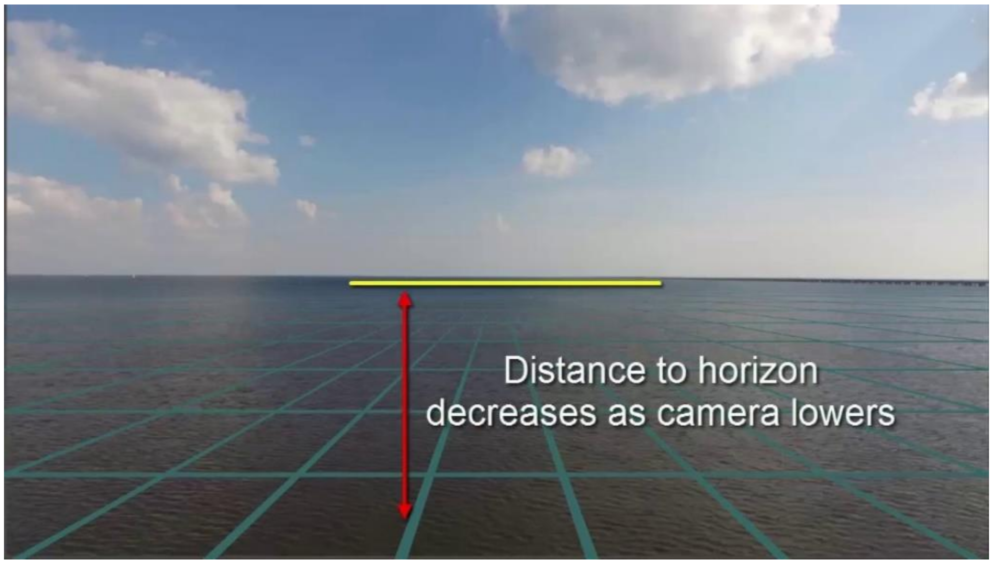


Fig 59: Horizon/Camera Relationship. (The Horizon Line Changes, 2018)

These timestamps were processed by the system and stored for access later after the images were processed. This was the key objective that would later combine the inputs of the network with the correlated images, both of which represent ground truth of every given observation. Next, the system separated the URLs and the image names, so the image names could be utilized again when saving the images locally. Since the timestamps were being stripped from the image names this concept made it easier to reference images when viewing them in a directory. It was easier to see when each image was captured just by viewing the image name and not having to retrieve the timestamp that was stored in a different location again. The end of the process involved a simple system that collected all the URLs and downloaded each image individually to a local data structure and paired the image with the related sky condition that was previously obtained from the application programming interface specific to the same location where the images were collected from. A simple flowchart outlining the system and processes is shown in Fig. 60.

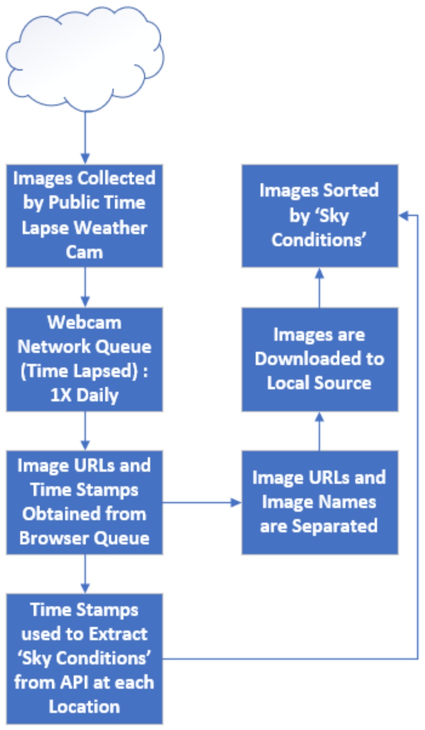

Images Collected by Public Time Lapse Weather Cam
Webcam Network Queue (Time Lapsed) : 1X Daily
Image URLs and Time Stamps Obtained from Browser Queue
Time Stamps used to Extract 'Sky Conditions' from API at each Location
Image URLs and Image Names are Separated
Images are Downloaded to Local Source
Images Sorted by 'Sky Conditions'

Fig 60: Public Weather Camera Image Collection Process

For the CNN to train adequately enough to be used for production and deployed for usage with the decision support system, ~10,000 images were needed for training and testing. Images were collected from the previously outlined process daily over the course of six months. The process was checked every other day to ensure it was still properly functioning and remove any non-irrelevant images being added to the system. This included images that were taken at night and would not be utilized for this study. This figure was much higher than expected and as much as 20% of the images were sorted into a separate folder for nighttime or late evening analysis. Clean, usable images were then paired together with their related timestamps and ground truth sky conditions retrieved from the application programming interfaces as previously described. Images were then preprocessed and resized using Open CV and the HOG algorithm previously outlined in an earlier section. Before and post-processed images are shown in Fig 61.

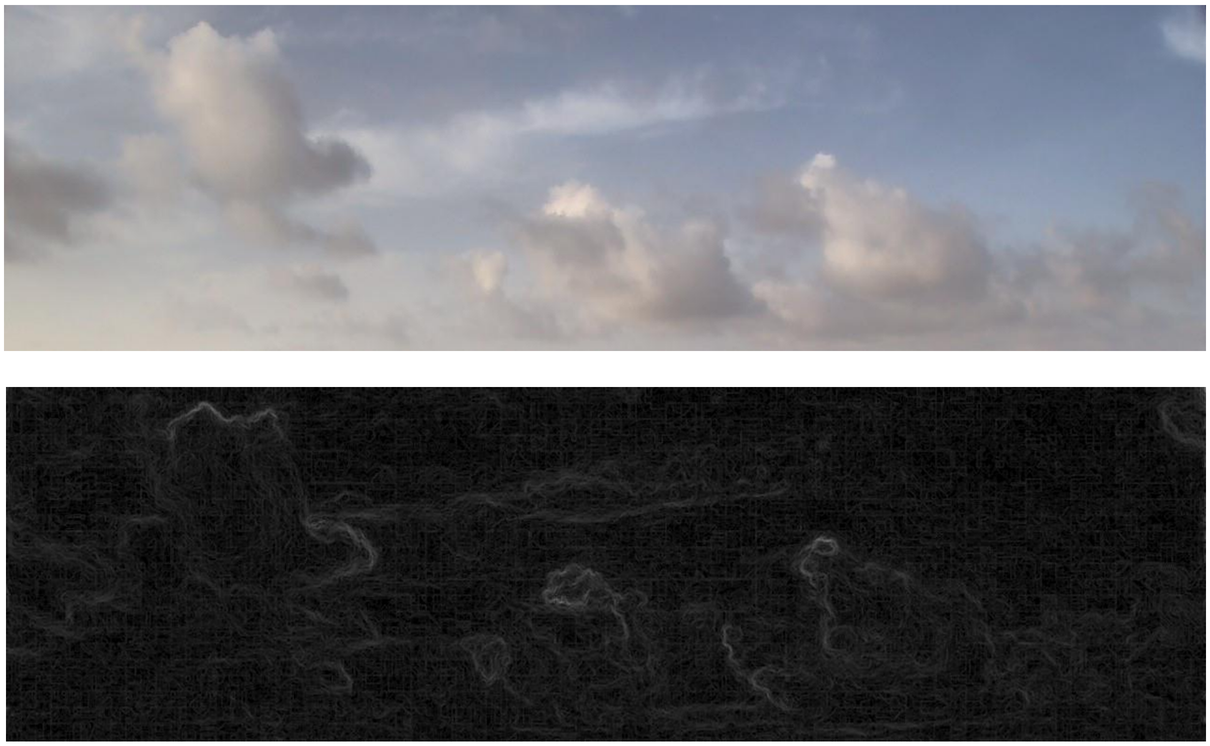


Fig 61: Web Camera Image Pre- and Post-Processing (shown cropped)

6.10. Training and Testing the CNN

A seven-layer CNN (Fig. 62) was designed for classifying the images, of which 8000 were used for training and 2000 used for testing. Minor image augmentation was also used during training to help with image preprocessing to prepare the data more for the CNN. Validation data was retrieved from other locations to create a small ~500 image validation set along with the related sky conditions for assessing the mean classifications of the model, which represented the ratio of correctly classified images to the incorrectly classified images. Twenty-five epochs processed over 8000 images using 70 steps per epoch and training time ~8 hours on a local workstation produced a model with 95.23%% Accuracy.

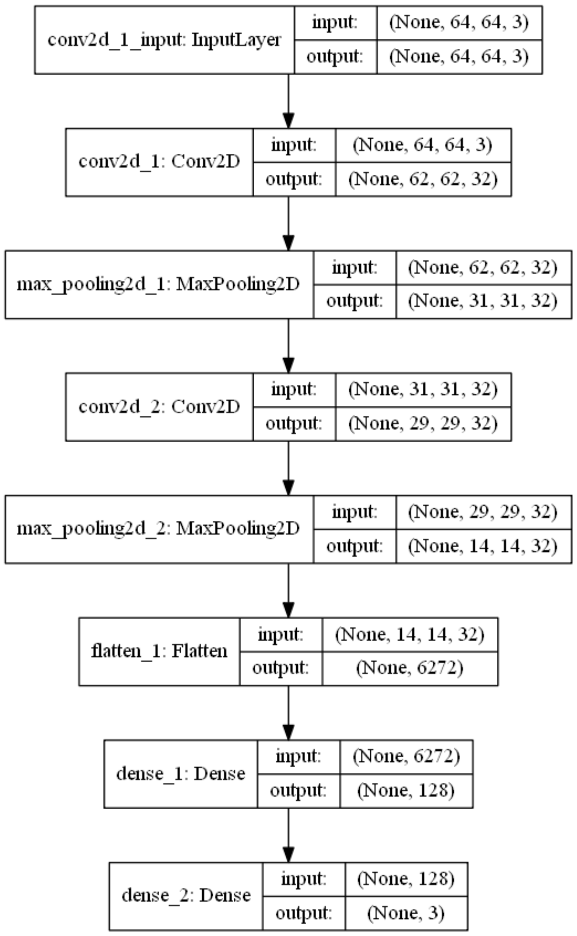

conv2d_1_input: InputLayer
input: (None, 64, 64, 3)
output: (None, 64, 64, 3)
conv2d_1: Conv2D
input: (None, 64, 64, 3)
output: (None, 62, 62, 32)
max_pooling2d_1: MaxPooling2D
input: (None, 62, 62, 32)
output: (None, 31, 31, 32)
conv2d_2: Conv2D
input: (None, 31, 31, 32)
output: (None, 29, 29, 32)
max_pooling2d_2: MaxPooling2D
input: (None, 29, 29, 32)
output: (None, 14, 14, 32)
flatten_1: Flatten
input: (None, 14, 14, 32)
output: (None, 6272)
dense_1: Dense
input: (None, 6272)
output: (None, 128)
dense_2: Dense
input: (None, 128)
output: (None, 3)

Fig 62: CNN Layers

After training and testing the model and applying the model to the validation set, an additional experiment was planned to assess if less images could be used for training the model as training time wasn't optimal enough for the retraining that would likely be needed for production models (Towards Data Science, 2018). The same network layer topology was used except the number of epochs were double and the number of images used for training the model was reduced by ½. In this experiment, the CNN model performed almost as well as it did previously with twice as many images, but the training time was reduced by 40% so a third model was built. The training images were reduced by ½ once again; the number of epochs was also doubled to 100 and the steps per epoch were increased to 100. This model produced a higher Accuracy than the first model, but only a negligible decrease in training time was measured. A last model was built using only 1000 images for testing, 100 epochs and 100 steps per epoch. The Accuracy of this model was very similar (95.48%) as obtained in the first model and training time remained roughly the same.

It appears that CNNs are sensitive to learning from images processed by the HOG and images of this nature (sky types). It is also viable that a lower number of images could be used for model building and the same or better accuracy obtained as shown in Table 20 for all trained and tested models. This process was accepted as the process to be used for classifying sky conditions as ‘sky types’ that would be used for improving GHI prediction methods.

Table 20: Image Classification Topology Tests

Number of Images	**Epochs**	**Steps/Epoch**	**Accuracy**
8000	25	70	95.23%
4000	50	70	93.14%
2000	100	100	96.14%
1000	100	100	95.48%

6.11. Time Series Regression

This study already validated deep learning models can successfully make hour ahead predictions, since time was an available variable a small experiment was planned to ascertain if predictions further than one hour ahead could be made using inputs from previous predictions made by the system as a time series.

Any problem that contains time as an element and some type of data that is sampled at the same frequency over such time can be considered a time series analysis problem (Chatfield, 2016). Time series regression methods use correlated trends and patterns derived from the time periods that are being studied to make forecasts about future values, simulate scenarios, interpret seasonality or wider time-based patterns and even provide elements of control to a system (Reikard, 2009).

This experiment was to the test the validity of using a Long Short Memory Unit. This was considered a test as there was some risk that this application and the structure of the data might not present the sequential structure needed for the LSTM to produce valid results. Timeseries regression models need continual and non-interrupted sequences of time to build the correlated trend that is needed to make predictions more than one-hour ahead (Reikard, 2009). The main concern here with using any type of timeseries regression method was the gap in GHI presented by nighttime conditions.

The same process that was followed for building the machine learning predictive models could not be followed in this manner. Rather than utilizing an 8 AM to 4 PM time frame as previously studied, an entire day or longer noninterrupted spans of prior predictions would need to be available for any timeseries regression methods to be useful, but the hybrid model making the predictions wasn’t designed for such. If the LSTM didn’t produce viable results, the GHI prediction model itself would be used, as was validated in the previous chapter, for hour-ahead predictions.

The LSTM used for this experiment was designed in a very similar manner as the previous deep learning methods were. It utilized six different layers, two of which were special layers called dropout layers, and function just like cross validation in supervised learning. These were necessary because LSTMs are prone to overfit on trends in time series (Gal & Ghahramani, 2016). Mean Squared Error was used as the Loss parameter instead of RMSE that was utilized in the previous Recurrent Neural Networks and, rather than utilizing ‘Adam’ as the optimizer, RMSprop was used. This optimizer is better structured for use on timeseries problems that are prone to overfit and normally have diminished learning rates over time even if an LSTM does not experience vanishing gradient issues (Krause et al., 2016).

As previously expressed, the main concern with this type of approach was centered on the large fluctuations in the sequence of time being used for the model. The model could utilize the lower amounts of GHI as a sequence and train on such, but there was some concern regarding how the model might perform when the trend suddenly changed as the daytime hours begin to be introduced into the model. This wide variance was a concern that could only be validated or challenged with testing. Minor data preprocessing had to be performed prior to the test being implemented. Normalization of the data is mandatory when using an LSTM and all zero values contained in the timeframe had to be incremented by one unit to prevent division by zero errors that occur during normalization. The data was divided into separate test and train sets and the training time series was presented to the designed network that used 100 epochs used for the training. The LSTM model performed well with some observations when GHI was high, but did not adequately predict GHI when the trend was suddenly reduced. This sequencing (Fig. 63) variation was too much for the cells in the LSTM to use in an advantageous way and the learning rates couldn’t translate the rapid change in variation. The LSTM was not validated as a useful method to use for time series regression so the deep learning multiple regression model would continue to be used for hour ahead predictions.

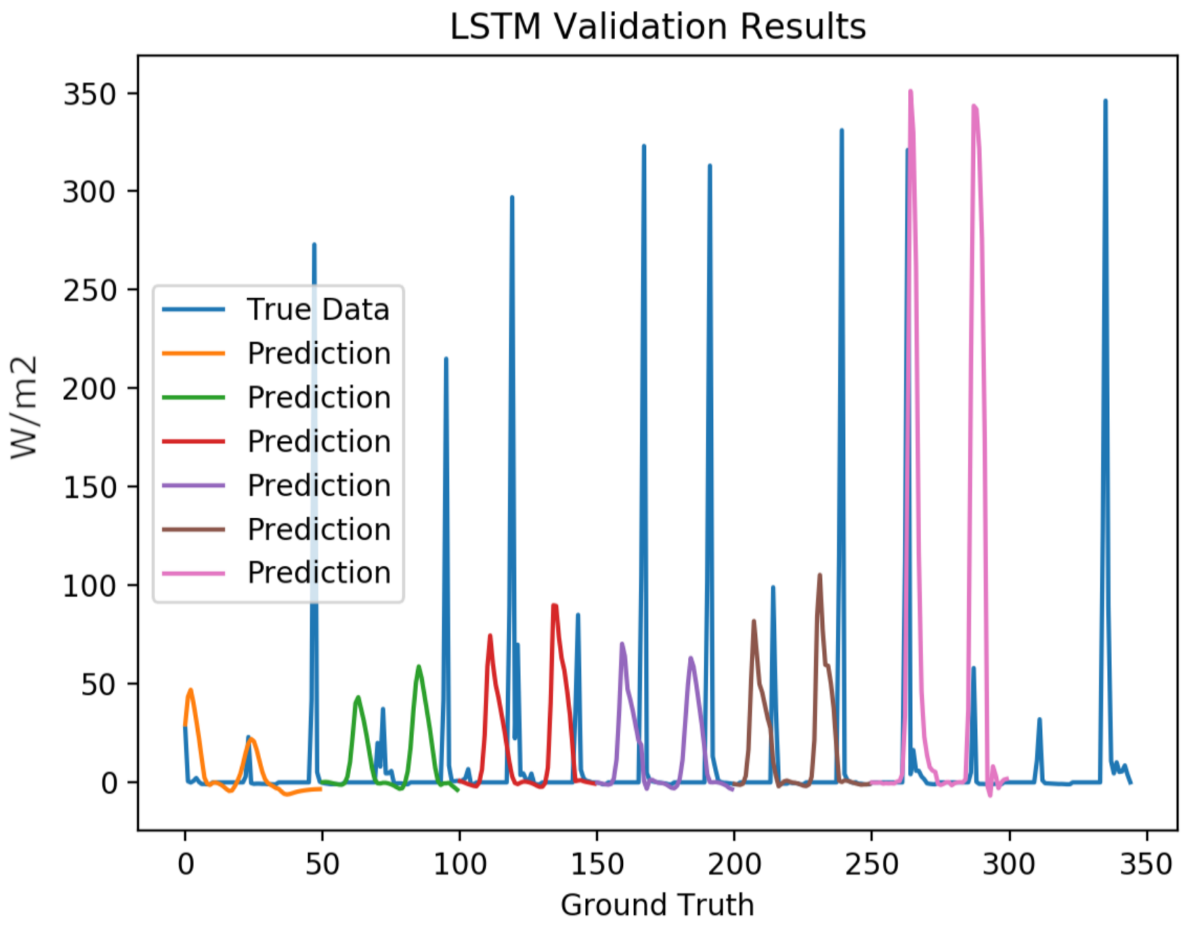


Fig 63. LSTM Results

6.12. Decision Support Systems (DSS)

This section work outlines the design and implementation of a dashboard and decision-support system that was driven by data provided by the hybrid solar irradiance prediction method derived from parts one and two of the project. The decision-support system also utilized numerical weather data from a local collection source and sky images from the new collection system.

Decision Support Systems are often found on the user side of data-driven systems such as the GHI prediction model and the overall system that has been described and implemented in this work. Decision Support Systems are created to solved both semi-structured and unstructured problems. The DSS that is used in this process solves structured problems as the data is being provided to the system and simple decisions are being made about outcomes to assist users and provide feedback about the system. Most decision-support systems are divided into two processes: decision-making and problem-solving (Power, 2002). The decision-making portion of the process utilizes programmed intelligence that can be driven by algorithmic processes or simple heuristics and is controlled by the overall system and interface design (Bonczek et al., 2014). Both objectives support, power and deliver any decisions made by the system. The problem-solving portion of the process involves both implementation and monitoring. Implementation in a DSS is an extra layer of interface system process that allows the system to not only assist with making decisions, but also utilize the same decisions to make automatic adjustments to systems.

The DSS created for this work did not contain an implementation objective. However, it did contain a monitoring section that expressed general information about the health and needs of the GHI prediction system. This type of information can be helpful to users and will provide about any system needs.

There are three different solution types that drive DSSs: optimization, sacrificial and heuristics (Bonczek et al., 2014). An optimization solution involves a programmatic approach to finding the best solution. More recent DSSs take this approach and utilize genetic algorithms or evolutionary computing to help make decisions based on input provided from various sources (Pearl, 2014). These decisions are often based on and derived by previous best practices, but in a programmatic way that supports drilling down to a minimum number of conclusions. The sacrificial method or model involves presenting the user with an acceptable decision, which might not be most optimal, but often can be utilized for the purpose it is being derived for (Gottinger & Weimann, 1992). The last solution type is a solution that power the DSS used here, which is driven by commonly accepted methods or procedures that normally would otherwise derive an acceptable solution. This heuristic type approach is commonly found in simpler systems (Power, 2002) that are data-driven. These data-driven systems utilize the output from systems such as the GHI prediction model and proven ‘rules of thumb’ to inform the user about simple decisions. No evolutionary computing is utilized in this type of DSS as was is driven by a simple system of conditional statements that are programmatically implemented and based on the output of the GHI system.

There are numerous factors to consider when evaluating problem-solving and decision-support and there are often multiple objectives involved in making decisions along with possible alternatives and both intended and unintended actions. In addition to this decision-making capability, most DSS also provide options reports and visualizations to the user. This is especially prevalent in data-driven decision based systems such as the one being presented here. Systems must offer the user not only a general understanding of the data being presented, but also an easy way to understand what data is being visualized and how it can be used best (McLeod & Schell,2007). More event-driven DSS provide alternatives and options for simulations and different scenarios, but this DSS (Fig. 64) did not provide such options as the GHI prediction model was built to predict GHI at any given time and location and predict GHI values in the future; this is primarily the type of information that is conveyed by this system, along with basic informative type displays.

6.13. System Requirements

When designing interface-based system such as this is paramount to understand the user requirements before any design or conceptualization occurs (Adams, 2014). This DSS was designed and implemented primarily for the use by electrical production companies, but could also be used by commercial installers and residential adopters. These three audiences all have different needs and different levels of understanding regarding GHI and how the system works. While this constraint provided some level of limitation over a more generalized approach to design, six high-level objectives that were common to all three audiences were conceived as follows:

1. Gather, display and update numerical weather data every hour (max)
2. Collect and display sky-based imagery from a local source every hour (max)
3. Show GHI predicted and actual values every hour (max)
5. Make simple decisions about electricity production based on GHI predictions
6. Be easily ported to mobile applications using a web application framework

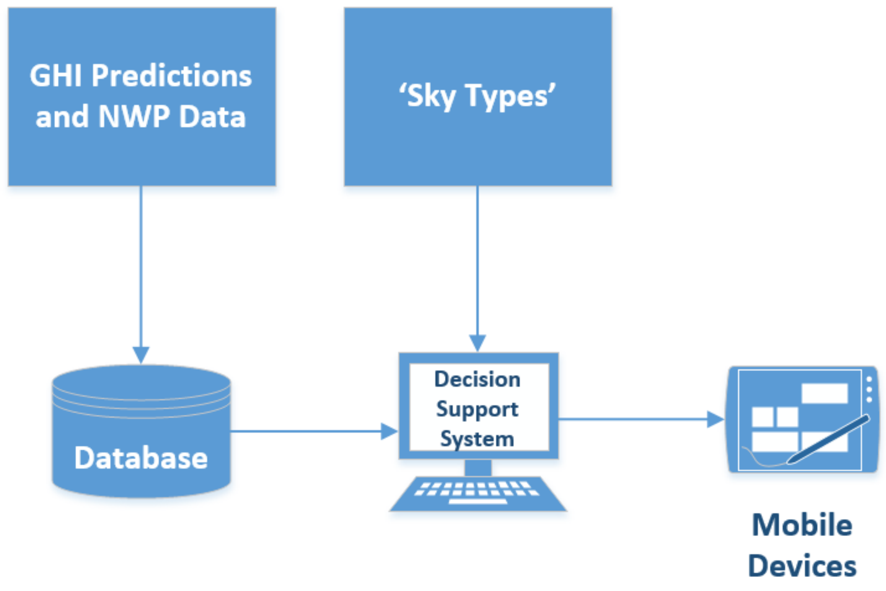


Fig 64. DSS Processes

6.14. User Requirements

Decision-Support Systems provide a great tool to assist organizations and individuals with day-to-day operational support but these types of systems also require a well-designed interface for users to interact with and obtain information from. Without such an interface, DSS would not have a vehicle for delivering and displaying the information that is used to help users make decisions. The specific type of interface that was used with the new hybrid GHI system is called a dashboard; the name is taken from the similarities between its functionality and those found in automobile’s dashboard (Few, 2006). Dashboards represent an interface that can properly convey a collection or group of visualizations and provide a presentation structure that is often restricted to a single page. Dashboards are commonly used for this type of application and are often used as a standalone tool for quickly accessing and using frequently needed information (Turban et al., 2013). There are two different types of dashboards: operational dashboards that convey important information for time sensitive needs and analytical dashboards that simply provide up-to-date information to users (Phippen, Sheppard & Furnell, 2004). This information is often data-driven and relevant in everyday tasks. The dashboard type used for this system is somewhat of a hybrid approach. Both types of methods would be needed as operational information and analytical information needed to be satisfied in the design requirements of the system, which were as follows:

1. The design would need to be web-based and reside on one page
2. The overall structure of the dashboard would need to be relevant to the needs of the user and the requirements of the system as previously outlined
3. Smaller areas of the dashboard needed for educating the user regarding what type of data is being displayed and what it can be used for
4. A section must be present in the dashboard where data can be visualized in a high-level summary
5. The dashboard design must be easily ported to mobile devices

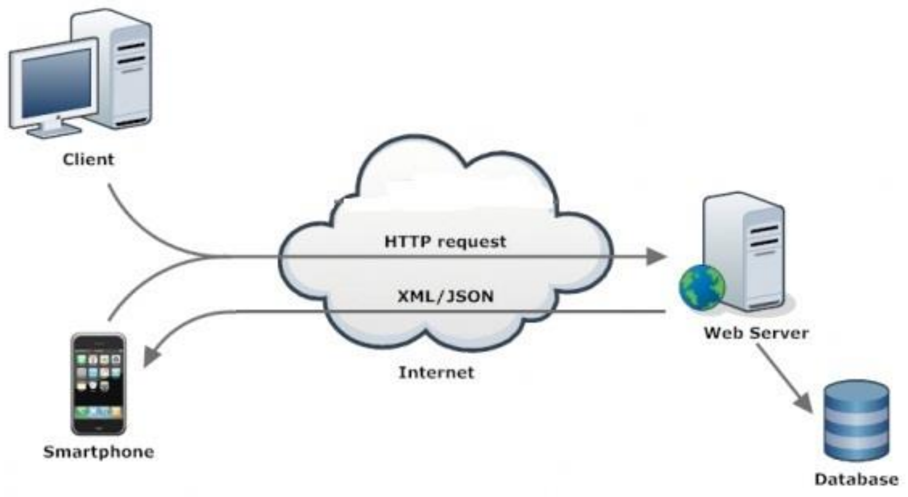
6.15. Design and Testing

The design of many dashboard systems that are data-driven is often limited to the technology surrounding presenting the relevant areas of visualizations that are needed to help users make decisions. There are many open source dashboard frameworks available that meet the user and system requirements for this project so many of the design aspects could easily be met by utilizing an open source solution that was already available. However, the backend processes of data analysis, processing and information delivery would still need to be created to populate any solution that was chosen. A predesigned open-source solution that would satisfy all design and usability requirements was chosen. To test the overall operation of the DSS and the system dashboard, a test heuristic was established and a sample test of values created so the system would have some data to operate on as it was not yet connected to the GHI prediction model. Using a simple heuristic of less than 25% of the maximum amount of GHI being available in an area as being the criteria for production, the dashboard and decision-support system were deployed to a local server and connected to a database. The system was programed to retrieve information from the database every 60 minutes, which represented the same time frequency as the GHI prediction model would be using. A small Python script generated new random data and updated the database and the DSS retrieved the new information from the database and populated the dashboard as required. This was a simple test and validation method mainly just to check the functionality of the network connectivity paths between the DSS and the database that was going to be further utilized with the GHI prediction model when it was online. Testing and validation performed as required so the only objective left to add and test was a time series regression method.

6.16. Hybrid GHI Process Testing and Validation

The main parts and processes of the system were designed, created, tested and the results validated and all objectives surrounding image collection, preprocessing and classification having been satisfied or otherwise addressed. A viable multiple regression model that utilized ‘sky types’ was validated to accept data and images from the CNN; the complete system now had to be tested and validated as a working system of components and processes. Its customary to temporarily deploy machine learning models that have been trained, tested and validated to assess how well these will generalize to data while in production. This process involved deploying the model into a server framework and building a web application backend environment that assisted the processes with delivering information from relevant application programming interfaces into the model in the form of inputs. This process also involved delivering the classified image labels from the same classification predictions into the multiple regression model. Both objectives served as features for making predictions during deployment.

To test the viability of the complete system, a smaller deployment framework, Flask, was installed on a local server and served as an API endpoint that sent information to and from the trained models. This process created the backend functionality for what is commonly referred to as a REST Service and is shown in Fig. 65. The trained models (Image Classification model and Multiple Regression Model) were sent information from a web cam and its related location's numerical weather data through this architecture. The regression model used the sky condition label produced by the classification model as a feature and made a GHI prediction; the same framework delivered a response to DSS. During testing, a simple terminal interface served as placeholder for the DSS.

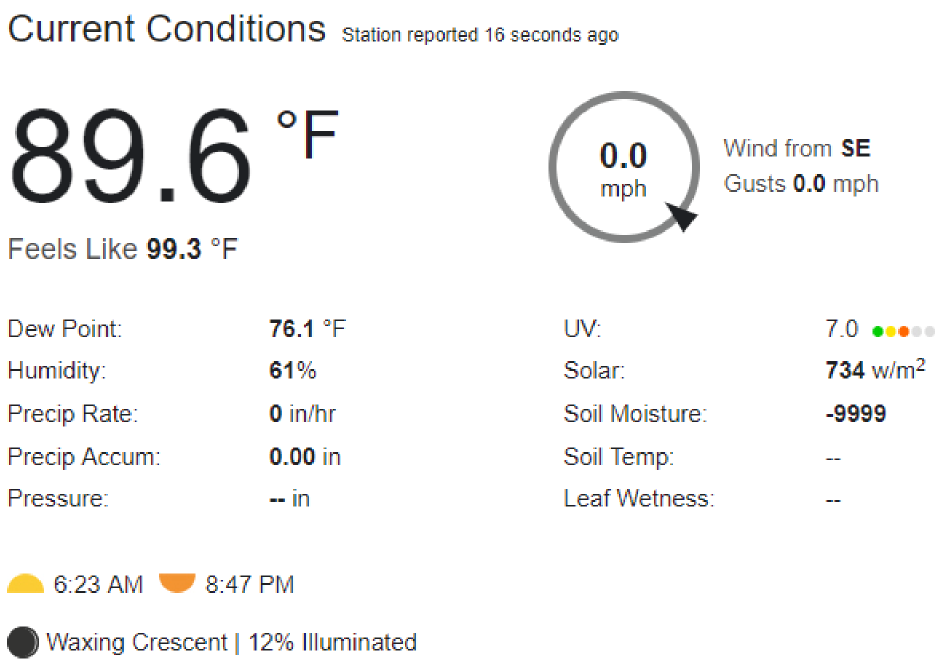


.

Fig 65: REST Web Service. (Phpflow.com, 2012).

The deep learning model were serialized (made ready for use) and deployed into the new environment; testing was done using a single image and a single API call to retrieve of the related numeric weather data. This was completed with data from the local NWS office for testing the system, but those aren't located throughout all the regions where images were being captured so a process like 'crowd sourcing', or collecting data from multiple public sources at once, was used to obtain local GHI readings that were used for obtaining the ground truth needed for validating the predictions made by the system.

There are thousands of publicly accessible weather stations deployed by hobbyists around the United States and some of these stations measure GHI as shown in Fig. 66, but most do not contain sky conditions. Weather Underground is a commercial weather service that provides a platform for amateurs and hobbyists to deploy their own weather stations on in exchange for openly sharing their weather data with the public.


Current Conditions Station reported 16 seconds ago

89.6 °F

Feels Like **99.3** °F

0.0 mph

Wind from **SE**
Gusts **0.0** mph

Dew Point:	**76.1** °F	UV:	7.0 ●●●●●
Humidity:	**61**%	Solar:	**734** w/m^2
Precip Rate:	**0** in/hr	Soil Moisture:	**-9999**
Precip Accum:	**0.00** in	Soil Temp:	--
Pressure:	**--** in	Leaf Wetness:	--

6:23 AM 8:47 PM

Waxing Crescent | 12% Illuminated

Fig 66: Publicly Accessible Weather Station Reading (Atlanta, GA)

Weather stations with the capability of measuring GHI were found in each of the regions as close to the source of the images as possible to capture local sky conditions and GHI from two related locations.

The validation process was tested manually with 100 different API calls and 100 different classified images before the models were put into production in an automated system that made API calls every 60 minutes at a single location; this was done between the hours of 8:00 AM and 4:00 PM. This was necessary to check the variance and accuracy of the models; the assessment metric used for validation was RMSE and predictions were compared with hourly forecasted GHI from the GFS. This metric was calculated from the predicted GHI and the actual GHI retrieved from the ground truth weather station at the same location an hour later than the predictions were made. Validation results were monitored to ensure that the system was working properly; the validation results are shown in Table 21.

Table 21: Hour Ahead Average RMSE Values

Prediction Time = T	**New Hybrid Model**	**GFS Forecasted GHI**
T + 1	28.74 W/m^2	32.15 W/m^2
T + 2	27.26 W/m^2	45.10 W/m^2
T + 3	29.16 W/m^2	51.36 W/m^2
T + 4	27.59 W/m^2	22.15 W/m^2

After validating the system at a single location, the system was tested at scale over the course of seven days making hour-ahead predictions using weather conditions forecasted by the GFS and sky imagery captured at the current prediction time ***T***. MAPE was used to assess the differences between the predicted values made at time ***T*** for time ***T+1*** and the actual values measured later at time ***T+1***; the results are shown in Table 22; MAPE was 2.4%.

6.17. Conclusion and Lesson Learned

Phase III of this work (Fig. 67) uncovered many new findings and some failures, both of which can be useful for future researchers in this same area. The first trial of deploying the image collection system resulted in collecting images not useable for training the CNN. The design of the device was optimized for deployment in exterior weather conditions with a minimal footprint and minimal maintenance being high-level requirements as previously studied imagers were very large and utilized an internal workstation for image processing and camera control.

Table 22: Average MAPE (7 Days of Testing)

Location	**Hour-Ahead GHI Predictions**
Atlanta, GA: Dekalb EMA	9.82%
Tucker, GA: Stone Mountain	8.67%
Miami, FL: Port of Miami	11.15%
Phoenix, AZ: City of Phoenix	7.23%
San Francisco, CA	12.16%
New Orleans, LA	10.03%
Panama City Beach, FL	11.02%
Ocean City, MD	13.32%
Oak Harbor, WA	9.87%
Tulsa, OK	12.65%

Although the image collection process did result in failure, the overall design of the unit was successful as the Raspberry Pi could successfully be deployed and operate in external weather conditions as required and do so over an extended period. The external hemispheric dome also caused numerous issues during testing, most of which were due to its shape interacting directly with the sun like the lenses in a pair of glasses. Not only did the shape cause magnification and distortion of the images, but captured images also contained numerous lens flares captured as the Sun's rays passed through the dome throughout the day.

The second trial of the image collection process successfully validated that an inexpensive external camera mounted 20'-0" above the ground and oriented parallel to the ground could successfully be used to collect images for training a CNN to correctly classify sky conditions. In addition, it was discovered that HOG worked very well when classifying sky type images processed by HOG with a CNN. Also, a lower number of images could be used to train the network. This will make the training process more optimal and take less time than traditional methods where no image processing is done prior to training. Part three of this work also validated a new hybrid approach to GHI prediction can be successfully implemented with the use of local images serving in the place of sky conditions.

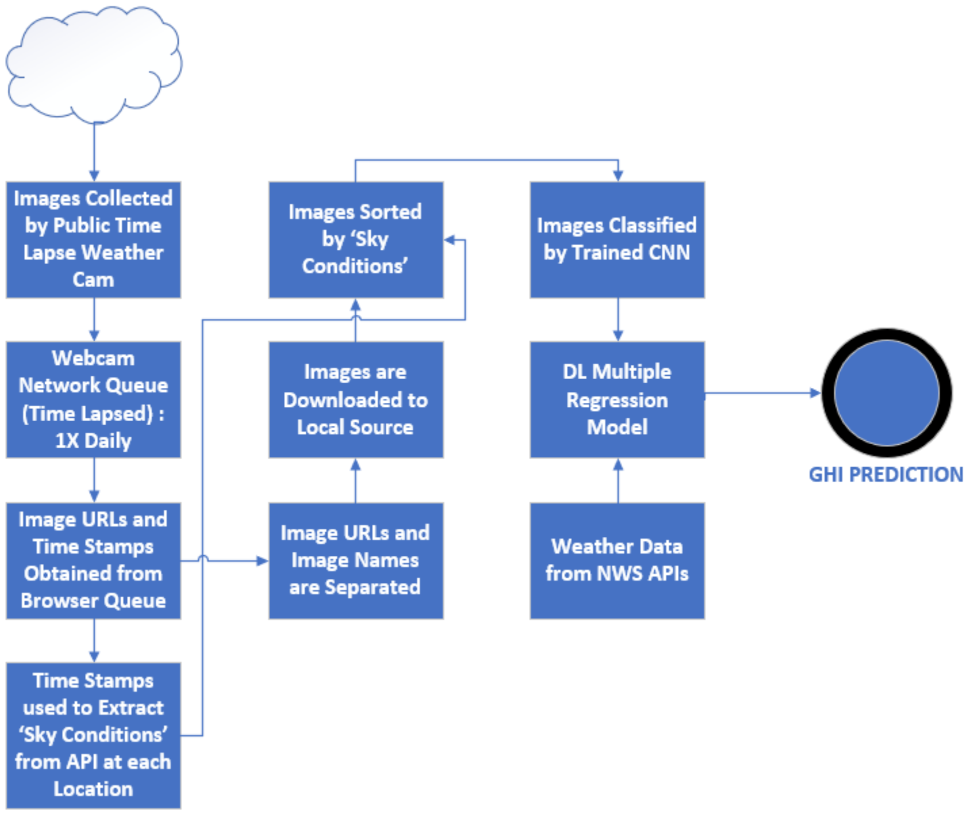

Images Collected by Public Time Lapse Weather Cam
Webcam Network Queue (Time Lapsed) : 1X Daily
Image URLs and Time Stamps Obtained from Browser Queue
Time Stamps used to Extract 'Sky Conditions' from API at each Location
Images Sorted by 'Sky Conditions'
Images are Downloaded to Local Source
Image URLs and Image Names are Separated
Images Classified by Trained CNN
DL Multiple Regression Model
Weather Data from NWS APIs
GHI PREDICTION

Fig 67: Final System Design

The hybrid GHI prediction model was also validated for the use in making hour-ahead predictions with an average error of 1.28% over one day from one location. This new process was validated in various locations across the United States with a Mean Average Error of 10.59%, which is better than similar approaches previously studied that did not generalize to other locations and were only applicable to clear sky days.

This new hybrid method is also relevant to any location and any weather conditions. This method (Deep learning Classification and a Deep Learning Multiple Regression Model) can also be scaled and deployed anywhere numerical weather data and local images can be obtained; no satellite data is needed. This not only makes the system less expensive to deploy, but also ensures model retraining requirements will be much lower over time as the outcome variance is increased and the model trained with more images.

CHAPTER SEVEN

CONCLUSIONS AND FUTURE WORK

This research was performed to increase the optimization of GHI prediction through the design and implementation of a new hybrid prediction system that utilized sky conditions as a replacement variable for cloud types identified by overhead satellites. The current processes can not only be optimized with this better replacement feature but can also be made more relevant to specific locations rather than regional forecasts.

Phase one of this work began by studying the relevancy and sensitivity deep learning methods have on numerical weather data that is often used for predicting Global Horizontal Irradiance. To make a comparison between current methods and deep learning methods, baseline classification and regression models were created using supervised learning algorithms. This was also done to establish success metrics that could be compared to those derived from previous studies. A deeper understanding of the data was also presented in this work; this was done to provide a foundation and understanding of the numerical weather data involved in these processes and to better understand the types of supervised learning and deep learning algorithms that could be useful to improving this area.

After performing exploratory data analysis and establishing supervised classification and regression baseline models and metrics, deep learning models were built using the same data. This baseline experiment provided evidence in the form of performance metrics with lower RMSE and higher R^2 values in the deep learning regression models than the metrics ascertained by the regression models built from supervised learning methods. Metrics were compared to other studies and RNNs outperformed current supervised methods used in GHI prediction. This was also the case when comparing classification methods from both processes; a higher Accuracy was obtained using the deep learning models. In both classification processes, the dependent variable was partitioned to create four categories representing different ranges of GHI values. It was necessary to create classification models to ascertain if this partitioning would impact the optimization of the prediction models and prior studies had also used this method. The major finding from this portion of the work was that RNNs outperform supervised learning methods by as much as 60% in regression and 10% in classification models. The findings surrounding discretization can also be useful for creating processes that are more generalization and can be used for general estimates where ranges of GHI are more useful or enough to be beneficial.

Future work includes the further investigation of different substitute features as well as the investigation of the relevance and correlation the elements of time have when compared only to GHI and not using any of the remaining numerical weather data. It could be likely that Recurrent Neural Networks could work using only time in the feature space, but this will need to be investigated further before it can be validated.

Current hybrid GHI prediction methods utilize numerical weather data and external data in a combined process. In previously discussed works, these hybrid systems most often utilized data related to sky conditions, clear sky indices, angles of the sun and other meteorological variables. This work, which focused on a new hybrid method, proposed using a new feature as a substitute for the satellite-based and satellite collected cloud types that are prevalent in current solar prediction models. This cloud type feature was relevant to the predictive capacity of the models as was proven by the increase in error in the models when it was removed. To find a suitable replacement for this feature, a new variable that was inexpensive and easy to obtain was proposed.

This new variable, referred to as sky types, was directly correlated to sky conditions that were provided from regional offices of the National Weather Service in the form of labels. To ascertain if sky condition labels could be utilized or substituted for cloud types, new deep learning models using Recurrent Neural Networks were built using sky conditions as a replacement feature. As was previously done, classification and regression models using deep learning methods were created to ascertain if sky conditions would match or outperform the previous feature representing cloud types.

While classification methods did not validate the use of sky types in this manner, multiple regression deep learning models validated that the use of sky types was not only a valid substitution for cloud types, but also a more optimal feature for training as a lower network topology could be used to obtain an optimal RMSE. Sky conditions can be used in the place of cloud types and not only is this a better feature, but it is easier and less expensive to obtain and can be collected at any location. This makes the prediction of GHI availability relevant to the same location.

To apply the use of sky types in a GHI hybrid prediction model, a system had to be designed to derive the same sky type labels that were provided by the regional offices of the National Weather Service and obtain a suitable replacement for their use. This new system would include an image collection system and a system for processing images. A system for classifying images as each sky type label, a system for collecting numerical weather data specific to a location instead of regional areas and a control system for information delivery and processing were also implemented. Lastly, a user based system for making decisions and presenting the information was created and deployed.

To establish a process for directly correlating labels provided from the National Weather Service with real time images from the same locations, the new collection system needed to capture images at regular intervals that directly correlated with sky condition labels observed at the same interval and reported by the National Weather Service. These frequently reported sky condition labels were used as outcomes for a Convolution Neural Network and the directly correlated, real-time images from the sky were used as the related inputs of the Convolution Neural Network. This network was trained to learn the ability to classify real-time images being delivered to the hybrid model; this would remove the need for any further labeling provided by the National Weather Service. This new system can classify sky conditions on its own based on the rules it learned during previous training. These new classifications served as sky types for the new hybrid GHI prediction model.

The image collection system implementation process contained two different trials; the first trial failed and the second trial was successful. Modifications between the trials included moving from a ground-based collection system to a scaled system that utilized publicly available equipment that was easily accessible, already in place and met the predefined requirements of the collection system. The second trial process for image collection was validated using data from ten different locations across the United States. Minor image preprocessing was implemented to resize the images and create a region of interest before utilizing the images for training and testing a CNN. The results were acceptable, but further image preprocessing was performed and the Histogram of Oriented Gradients (HOG) tested to ascertain if any model improvements could be gained. This new method was found to be impactful and was validated for use by an increase in classification Accuracy. In addition, a lower network topology and a lower number of images were required for obtaining similar results. Future work in this area is centered on process optimization and a scalability. This system has been tested on a small number of locations and will need more locations to prove the potential impact any possible distributed collection might have on the model's classification capacity or ability. Major findings in this portion of the work included validating HOG as being beneficial for preprocessing images used for training a CNN when used to classify sky conditions; its implementation increased classification Accuracy, reduced the CNN design requirements and reduced the number of images and time required for model training.

After validating the new image collection process, a process for collecting numerical weather data from the same locations where the sky condition images were collected from was designed and implemented. Since regional National Weather Service offices provided valid numerical weather data and local weather data was needed, a nationwide network of amateur weather stations was utilized to ascertain localized, specific numerical weather data. This numeric data was collected from the same 10 locations where the relating images were taken.

These two local collections of data, sky condition images and numerical weather data, served as the inputs for the new hybrid GHI prediction system. A collection and distribution process to ensure adequate and scalable information retrieval and delivery was designed and the new GHI hybrid prediction model populated with real-time information collected from both sources. The system was validated with real-time ground truth GHI amounts from the same ten locations with a total Mean Error of just over 10% and tested for time related predictions at a single location with a MAPE of 2.40% and an average RMSE of 41.26 W/m^2. This exceeds both Amrouche & Le Pivert (2014) and Chu et al (2014) and is better than GFS forecasted GHI on high weather variability days (as much as 32% when used hour-to-hour at a single location). Findings from this portion include the validation of a localized hybrid GHI prediction model implemented using locally sourced numerical weather data and real-time images. This new hybrid system is not constrained to clear sky days and can be used anywhere numerical weather data and images can be captured.

Future work in this are involves creating a process for acquiring ‘sky types’ from any images embedded in social media feeds that include geotags and timestamps. This method will also require a new feature extraction process to extract only the sky relevant data from the images and retrieve the needed numerical weather data from the weather station located closest to the geotagged post. In addition, additional locations for collecting real-time sky condition images also need to be added to the model to create a better range of generalization capacity based on the learning obtained and represented by images from other areas of the country.

The system and process utilized local resources on a local server and have not been tested for large-scale production, although it was built with such as a requirement. The underlying deep learning libraries that were used in this study have been widely adopted and used in large-scale systems and the hybrid GHI prediction model is a candidate for such. Minimal modification would be needed to the backend systems that were designed for this work to place it into production in a large-scale application; future work will be done to satisfy this requirement. In addition, the same new hybrid model that was described here could also be used to predict sky conditions instead of GHI. This introduces the possibility of creating a second model that could be utilized to predict a feature for the GHI hybrid model. Future work is planned around this type of implementation to ascertain if a timeseries classification model can be created to predict not only GHI from the hybrid approach but also the type of upcoming sky types. These candidates, along with their deployment in a distributed network, would optimize GHI prediction across an entire region and would be derived from a collection of local predictions from the same region.

An LSTM unit was tested as a possible candidate for multistep timeseries regression after the hybrid model was validated. This new model used predictions from the new hybrid prediction model as its inputs and train using the relationship between every time step and the amount of GHI as sequences. A second approach utilizing the hybrid model for timeseries regression was established as a secondary plan as there was high risk associated with using a complete days' worth of data, which included high variance. The LSTM unit was found to not be a good candidate for multistep timeseries regression as the model performed poorly due to not being able to ascertain and learn the strong variances that were present and presented by the lower GHI values in the data set. To satisfy the time series prediction requirement of the system, the hybrid GHI prediction model would be used to forecast relative GHI since the three elements of time were already features in the model and it was an RNN. This prediction is not based off trend or seasonality as many timeseries regression models are, but is related to the three elements of time serving as features and related GHI as their outcomes; the combination of such creating the sequence needed for the RNN.

A decision support system was designed and implemented to not only receive output from the hybrid GHI prediction model, but also use this output to assist users with making decisions. These decisions were mainly centered on commercial applications directed at electricity producers making decisions surrounding electricity generation at any given time based on GHI forecasts that are made by the model. This decision support system was created and delivered in a dashboard type format that was displayed on a single webpage and with a framework that was easily ported to a mobile application. This would allow the system to be adopted and used by consumers and installation technicians as well as stakeholders at energy production facilities that may want to monetize a new process. Future work in this area includes developing a multiplatform mobile device application that also has the capability to crowd source local images and geolocation information for the system. This will provide additional data for the hybrid model as the mobile application is used.

REFERENCES

A demonstration of the HOG feature extraction method: (a | Open-i. (2018). [Figure] Openi.nlm.nih.gov. Retrieved 22 June 2018, from https://openi.nlm.nih.gov/detailedresult.php?img=PMC4970176_sensors-16-01134-g002&req=4

Aguiar, L. M., Pereira, B., David, M., Diaz, F., & Lauret, P. (2015). Use of satellite data to improve solar radiation forecasting with Bayesian Artificial Neural Networks. Solar Energy, 122, 1309-1324.

Adams, A. (2014). Usability testing in information design. In Visual information for everyday use (pp. 37-54). CRC Press.

Adeli, H., & Hung, S. L. (1994). Machine learning: neural networks, genetic algorithms, and fuzzy systems. John Wiley & Sons, Inc.

Al-Amoudi, A., & Zhang, L. (2000). Application of radial basis function networks for solar-array modelling and maximum power-point prediction. IEE Proceedings-Generation, Transmission and Distribution, 147(5), 310-316.

Alonso-Montesinos, J., Martínez-Durbán, M., del Sagrado, J., del Águila, I. M., & Batlles, F. J. (2016). The application of Bayesian network classifiers to cloud classification in satellite images. Renewable Energy, 97, 155-161.

Alpaydin, E. (2014). Introduction to machine learning. MIT press.

Alzahrani, A., Shamsi, P., Dagli, C., & Ferdowsi, M. (2017). Solar Irradiance Forecasting Using Deep Neural Networks. Procedia Computer Science, 114, 304-313.

Aman, S., Frincu, M., Charalampos, C., Noor, U., Simmhan, Y., & Prasanna, V. (2014). Empirical comparison of prediction methods for electricity consumption forecasting. University of Southern California, Tech. Rep, 14-942.

Amrouche, B., & Le Pivert, X. (2014). Artificial neural network based daily local forecasting for global solar radiation. Applied energy, 130, 333-341.

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., ... & De Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. In Advances in Neural Information Processing Systems (pp. 3981-3989).

Appels, R., Lefevre, B., Herteleer, B., Goverde, H., Beerten, A., Paesen, R., ... & Poortmans, J. (2013). Effect of soiling on photovoltaic modules. Solar energy, 96, 283-291.

Arel, I., Rose, D. C., & Karnowski, T. P. (2010). Deep machine learning-a new frontier in artificial intelligence research [research frontier]. IEEE computational intelligence magazine, 5(4), 13-18.

Arking, A., & Childs, J. D. (1985). Retrieval of cloud cover parameters from multispectral satellite images. Journal of Climate and Applied Meteorology, 24(4), 322-333.

Babyak, M. A. (2004). What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. Psychosomatic medicine, 66(3), 411-421

Badescu, V., & Dumitrescu, A. (2016). Simple solar radiation modelling for different cloud types and climatologies. Theoretical and applied climatology, 124(1-2), 141-160.

Bagging, boosting, and variants. Machine learning, 36(1-2), 105-139.

Baharin, K. A., Rahman, H. A., Hassan, M. Y., & Kim, G. C. (2013, December). Hourly irradiance forecasting for peninsular malaysia using dynamic neural network with preprocessed data. In Research and Development (SCOReD), 2013 IEEE Student Conference on (pp. 191-197). IEEE.

Bahman Kermanshahi, “Recurrent neural network for forecasting next 10 years loads of nine Japanese utilities,” Neurocomputing, vol. 23, pp. 125-133, December 1998

Banda, J. M., & Angryk, R. A. (2009, August). On the effectiveness of fuzzy clustering as a data discretization technique for large-scale classification of solar images. In Fuzzy Systems, 2009. FUZZ-IEEE 2009. IEEE International Conference on (pp. 2019-2024). IEEE.

Bankert, R. L., Mitrescu, C., Miller, S. D., & Wade, R. H. (2009). Comparison of GOES cloud classification algorithms employing explicit and implicit physics. Journal of applied meteorology and climatology, 48(7), 1411-1421.

Barker, H. W., Stephens, G. L., Partain, P. T., Bergman, J. W., Bonnel, B., Campana, K., ... & Edwards, J. (2003). Assessing 1D atmospheric solar radiative transfer models: Interpretation and handling of unresolved clouds. Journal of Climate, 16(16), 2676-2699.

Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD explorations newsletter, 6(1), 20-29.

Bauner, C., & Crago, C. L. (2015). Adoption of residential solar power under uncertainty: Implications for renewable energy incentives. Energy Policy, 86, 27-35.

Basden, J., & Cottrell, M. (2017). How utilities are using blockchain to modernize the grid. Harvard Business Review.

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms:

Belsley, D. A. (1991). Conditioning diagnostics: Collinearity and weak data in regression (No. 519.536 B452). New York: Wiley.

Ben-David, A. (2008). About the relationship between ROC curves and Cohen's kappa. Engineering Applications of Artificial Intelligence, 21(6), 874-882.

Benmouiza, K., & Cheknane, A. (2013). Forecasting hourly global solar radiation using hybrid k-means and nonlinear autoregressive neural network models. Energy Conversion and Management, 75, 561-569.

Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. Artificial intelligence, 97(1-2), 245-271.

Bonczek, R. H., Holsapple, C. W., & Whinston, A. B. (2014). Foundations of decision support systems. Academic Press.

Borowy, B. S., & Salameh, Z. M. (1994). Optimum photovoltaic array size for a hybrid wind/PV system. IEEE Transactions on energy conversion, 9(3), 482-488.

Box, G.E.P., Jenkins, G.M., Reinsel, G.C., 1994. Time Series Analysis: Forecasting and Control. Prentice Hall Inc., Englewood Cliffs, New Jersey.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern recognition, 30(7), 1145-1159.

Bradley, J., & Amde, M. (2015). [Figure] Random Forests and Boosting in MLlib. Databricks. Retrieved 22 June 2018, from https://databricks.com/blog/2015/01/21/random-forests-and-boosting-in-mllib.html

Breiholz, A. E., Kronfeld, K. M., & Murray, G. D. (2018). U.S. Patent No. 9,979,934. Washington, DC: U.S. Patent and Trademark Office.

Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

Buch, K. A., Sun, C. H., & Thorne, L. R. (1995, March). Cloud classification using whole-sky imager data. In of the 5th Atmospheric Radiation Measurement Science Team Meeting, San Diego, CA, USA (pp. 19-23).

Brownlee, J. (2016). Parametric and Nonparametric Machine Learning Algorithms. Machine Learning Mastery. Retrieved February 20, 2018, from https://machinelearningmastery.com/parametric-and-nonparametric-machine-learning-algorithms/

Buehlmann, P. (2006). Boosting for high-dimensional linear models. The Annals of Statistics, 559-583.

Cao, J., & Lin, X. (2008). Study of hourly and daily solar irradiation forecast using diagonal recurrent wavelet neural networks. Energy Conversion and Management, 49(6), 1396-1406.

Caruana, R., & Niculescu-Mizil, A. (2006, June). An empirical comparison of supervised learning algorithms. In Proceedings of the 23rd international conference on Machine learning (pp. 161-168). ACM.

Castrounis, A. (2018). Artificial Intelligence, Deep Learning, and Neural Networks, Explained. Kdnuggets.com. Retrieved 19 June 2018, from https://www.kdnuggets.com/2016/10/artificial-intelligence-deep-learning-neural-networks-explained.html

Cazorla, A., Olmo, F. J., & Alados-Arboledas, L. (2008). Development of a sky imager for cloud cover assessment. JOSA A, 25(1), 29-39.

Cermak, J., Wild, M., Knutti, R., Mishchenko, M. I., & Heidinger, A. K. (2010). Consistency of global satellite-derived aerosol and cloud data sets with recent brightening observations. Geophysical Research Letters, 37(21).

Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE) Arguments against avoiding RMSE in the literature. Geoscientific model development, 7(3), 1247-1250.

Chakraborty, P., Marwah, M., Arlitt, M. F., & Ramakrishnan, N. (2012, July). Fine-Grained Photovoltaic Output Prediction Using a Bayesian Ensemble. In AAAI.

Charge electric cars by day to fill 'belly of the duck'. (2018). Financial Review. Retrieved 24 June 2018, from https://www.afr.com/news/power-ledger-charges-electric-cars-by-day-to-fill-the-belly-of-the-duck-20180619-h11ko2

Chaouachi, A., Kamel, R. M., Ichikawa, R., Hayashi, H., & Nagasaka, K. (2009). Neural network ensemble-based solar power generation short-term forecasting. World Academy of Science, Engineering and Technology, 54, 54-59.

Chen, C., Duan, S., Cai, T., & Liu, B. (2011). Online 24-h solar power forecasting based on weather type classification using artificial neural network. Solar Energy, 85(11), 2856-2870.

Chen, Gang. (2016). [Figure] A Gentle Tutorial of Recurrent Neural Network with Error Backpropagation.

Chen, J. L., Liu, H. B., Wu, W., & Xie, D. T. (2011). Estimation of monthly solar radiation from measured temperatures using support vector machines–a case study. Renewable Energy, 36(1), 413-420.

Chen, T., & Chen, H. (1995). Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. IEEE Transactions on Neural Networks, 6(4), 904-910.

Chu, Y., Pedro, H. T., Nonnenmacher, L., Inman, R. H., Liao, Z., & Coimbra, C. F. (2014). A smart image-based cloud detection system for intrahour solar irradiance forecasts. Journal of Atmospheric and Oceanic Technology, 31(9), 1995-2007.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

Clough, S. A., Shephard, M. W., Mlawer, E. J., Delamere, J. S., Iacono, M. J., Cady-Pereira, K., ... & Brown, P. D. (2005). Atmospheric radiative transfer modeling: a summary of the AER codes. Journal of Quantitative Spectroscopy and Radiative Transfer, 91(2), 233-244.

Ciregan, D., Meier, U., & Schmidhuber, J. (2012, June). Multi-column deep neural networks for image classification. In Computer vision and pattern recognition (CVPR), 2012 IEEE conference on (pp. 3642-3649). IEEE.

CNN From the Ground Up, Liang2. (2018). [Figure] Blog.liang2.tw. Retrieved 21 June 2018, from https://blog.liang2.tw/2015Talk-DeepLearn-CNN/

Collobert, R., & Bengio, S. (2001). SVMTorch: Support vector machines for large-scale regression problems. Journal of machine learning research, 1(Feb), 143-160.

Confusion Matrix and Cost Matrix. DnI Institute. (2016). [Figure]. Dni-institute.in. Retrieved 2 June 2018, from http://dni-institute.in/blogs/confusion-matrix-and-cost-matrix/

Convolutional Neural Networks: Accelerating the Super-Resolution Convolutional. (2018). [Figure] Mmlab.ie.cuhk.edu.hk. Retrieved 18 June 2018, from http://mmlab.ie.cuhk.edu.hk/projects/

Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on (Vol. 1, pp. 886-893). IEEE.

Das, S. (2001, June). Filters, wrappers and a boosting-based hybrid for feature selection. In Icml (Vol. 1, pp. 74-81).

Data Irradiation. [Figure] (2018). Users.cecs.anu.edu.au. Retrieved 21 June 2018, from http://users.cecs.anu.edu.au/~Andres.Cuevas/Sun/Irrad/Irradiation.html

Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In Proceedings of the 23rd international conference on Machine learning (pp. 233-240). ACM.

Deep Learning in Digital Pathology. (2018).[Figure] Global-engage.com. Retrieved 22 June 2018, from http://www.global-engage.com/life-science/deep-learning-in-digital-pathology/

De Jeses, O., & Hagan, M. T. (2001). Backpropagation through time for a general class of recurrent network. In Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on (Vol. 4, pp. 2638-2643). IEEE.

Deng, L., & Yu, D. (2014). Deep learning: methods and applications. Foundations and Trends® in Signal Processing, 7(3–4), 197-387.

Desbois, M., Seze, G., & Szejwach, G. (1982). Automatic classification of clouds on METEOSAT imagery: Application to high-level clouds. Journal of Applied Meteorology, 21(3), 401-412.

Devasthali, R. (2018). [figure] Understanding the concept of simple linear regression. Towards Data Science. Retrieved 19 June 2018, from https://towardsdatascience.com/understanding-the-concept-of-simple-linear-regression-a572087c253

Dietterich, T. (1995). Overfitting and undercomputing in machine learning. ACM computing surveys (CSUR), 27(3), 326-327.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In Multiple classifier systems (pp. 1-15). Springer Berlin Heidelberg.

Ding, M., Wang, L., & Bi, R. (2011). An ANN-based approach for forecasting the power output of photovoltaic system. Procedia Environmental Sciences, 11, 1308-1315.

Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., ... & Münkemüller, T. (2013). Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. Ecography, 36(1), 27-46.

Domingos, P. (2012). A few useful things to know about machine learning. Communications of the ACM, 55(10), 78-87.

Edwards, R. E., New, J., & Parker, L. E. (2012). Predicting future hourly residential electrical consumption: A machine learning case study. Energy and Buildings, 49, 591-603.

Elith, J., Leathwick, J. R., & Hastie, T. (2008). A working guide to boosted regression trees. Journal of Animal Ecology, 77(4), 802-813.

English, S. J., Eyre, J. R., & Smith, J. A. (1999). A cloud-detection scheme for use with satellite sounding radiances in the context of data assimilation for numerical weather prediction. Quarterly Journal of the Royal Meteorological Society, 125(559), 2359-2378.

Fan, C., Xiao, F., & Wang, S. (2014). Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques. Applied Energy, 127, 1-10.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence, 32(9), 1627-1645.

Few, S. (2006). Information dashboard design.

Flach, P. (2012). Machine learning: the art and science of algorithms that make sense of data. Cambridge University Press.

Fisher, D. H., Pazzani, M. J., & Langley, P. (Eds.). (2014). Concept formation: Knowledge and experience in unsupervised learning. Morgan Kaufmann.

Foster, M. J., & Heidinger, A. (2013). PATMOS-x: Results from a diurnally corrected 30-yr satellite cloud climatology. Journal of Climate, 26(2), 414-425.

Friedman, J. H. (1997). On bias, variance, 0/1—loss, and the curse-of-dimensionality. Data mining and knowledge discovery, 1(1), 55-77.

Friedman, J. H. (2002). Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4), 367-378.

Gal, Y., & Ghahramani, Z. (2016). A theoretically grounded application of dropout in recurrent neural networks. In Advances in neural information processing systems (pp. 1019-1027).

Gala, Y., Fernández, Á., Díaz, J., & Dorronsoro, J. R. (2016). Hybrid machine learning forecasting of solar radiation values. Neurocomputing, 176, 48-59.

García, S., Fernández, A., Luengo, J., & Herrera, F. (2009). A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. Soft Computing, 13(10), 959.

García-Hinde, O., Gómez-Verdejo, V., Martínez-Ramón, M., Casanova-Mateo, C., Sanz-Justo, J., Jiménez-Fernández, S., & Salcedo-Sanz, S. (2016, July). Feature selection in solar radiation prediction using bootstrapped SVRs. In Evolutionary Computation (CEC), 2016 IEEE Congress on (pp. 3638-3645). IEEE.

Gensler, A., Henze, J., Sick, B., & Raabe, N. (2016, October). Deep Learning for solar power forecasting—An approach using AutoEncoder and LSTM Neural Networks. In Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on (pp. 002858-002865). IEEE.

Gill, M. K., Asefa, T., Kemblowski, M. W., & McKee, M. (2006). Soil moisture prediction using support vector machines. JAWRA Journal of the American Water Resources Association, 42(4), 1033-1046.

Gilbert, C. D., & Wiesel, T. N. (1992). Receptive field dynamics in adult primary visual cortex. Nature, 356(6365), 150.Godbole, S., & Sarawagi, S. (2004, May). Discriminative methods for multi-labeled classification. In Pacific-Asia conference on knowledge discovery and data mining (pp. 22-30). Springer, Berlin, Heidelberg.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning (Vol. 1). Cambridge: MIT press.

Gottinger, H. W., & Weimann, P. (1992). Intelligent decision support systems. Decision Support Systems, 8(4), 317-332.

Gueymard, C. A., & Ruiz-Arias, J. A. (2015). Validation of direct normal irradiance predictions under arid conditions: A review of radiative models and their turbidity-dependent performance. Renewable and Sustainable Energy Reviews, 45, 379-396.

Gupta, D., (2017). Fundamentals of Deep Learning – Introduction to Recurrent Neural Networks. [Figure]. Analytics Vidhya. Retrieved 15 June 2018, from https://www.analyticsvidhya.com/blog/2017/12/introduction-to-recurrent-neural-networks/

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.

Hall, M. A. (2000). Correlation-based feature selection of discrete and numeric class machine learning.

H. Su, J. Chu, Forecasting building energy consumption using neural networks and hybrid neuro-fuzzy system: a comparative study, Energy and Buildings 43 (10) (2011) 2893–2899

Hammer, A., Kühnert, J., Weinreich, K., & Lorenz, E. (2015). Short-term forecasting of surface solar irradiance based on meteosat-SEVIRI data using a nighttime cloud index. Remote Sensing, 7(7), 9070-9090.

Hand, D. J., Mannila, H., & Smyth, P. (2001). Principles of data mining (adaptive computation and machine learning) (pp. 361-452). Cambridge, MA: MIT press.
Hands-On Machine Learning with Scikit-Learn and TensorFlow. (2018). [Figure]. O'Reilly |

Hassan, M. A., Khalil, A., Kaseb, S., & Kassem, M. A. (2017). Potential of four different machine-learning algorithms in modeling daily global solar radiation. Renewable Energy, 111, 52-62.

Heinle, A., Macke, A., & Srivastav, A. (2010). Automatic cloud classification of whole sky images. Atmospheric Measurement Techniques, 3(3), 557-567.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

Huang, G. B., Zhou, H., Ding, X., & Zhang, R. (2012). Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 42(2), 513-529.

Huang, H., Yoo, S., Yu, D., Huang, D., & Qin, H. (2012, August). Correlation and local feature based cloud motion estimation. In Proceedings of the Twelfth International Workshop on Multimedia Data Mining (pp. 1-9). ACM.

Haykin, S., & Network, N. (2004). A comprehensive foundation. Neural networks, 2(2004), 41.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

Heaton, J. (2013). Feedforward backpropagation neural networks. Introduction to Neural Networks for Java.

Heidinger, A. K., Evan, A. T., Foster, M. J., & Walther, A. (2012). A naive Bayesian cloud-detection scheme derived from CALIPSO and applied within PATMOS-x. Journal of Applied Meteorology and Climatology, 51(6), 1129-1144

Hochreiter, S., Bengio, Y., Frasconi, P., & Schmidhuber, J. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Hocking, R. R. (1976). A biometrics invited paper. the analysis and selection of variables in linear regression. Biometrics 32(1): pp. 1–49.

Hong, T., Pinson, P., & Fan, S. (2014). Global energy forecasting competition 2012.

Hossain, M. R., Oo, A. M. T., & Ali, A. B. M. S. (2013). The combined effect of applying feature selection and parameter optimization on machine learning techniques for solar Power prediction. American Journal of Energy Research, 1(1), 7-16.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). A practical guide to support vector classification.

Hubel, D. and Wiesel, T. (1968). Receptive fields and functional architecture of monkey striate cortex. Journal of Physiology (London), 195, 215–243.

Ibrahim, S., Daut, I., Irwan, Y. M., Irwanto, M., Gomesh, N., & Farhana, Z. (2012). Linear regression model in estimating solar radiation in Perlis. Energy Procedia, 18, 1402-1412.

Inman, R. H., Pedro, H. T., & Coimbra, C. F. (2013). Solar forecasting methods for renewable energy integration. Progress in energy and combustion science, 39(6), 535-576.

Jade, A. M., Srikanth, B., Jayaraman, V. K., Kulkarni, B. D., Jog, J. P., & Priya, L. (2003). Feature extraction and denoising using kernel PCA. Chemical Engineering Science, 58(19), 4441-4448.

Jaeger, H. (2002). Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach (Vol. 5). Bonn: GMD-Forschungszentrum Informationstechnik.

Jiménez-Pérez, P. F., & Mora-López, L. (2016). Modeling and forecasting hourly global solar radiation using clustering and classification techniques. Solar Energy, 135, 682-691.

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260.

Jozefowicz, R., Zaremba, W., & Sutskever, I. (2015, June). An empirical exploration of recurrent network architectures. In International Conference on Machine Learning (pp. 2342-2350).K. Li,

Kashyap, Y., Bansal, A., & Sao, A. K. (2015). Solar radiation forecasting with multiple parameters neural networks. Renewable and Sustainable Energy Reviews, 49, 825-835.

Kerber, R. (1992, July). Chimerge: Discretization of numeric attributes. In Proceedings of the tenth national conference on Artificial intelligence (pp. 123-128). Aaai Press.

Khatib, T., Mohamed, A., Sopian, K., & Mahmoud, M. (2012). Assessment of artificial neural networks for hourly solar radiation prediction. International journal of Photoenergy, 2012.

Kim, G., Park, J., & Ryou, J. (2018). A Study on Utilization of Blockchain for Electricity Trading in Microgrid. In Big Data and Smart Computing (BigComp), 2018 IEEE International Conference on (pp. 743-746). IEEE.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kira, K., & Rendell, L. A. (1992). A practical approach to feature selection. In Machine Learning Proceedings 1992 (pp. 249-256).

Kohavi, R. (1995, August). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Ijcai(Vol. 14, No. 2, pp. 1137-1145).

Koprinska, I., Rana, M., & Agelidis, V. G. (2015). Correlation and instance based feature selection for electricity load forecasting. Knowledge-Based Systems, 82, 29-40.

Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160, 3-24.

Koza, J.R. (1992). Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT press

Krause, B., Lu, L., Murray, I., & Renals, S. (2016). Multiplicative LSTM for sequence modelling. arXiv preprint arXiv:1609.07959.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

Iacono, M. J., Delamere, J. S., Mlawer, E. J., Shephard, M. W., Clough, S. A., & Collins, W. D. (2008). Radiative forcing by long-lived greenhouse gases: Calculations with the AER radiative transfer models. Journal of Geophysical Research: Atmospheres, 113(D13).

Lalonde, J. F., Efros, A. A., & Narasimhan, S. G. (2012). Estimating the natural illumination conditions from a single outdoor image. International Journal of Computer Vision, 98(2), 123-145.

Langley, P. (1988). Machine Learning as an Experimental Science. Machine Learning, 3(1), 5-8.

Langley, P. (1996). Elements of Machine Learning. Morgan Kaufmann.

Lauret, P., Voyant, C., Soubdhan, T., David, M., & Poggi, P. (2015). A benchmarking of machine learning techniques for solar radiation forecasting in an insular context. Solar Energy, 112, 446-457.

Lawrence, R. L., & Wright, A. (2001). Rule-based classification systems using classification and regression tree (CART) analysis. Photogrammetric engineering and remote sensing, 67(10), 1137-1142.

Learning Curves for Machine Learning. Dataquest. (2018). Retrieved 21 June 2018, from https://www.dataquest.io/blog/learning-curves-machine-learning/

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436.

Lee, C. Y., Gallagher, P. W., & Tu, Z. (2016, May). Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In Artificial Intelligence and Statistics (pp. 464-472).

Le Pivert X, Sicot L, Merten J. (2009) A tool for the 24 hours forecast of photovoltaic production. In: Proceedings of the 24th European Photovoltaic Solar Energy Conference. Hamburg. p. 21–25

Lewis CD. (1982). International and business forecasting methods

Linares, L., Erickson, R. W., MacAlpine, S., & Brandemuehl, M. (2009, February). Improved energy capture in series string photovoltaics via smart distributed power electronics. In Applied Power Electronics Conference and Exposition, 2009. APEC 2009. Twenty-Fourth Annual IEEE (pp. 904-910). IEEE.

Liu, H., Motoda, H., Setiono, R., & Zhao, Z. (2010, May). Feature selection: An ever evolving frontier in data mining. In Feature Selection in Data Mining (pp. 4-13).

Lorenz E, Remund J, Müller SC, Traunmüller W, Steinmaurer G, Pozo D, et al. (2009) Benchmarking of different approaches to forecast solar irradiance. In: 24th European photovoltaic solar energy conference.

Lowd, D., & Domingos, P. (2005, August). Naive Bayes models for probability estimation. In Proceedings of the 22nd international conference on Machine learning (pp. 529-536). ACM.

Lv, K., Jiang, S., & Li, J. (2017). Learning Gradient Descent: Better Generalization and Longer Horizons. arXiv preprint arXiv:1703.03633.

M. Sengupta, L. Wald, S. Wilbert,Direct normal irradiance related definitions and applications: The circumsolar issue, Solar Energy,Volume 110,2014,Pages 561-577,ISSN 0038-092X, https://doi.org/10.1016/j.solener.2014.10.001.

Malhi, A., & Gao, R. X. (2004). PCA-based feature selection scheme for machine defect classification. IEEE Transactions on Instrumentation and Measurement, 53(6), 1517-1525.

Marquez, R., Pedro, H. T., & Coimbra, C. F. (2013). Hybrid solar forecasting method uses satellite imaging and ground telemetry as inputs to ANNs. Solar Energy, 92, 176-188.

Martínez-Chico, M., Batlles, F. J., & Bosch, J. L. (2011). Cloud classification in a mediterranean location using radiation data and sky images. Energy, 36(7), 4055-4062.

Mathiesen, P., & Kleissl, J. (2011). Evaluation of numerical weather prediction for intra-day solar forecasting in the continental United States. Solar Energy, 85(5), 967-977.

Maxwell, E. L. (1998). METSTAT—The solar radiation model used in the production of the National Solar Radiation Data Base (NSRDB). Solar Energy, 62(4), 263-279.

McLeod, R., & Schell, G. P. (2007). Management information systems. USA: Pearson/Prentice Hall.

Medsker, L. R., & Jain, L. C. (2001). Recurrent neural networks. Design and Applications, 5.

Mejia, F. A., & Kleissl, J. (2013). Soiling losses for solar photovoltaic systems in California. Solar Energy, 95, 357-363.

Mellit, A., Kalogirou, S. A., Shaari, S., Salhi, H., & Arab, A. H. (2008). Methodology for predicting sequences of mean monthly clearness index and daily solar radiation data in remote areas: Application for sizing a stand-alone PV system. Renewable Energy, 33(7), 1570-1590.

Mengelkamp, E., Gärttner, J., Rock, K., Kessler, S., Orsini, L., & Weinhardt, C. (2018). Designing microgrid energy markets: A case study: The Brooklyn Microgrid. Applied Energy, 210, 870-880.

McCandless, T. C., Haupt, S. E., & Young, G. S. (2015). A model tree approach to forecasting solar irradiance variability. Solar Energy, 120, 514-524.

Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (Eds.). (2013). Machine learning: An artificial intelligence approach. Springer Science & Business Media.

Mjolsness, E., & DeCoste, D. (2001). Machine learning for science: state of the art and future prospects. science, 293(5537), 2051-2055.

Mohandes, M. A., Halawani, T. O., Rehman, S., & Hussain, A. A. (2004). Support vector machines for wind speed prediction. Renewable Energy, 29(6), 939-947.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). Introduction to linear regression analysis (Vol. 821). John Wiley & Sons.

Mueller RW, Matsoukas C, Behr HD, Gratzki A, Hollmann R (2009) The CMSAF operational scheme for the satellite based retrieval of solar surface irradiance—a LUT based eigenvector hybrid approach. Remote Sens Environ 113:1012–1024

Murphy, K. P. (2006). Naive Bayes classifiers. University of British Columbia, 18.

Myers, D. (2006). Cloudy Sky Version of Bird's Broadband Hourly Clear Sky Model (Presentation) (No. NREL/PR-581-40115). National Renewable Energy Lab.(NREL), Golden, CO (United States).

Naive Bayes - machine learning algorithm for classification problems. (2017). [Figure] TechLeer. Retrieved 18 June 2018, from https://www.techleer.com/articles/200-naive-bayes-machine-learning-algorithm-for-classification-problems/

Nasrabadi, N. M. (2007). Pattern recognition and machine learning. Journal of electronic imaging, 16(4), 049901.

Neter, J., Kutner, M. H., Nachtsheim, C. J., & Wasserman, W. (1996). Applied linear statistical models (Vol. 4, p. 318). Chicago: Irwin.

Nguyen, T. T., & Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. IEEE Communications Surveys & Tutorials, 10(4), 56-76.

NOAA's National Weather Service - Glossary. (2018). Forecast.weather.gov. Retrieved 24 June 2018, from https://forecast.weather.gov/glossary.php?word=sky%20condition

Nolfi, S., Parisi, D., & Elman, J. L. (1994). Learning and evolution in neural networks. Adaptive Behavior, 3(1), 5-28.

Norman, D. A. (1999). Affordance, conventions, and design. interactions, 6(3), 38-43.

Nrel.gov. (2015). Retrieved 10 March 2018, from https://www.nrel.gov/docs/fy12osti/54601.pdf

O'Leary, D., & Kubby, J. (2017). Feature Selection and ANN Solar Power Prediction. Journal of Renewable Energy, 2017.

Olatomiwa, L., Mekhilef, S., Shamshirband, S., Mohammadi, K., Petković, D., & Sudheer, C. (2015). A support vector machine–firefly algorithm-based model for global solar radiation prediction. Solar Energy, 115, 632-644.

Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on (pp. 1717-1724). IEEE.

P. Blanc, B. Espinar, N. Geuder, C. Gueymard, R. Meyer, R. Pitz-Paal, B. Reinhardt, D. Renné,

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up: sentiment classification using machine learning techniques. In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10 (pp. 79-86). Association for Computational Linguistics.

Papageorgiou, C., & Poggio, T. (2000). A trainable system for object detection. International Journal of Computer Vision, 38(1), 15-33.

Pearl, J. (2014). Probabilistic reasoning in intelligent systems: networks of plausible inference. Elsevier.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas,

Peng, Z., Yu, D., Huang, D., Heiser, J., & Kalb, P. (2016). A hybrid approach to estimate the complex motions of clouds in sky images. Solar Energy, 138, 10-25.

PEP 8 -- Style Guide for Python Code. (2018). Python.org. Retrieved 22 June 2018, from https://www.python.org/dev/peps/pep-0008/?

Performance, P., & PoA, P. (2018). [Figure] PV System Performance: GHI to PoA - EcoSmart™ Solar. EcoSmart™ Solar. Retrieved 26 June 2018, from https://ecosmartsun.com/pv-system-performance-3/pv-system-performance-ghi-to-poa/

Perez, R. K. Moore and P. Stackhouse. (2006): Independent Validation of NDFD-Based Solar Radiation Forecasts. Proc. ASES Annual Conference, Denver, CO

Perez, R., Moore, K., Wilcox, S., Renné, D., & Zelenka, A. (2007). Forecasting solar radiation–Preliminary evaluation of an approach based upon the national forecast database. Solar Energy, 81(6), 809-812.

Perez, R., Kivalov, S., Schlemmer, J., Hemker Jr., K., Renne´, D., Hoff, T., (2010). Validation of short and medium term operational solar radiation forecasts in the US. Solar Energy 84 (12), 2161–2172

Perez, R., E. Lorenz, S. Pelland, M. Beauharnois, G. Van Knowe, K. Hemker, D. Heinemann, J. Remund, S. Mu¨ller,W. Traunmuller, G. Steinmauer g, D. Pozo, J. Ruiz-Arias,V. Lara-Fanego, .L. Ramirez-Santigosa, M. Gaston-Romero, and L. Pomares. (2013). Comparison of numerical weather prediction solar irradiance forecasts in the U.S., Canada and Europe. Solar Energy, 94: 305–326.

Phippen, A., Sheppard, L., & Furnell, S. (2004). A practical evaluation of Web analytics. Internet Research, 14(4), 284-293.

Pieroni, A., Scarpato, N., Di Nunzio, L., Fallucchi, F., & Raso, M. (2018). Smarter City: Smart Energy Grid based on Blockchain Technology. International Journal on Advanced Science, Engineering and Information Technology, 8(1), 298-306.

Piri, J., Shamshirband, S., Petković, D., Tong, C. W., & ur Rehman, M. H. (2015). Prediction of the solar radiation on the Earth using support vector regression technique. Infrared Physics & Technology, 68, 179-185.

Polotan, M. Supervised Learning (2015). [Figure] Morganpolotan.wordpress.com. Retrieved 2 June 2018, from https://morganpolotan.wordpress.com/tag/supervised-learning/

Poudel, P., & Jang, B. (2017). Solar Power Prediction Using Deep Learning.

Power, D. J. (2002). Decision support systems: concepts and resources for managers. Greenwood Publishing Group.

PV Performance Modeling Collaborative. Sun Position. [Figure] (2018). Pvpmc.sandia.gov. Retrieved 10 June 2018, from https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/sun-position/

Quinlan, J. R. (1987). Simplifying decision trees. International journal of man-machine studies, 27(3), 221-234.

R. Rogers, M.K. Yau, A Short Course in Cloud Physics, third ed., Pergamon Press, Elmsford, NY, 1989.

Rajagopalan, B., & Lall, U. (1999). A k-nearest-neighbor simulator for daily precipitation and other weather variables. Water resources research, 35(10), 3089-3101.

Ram, J. P., Babu, T. S., & Rajasekar, N. (2017). A comprehensive review on solar PV maximum power point tracking techniques. Renewable and Sustainable Energy Reviews, 67, 826-847.

Ramedani, Z., Omid, M., Keyhani, A., Shamshirband, S., Khoshnevisan, B., 2014. Potential of radial basis function based support vector regression for global solar radiation prediction. Renew. Sustain. Energy Rev. 39 (1), 1005–1011.

Rasmussen, C. E. (2004). Gaussian processes in machine learning. In Advanced lectures on machine learning (pp. 63-71). Springer, Berlin, Heidelberg.

Raschka, S. (2018). [Figure] Machine Learning FAQ. Retrieved 19 June 2018, from https://sebastianraschka.com/faq/docs/visual-backpropagation.html

Rauch, M. (2011, October). Mobile documentation: Usability guidelines, and considerations for providing documentation on Kindle, tablets, and smartphones. In Professional Communication Conference (IPCC), 2011 IEEE International (pp. 1-13). IEEE.

Regression. (2018). [Figure]. What Measures to Look at to Determine Overfitting in Linear Regression. Cross Validated. Retrieved 1 June 2018, from https://stats.stackexchange.com/questions/192007/what-measures-you-look-at-the-determine-over-fitting-in-linear-regression

Reikard, G. (2009). Predicting solar radiation at high resolutions: A comparison of time series forecasts. Solar Energy, 83(3), 342-349.

Revolvy, L. (2018). [Figure] "Decision tree" on Revolvy.com. Revolvy.com. Retrieved 20 June 2018, from https://www.revolvy.com/main/index.php?s=Decision+tree

Rish, I. (2001, August). An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41-46). IBM.

Robert, C. (2014). Machine learning, a probabilistic perspective.

Romero, A., Gatta, C., & Camps-Valls, G. (2016). Unsupervised deep feature extraction for remote sensing image classification. IEEE Transactions on Geoscience and Remote Sensing, 54(3), 1349-1362.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

Safari. Retrieved 18 June 2018, from https://www.oreilly.com/library/view/hands-on-machine-learning/9781491962282/ch04.html

Sahami, M. (1996, August). Learning Limited Dependence Bayesian Classifiers. In KDD (Vol. 96, No. 1, pp. 335-338).

Schalkoff, R. J. (1997). Artificial neural networks (Vol. 1). New York: McGraw-Hill.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural networks, 61, 85-117

Scikit-learn: Machine learning in Python. (2011). Journal of machine learning research, 12(Oct), 2825-2830.

Sengupta, M., & Gotseff, P. (2013). Evaluation of clear sky models for satellite-based irradiance estimates (No. NREL/TP-5D00-60735). National Renewable Energy Lab.(NREL), Golden, CO (United States).

Sharma, N., Sharma, P., Irwin, D., & Shenoy, P. (2011, October). Predicting solar generation from weather forecasts using machine learning. In Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on (pp. 528-533). IEEE.

Schapire, R. E. (2003). The boosting approach to machine learning: An overview. In Nonlinear estimation and classification (pp. 149-171). Springer, New York, NY.

Shi, C., Wang, C., Wang, Y., & Xiao, B. (2017). Deep Convolutional Activations-Based Features for Ground-Based Cloud Classification. IEEE Geoscience and Remote Sensing Letters, 14(6), 816-820.

Shi, J., Lee, W. J., Liu, Y., Yang, Y., & Wang, P. (2012). Forecasting power output of photovoltaic systems based on weather classification and support vector machines. IEEE Transactions on Industry Applications, 48(3), 1064-1069.

Silverman, B. W. (2018). Density estimation for statistics and data analysis. Routledge.

Singh, M., & Glennen, M. (2005). Automated ground-based cloud recognition. Pattern analysis and applications, 8(3), 258-271.

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. Statistics and computing, 14(3), 199-222.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In Advances in neural information processing systems (pp. 2951-2959).

Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006, December). Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In Australasian joint conference on artificial intelligence (pp. 1015-1021). Springer, Berlin, Heidelberg.

Solar Radiation | National Centers for Environmental Information (NCEI) formerly known as National Climatic Data Center (NCDC). (2018). Ncdc.noaa.gov. Retrieved 18 March 2018, from https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/solar-radiation

Specht, D. F. (1991). A general regression neural network. IEEE transactions on neural networks, 2(6), 568-576.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958.

Srivastava, S., & Lessmann, S. (2018). A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data. Solar Energy, 162, 232-247.

Sun, Y., Szűcs, G., & Brandt, A. R. (2018). Solar PV output prediction from video streams using convolutional neural networks. Energy & Environmental Science.

Sun, H., Gui, D., Yan, B., Liu, Y., Liao, W., Zhu, Y., ... & Zhao, N. (2016). Assessing the potential of random forest method for estimating solar radiation using air pollution index. Energy Conversion and Management, 119, 121-129.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).

Suykens, J. A., & Vandewalle, J. (1999). Least squares support vector machine classifiers. Neural processing letters, 9(3), 293-300.

Tan, A. C., & Gilbert, D. (2003). Ensemble machine learning on gene expression data for cancer classification.

Tapakis, R., & Charalambides, A. G. (2013). Equipment and methodologies for cloud detection and classification: A review. Solar Energy, 95, 392-430.

Taylor, R. (1990). Interpretation of the correlation coefficient: a basic review. Journal of diagnostic medical sonography, 6(1), 35-39.

The Horizon Line Changes with the Observers Height. (2018). [Figure] YouTube. Retrieved 23 June 2018, from https://www.youtube.com/watch?v=QpocuhmJ4VM

The Journey of a Machine Learning model from Building to Retraining. (2018). Towards Data Science. Retrieved 24 June 2018, from https://towardsdatascience.com/the-journey-of-a-machine-learning-model-from-building-to-retraining-fe3a37c32307

The Machine Learning Process. (2016). [Figure] A load of stuff about Azure IaaS and Azure IoT. Retrieved 19 June 2018, from https://blogs.msdn.microsoft.com/david/2016/11/11/the-machine-learning-process/

Toğrul, I. T., & Onat, E. (1999). A study for estimating solar radiation in Elaziğ using geographical and meteorological data. Energy Conversion and Management, 40(14), 1577-1584.

Torralba, A., Murphy, K. P., & Freeman, W. T. (2004, June). Sharing features: efficient boosting procedures for multiclass object detection. In Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on (Vol. 2, pp. II-II). IEEE.

Towards Data Science. (2017). [Figure] Towards Data Science. Retrieved 1 June 2018, from https://towardsdatascience.com/linear-regression-in-python-9a1f5f000606

Tsai, G. (2010). Histogram of oriented gradients. University of Michigan, 1(1), 1-17.

Tso, G. K., & Yau, K. K. (2007). Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. Energy, 32(9), 1761-1768.

Turban, E., King, D., Sharda, R., & Delen, D. (2013). Business intelligence: a managerial perspective on analytics. Prentice Hall, New York.

Turrado, C. C., López, M. D. C. M., Lasheras, F. S., Gómez, B. A. R., Rollé, J. L. C., & Juez, F. J. D. C. (2014). Missing data imputation of solar radiation data under different atmospheric conditions. Sensors, 14(11), 20382-20399.

Types of Web Services SOAP, XML-RPC and Restful - Phpflow.com. (2012). [Figure] Phpflow.com. Retrieved 30 June 2018, from https://www.phpflow.com/php/web-service-types-soapxml-rpcrestful/

Van der Walt, Stéfan & Schönberger, Johannes & Nunez-Iglesias, Juan & Boulogne, François & Warner, Joshua & Yager, Neil & Gouillart, Emmanuelle & Yu, Tony & scikit-image contributors, the. (2014). scikit-image: Image processing in Python. PeerJ. 2. 10.7717/peerj.453.

Verzijlbergh, R. A., Heijnen, P. W., de Roode, S. R., Los, A., & Jonker, H. J. (2015). Improved model output statistics of numerical weather prediction based irradiance forecasts for solar power applications. Solar Energy, 118, 634-645.

Wallach, D., & Goffinet, B. (1989). Mean squared error of prediction as a criterion for evaluating and comparing system models. Ecological modelling, 44(3-4), 299-306.

Wang, Z., & Bovik, A. C. (2009). Mean squared error: Love it or leave it? A new look at signal fidelity measures. IEEE signal processing magazine, 26(1), 98-117.

Wang, F., Zhen, Z., Mi, Z., Sun, H., Su, S., & Yang, G. (2015). Solar irradiance feature extraction and support vector machines based weather status pattern recognition model for short-term photovoltaic power forecasting. Energy and Buildings, 86, 427-438.

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate research, 30(1), 79-82.

Willson, R. C., Gulkis, S., Janssen, M., Hudson, H. S., & Chapman, G. (1981). Observations of solar irradiance variability. Science, 211(4483), 700-702.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.

Xie, Y., Sengupta, M., & Dudhia, J. (2016). A Fast All-sky Radiation Model for Solar applications (FARMS): Algorithm and performance evaluation. Solar Energy, 135, 435-445.

Xingjian, S. H. I., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., & Woo, W. C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In Advances in neural information processing systems (pp. 802-810).

Yadav, A. K., Malik, H., & Chandel, S. S. (2014). Selection of most relevant input parameters using WEKA for artificial neural network based solar radiation prediction models. Renewable and Sustainable Energy Reviews, 31, 509-519.

Yohanna, J.K., Itodo, I.N., Umogbai, V.I., 2011. A model for determining the global solar radiation for Makurdi, Nigeria. Renew. Energy 36, 1989–1992.

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015). Understanding neural networks through deep visualization. arXiv preprint arXiv:1506.06579.

Yu, L., & Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. Journal of machine learning research, 5(Oct), 1205-1224.

Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham.

Zeng, J., & Qiao, W. (2013). Short-term solar power prediction using a support vector machine. Renewable Energy, 52, 118-127.

Zurada, J. M. (1992). Introduction to artificial neural systems (Vol. 8). St. Paul: West.