Towards Broad, Low-Cost Solar Radiation Forecasting using Machine Learning

Nikolaos Chantampakis

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Electrical and Computer Engineer

at the
Technical University of Crete
March, 2025

Committee:
Professor Georgios Chalkiadakis
Associate Professor Vasilis Samoladas
Associate Professor Athanasios Panagopoulos

Abstract

In the past few years, solar radiation prediction has been paramount in a multitude of sectors, from energy production via renewable energy sources, to tracking climate change, among others. So far, work in the area lacks in large area coverage, ease of access, or uses past solar radiation readings, relying on related equipment being already on-site. In this work, we provide insight into the efficacy of neural networks in the area, accompanied with data sourced from varying providers. In order to achieve this, we create and vet a weather reading dataset from a large variety of stations, which are more indicative of what smaller organizations or individuals may have access to, instead of more tailored datasets. We utilize this dataset to train a number of neural networks, each with different architectures, and evaluate their results so as to set a standard to be improved upon in later work utilizing a similar type of dataset. The results indicate that, even utilizing a much broader dataset than what has been used in the past, neural networks show promise in this area, especially with more targeted implementations.

Περίληψη

Τα τελευταία χρόνια, η πρόβλεψη ηλιακής ακτινοβολίας έχει γίνει αναπόσπαστο κομμάτι πλήθους τομέων, όπως, μεταξύ άλλων, η παραγωγή ενέργειας μέσω ανανεώσιμων πηγών και η παρακολούθηση της κλιματικής αλλαγής. Ως τώρα, οι σχετικές εργασίες γύρω από το θέμα είτε καλύπτουν πολύ μικρό χώρο, είτε είναι δυσπρόσιτες/μη προσβάσιμες από τον περισσότερο κόσμο, είτε χρησιμοποιούν χρονοσειρές μετρήσεων ηλιακής ακτινοβολίας ως εισόδους, βασιζόμενες στην ύπαρξη σχετικού εξοπλισμού στο σημείο-στόχο. Στην παρούσα διπλωματική εργασία, εξετάζουμε την αποδοτικότητα των νευρωνικών δικτύων εκπαιδευμένων με δεδομένα από διάφορες πηγές. Συγκεκριμένα, δημιουργούμε και ελέγχουμε ένα σύνολο μετρήσεων καιρικών συνθηκών από μεγάλο πλήθος σταθμών, ως πιο ενδεικτικό του τύπου δεδομένων στα οποία μπορούν να έχουν πρόσβαση μικρότερες οργανώσεις ή επιμέρους άτομα. Προτεραιοποιούμε τη χρήση τέτοιων μετρήσεων έναντι της χρήσης πιο προσαρμοσμένων δεδομένων. Αξιοποιούμε το σύνολο δεδομένων μας για να εκπαιδεύσουμε ένα πλήθος νευρωνικών δικτύων με διαφορετικές αρχιτεκτονικές, και αξιολογούμε τα αποτελέσματά τους ώστε να θέσουμε ένα πρότυπο προς βελτίωση σε μελλοντικές εργασίες που θα χρησιμοποιούν ένα παρόμοιο σύνολο δεδομένων. Τα αποτελέσματα δείχνουν ότι ακόμη και με ένα πολύ μεγαλύτερο/ευρύτερο σύνολο δεδομένων από όσα έχουν χρησιμοποιηθεί ως τώρα, συγκεκριμένες στοχευμένες υλοποιήσεις νευρωνικών δικτύων μπορεί να είναι αποτελεσματικές στο συγκεκριμένο πρόβλημα.

Acknowledgements

To start, I have to express my thanks to Dr. Georgios Chalkiadakis as well as Dr. Athanasios Panagopoulos for entrusting me with expanding a major undertaking of theirs in the past, being very patient with my delays, and in particular Dr. Panagopoulos for offering invaluable assistance in figuring out a sector I was not particularly familiar with, that of solar radiation, as well as generally being able to nudge me in the right direction whenever I would encounter an obstacle.

I would be remiss if I did not also mention my gratitude to my parents, who supported me in whatever way the could throughout my study at the Technical University of Crete, as well as the university itself for providing me and other students like me with access to such high quality education in our field.

Finally, a massive thanks to my friends, for providing a place and a group in which I could relax, voice my frustrations, and take my mind off things whenever I needed. Without them, this project would have probably taken even longer than it already has.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Accurate prediction of global horizontal irradiance (GHI) is becoming increasingly important, as it can support numerous applications, from intelligent agriculture via irrigation scheduling, to power output prediction for photovoltaic systems [1], or fire hazard forecasting [2]. Ideally, a GHI prediction method would be applicable to a large part (if not most of) the world, would utilize cheap and easily accessible data, such as publicly available weather forecasts, and, of course, be highly accurate. In this context, machine learning, particularly neural networks (NNs), offer a promising solution due to their ability to transform cheap, bulk or limited data to much more useful predictions.

Works attempting to utilize NNs for predicting GHI often rely on area-limited ground-based observations or satellite data, such as the work of [3], [4], [5] and [6], which end up being sparse and lacking in spatial or temporal resolution.

The ground work for our approach has been laid in the work of RENES [7], which implemented a basic MLP along with traditional methods of estimating GHI, and used data from publicly available (at the time) sources for testing, and our work follows the paradigm that was set by said work. However, RENES does not offer wide area coverage either, its NN architecture is very outdated by this point in time, and the clear sky model utilized seems to not be accurate, as we show later in this work.

This thesis develops a NN model to predict all-sky GHI, mimicking the functionality of a pyranometer, on a broad scale, spanning the entirety of Europe as well as North America, along with few points in other continents. It accomplishes this by utilizing data provided by Visual Crossing[1], a comprehensive weather data provider that has been able to offer a rich dataset of historical data based on live readings, spanning a reasonably large percentage of the globe. Simultaneously, the cost of developing such a dataset is kept low by only requiring access to any regular weather data providers, instead of more niche/expensive access to specialized institutions. By utilizing this rich dataset, the NN aims to provide accurate predictions of GHI, overcoming the limitations of traditional methods.

Firstly, the thesis investigates the sanitization requirements of a

[1] www.visualcrossing.com

large-scale historical weather dataset that is provided by thousands of different entities, with differing equipment, as well as the preprocessing techniques necessary to transform raw weather data into suitable inputs for the NN. Secondly, we design a robust NN architecture capable of handling the vast and diverse data inputs required for large scale GHI prediction. Thirdly, we evaluate the model's performance against previous work in the field, with commentary concerning the differing configurations between this work and the rest. By providing a solid, easily accessible standard for GHI prediction, further study may be encouraged into the matter.

The results of this work, aside from its contribution by creating the aforementioned dataset, indicate that neural networks show promise for broader area, all-sky solar radiation prediction. In fact, we show that older, extremely simple NN architectures are not suited for handling this volume of data, and that deeper but still relatively simple NN architectures perform better, but have a sort of "ceiling", which may be improved upon by other methods, discussed later. In more detail, the contributions of this work are the following:

- Devised a sanitization procedure for quality control and refinement of large scale solar radiation data
- Compared clear sky models and identified the most accurate one

(with higher face validity)

- Created broad weather reading dataset suitable for training solar radiation prediction models that use openly available weather forecasting reports as inputs
- Investigated and analyzed the effect of different weather elements in solar radiation showing that humidity and temperature have the highest influence while timestamps and otherwise more useful climate classifications have the lowest
- Tested multiple standard NN architectures trained on the aforementioned dataset, showing that: 1) More complex/bespoke NN architectures can benefit from larger number of features 2) there is improvement in comparison to older work, but there is a "ceiling" for run-of-the-mill MLPs

Chapter 2

Related Work

In this chapter, we present work that is more closely related to ours, in a limitation-centric way. Quite a few publications have attempted this undertaking, all with different implementations, limitations and results.

The most common factor that seems to be missing is wide area coverage. It is usual for work in the field utilizing NNs (e.g. [5], [6], [8], [3]) to rely on very few, concentrated locations, from a single one in [5] and [6] to 12, all within Turkey, in [3]. This seems to be a byproduct of these works using more curated sources for their data, and not relying on commercially available data, such as what may be provided from a weather data provider with access to a large amount of stations, which may be more akin to big data processing.

One work that attempted to cover a larger area was RENES [7], whose data, although sourced from 10 locations, spanned the entire

Mediterranean belt, along with a single point in Copenhagen, Denmark. This was accomplished by using data that was, at the time, openly available to anyone, and sourced from smaller scale stations, possibly even individuals like hobbyists. One final note regarding data usage is the fact that some works, such as [3] and [4], may not even use weather data, but instead use satellite image data, estimating weather conditions via imagery.

Another area in which there are huge differences regarding this topic is the implementation of NNs. Works like RENES [7] and [8] implement very simple artificial NNs in the form of a standard, modern MLP, while other publications, attempt to craft more specific networks to fit the problem and data provided, like implementing a (Convolutional) Long Short-Term Memory network to maximize the effectiveness of their GHI time series data [6], using an evolutionary algorithm to train a standard ANN [4], or other, NN-adjacent models, such as an adaptive network-based fuzzy inference system (ANFIS) [5].

As far as more different approaches to the entire problem go, instead of attempting to predict a specific amount of mean solar radiation expected in the span of an hour (point prediction), one may try to predict a range of values between which the true value can be, under a specific confidence level (probabilistic prediction). This style of prediction is less suited towards NNs, but there have been attempts

to achieve this with other methods, such as utilizing tree boosting via the XGBoost algorithm described in [9], which was implemented in [10], using data sourced from NREL.

Although outside the scope of this work, we would also be remiss not to mention other types of NNs that have been introduced, but not yet used in the field, and seem to be a great match for our exact problem. More specifically, we refer to the recently published work regarding Kolmogorov-Arnold Networks (KANs) [11], in which the networks described claim to offer what our work is looking for - a network structure that works well when attempting to simulate more complex functions, such as predicting all-sky radiation, while handling large amount of data well. Another type of network that claims to be well-suited for large-scale tabular data is TabNet [12], which has been used in the field for photovoltaic power output prediction [13], but it would be interesting to see its performance in a more generalized version of the problem.

Chapter 3

Dataset

In order to develop a neural network of such scale, acquiring a comprehensive and high-quality dataset is paramount and arises as a prominent challenge on its own. The tasks necessary to derive such a dataset include collecting the data, sanitizing it, and preprocessing it. The contents of a data sample that is later split into subsets and utilized to train our NNs can be seen in Table 3.1. The distribution of the stations where the samples were fetched from is shown in Figure 3.1.

3.1 Acquiring the Dataset

We utilized Visual Crossing's API to fetch hourly weather data from 2018 to 2023, for each city with a population more than 100,000, according to a table provided by Opendatasoft[1], which in itself has

[1] https://data.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-1000%40public/table

Table 3.1: Contents of an Hourly Sample

Element
Cloud cover, in %
Mean temperature, in °F
Relative humidity, in %
Wind speed, in miles per hour
AQI
Beam radiation, in W/m^2
Diffuse radiation, in W/m^2
Dew point, in °F
Simplified Solis clear sky estimation, in W/m^2
Latitude
Longitude
Köppen–Geiger climate classification
UNIX timestamp, in milliseconds
Visibility, in miles

been crowdsourced from GeoNames[2].

The data consists of an interpolated combination of hourly readings received from up to 3 observer stations in a 50 mile radius from the given coordinates, if not near a "major reporting station". A sample of an hourly reading, including its contents, along with a description, can be found in Table 3.2.

The final result of this retrieval was a large number of files, each titled as the set of coordinates used to fetch data from that location (if available), containing 6 years' worth of "unsanitized" hourly weather data per location.

Aside from the natural factors that are provided by the dataset, there are also man-made factors that may obstruct the atmosphere

[2] https://geonames.org

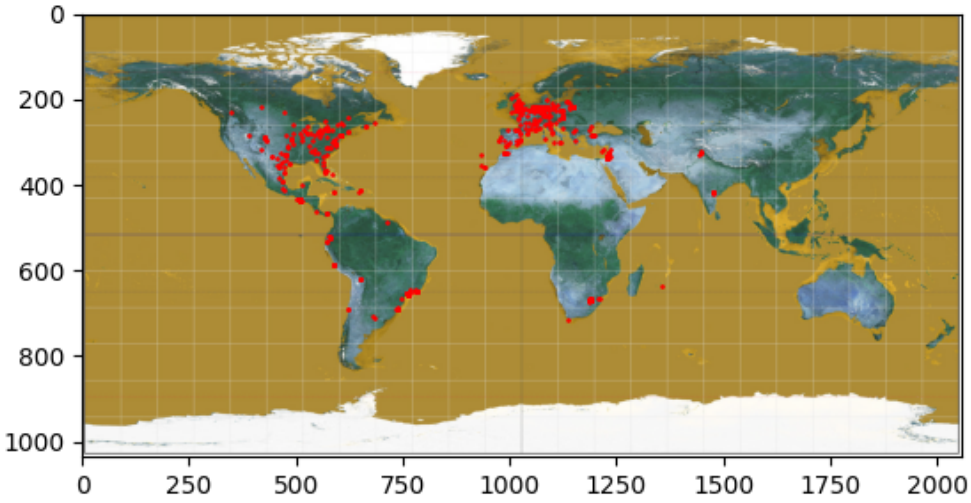


Figure 3.1: Distribution of stations in the 500 samples used

and as such reduce the amount of solar radiation reaching the ground. One of these factors that we can somewhat account for is air pollution, which has been shown to reduce the amount of radiation reaching the earth by a non-negligible percentage [14]. The amount of pollution in the air of a certain area can be inferred from EPA's index for reporting air quality, the Air Quality Index (AQI)[3].

Another factor that may be important in GHI prediction is climate classification, and it has been utilized in this field for a very

[3]Fetched using the Ozon3 Python library https://github.com/Ozon3Org/Ozon3

Table 3.2: Sample Hourly Observer Reading

Field	**Value**
wdir (wind direction, in degrees)	296
temp (mean temperature, in °F)	43.2
maxt (max temperature, in °F)	43.2
visibility (in miles)	19.4
wspd (wind speed, in miles per hour)	14.1
datetimeStr (ISO 8601 date and time string)	2019-01-04T13:00:00+01:00
solarenergy (in MJ/m^2)	0.1
heatindex (in °F)	null
cloudcover (in %)	100
mint (min temperature, in °F)	43.2
datetime (UNIX timestamp, in milliseconds)	1546606800000
precip (precipitation, in inches)	0
solarradiation (in W/m^2)	27
weathertype (description of notable weather conditions)	Sky Unchanged
snowdepth (snow currently on the ground, in inches)	0
sealevelpressure (in mbar)	1036.4
snow (snowfall for the given hour, in inches)	0
dew (dew point, in °F)	36.1
humidity (relative humidity, in %)	75.99
precipcover (precipitation coverage, in % of the hour)	null
wgust (wind gust, in miles per hour)	21.4
conditions (simpler text description of weather conditions)	Overcast
windchill (in °F)	36.2
info	null

long time, as seen in [15]. In our case, we can use the idea presented in Hottel's paper, and introduce this concept to the NN through the Köppen–Geiger climate classification. The Köppen–Geiger climate classification (KCC) is a system used to categorize the world's climates based on temperature and precipitation patterns, which was initially defined in [16] and later added to by R. Geiger. It divides climates into five main groups: tropical, dry, temperate, continental, and polar, with subcategories to further describe specific variations

in these regions. The library we used[4] utilizes an updated, more accurate version of the maps shown in the initial publication, proposed in [17] and [18]. We can then convert the unique strings provided by the library into unique integers, which can then be passed into and processed by our model.

Finally, one more easily accessible addition is a clear sky GHI estimate that may provide our NNs with a good starting-off point in order to integrate the rest of the elements en route to estimating the all-sky GHI. Initially, we intended to use the Pysolar[5] library, as a similar implementation to it had been utilized before in [7]. Both of these attempt to implement the clear-sky estimation described in [19]. However, after noticing some very irregular results in the early stages of testing our NNs, we decided to test the module's performance versus our actual readings, and the results of said testing are shown in Figure 3.2. Note that the values where the actual reading is much larger than the clear sky estimation have been cut off, assumed to be faulty. Further study of these results showed that Pysolar had a tendency to underestimate GHI values early in the day, mostly in the 7:00 - 10:00 timeslot, while getting back on track during peak sunlight hours, which explains the behavior displayed in the image.

For this reason, the clear-sky model we have opted to use is the

[4] https://github.com/cwru-sdle/kgcpy
[5] https://github.com/pingswept/pysolar

simplified Solis model, as mentioned in Chapter 2 and implemented in the pvlib Python library[6] [20]. Its general performance can be seen in Figure 3.3. We can clearly see a more expected distribution of values, with the majority of near-zero cloud cover readings congregating around the equality line, and moving farther away as cloud cover increases.

For comparison purposes, we also included clear sky beam and diffuse radiation estimations, as calculated in [7]. These were not used in the evaluation of our model, as the simplified Solis model provides more accurate estimations.

3.2 Quality Control

In any scientific analysis, the reliability of the data plays a crucial role in ensuring the validity of the results. This is particularly true in meteorological studies, where accurate weather readings are essential for forecasting, climate modeling, and environmental assessments. However, raw weather data often contains errors and inconsistencies due to various factors such as sensor malfunctions, human error during manual observations, or transmission issues. These inaccuracies can lead to significant distortions in analysis and conclusions if not properly addressed.

[6] https://pvlib-python.readthedocs.io/en/stable/index.html

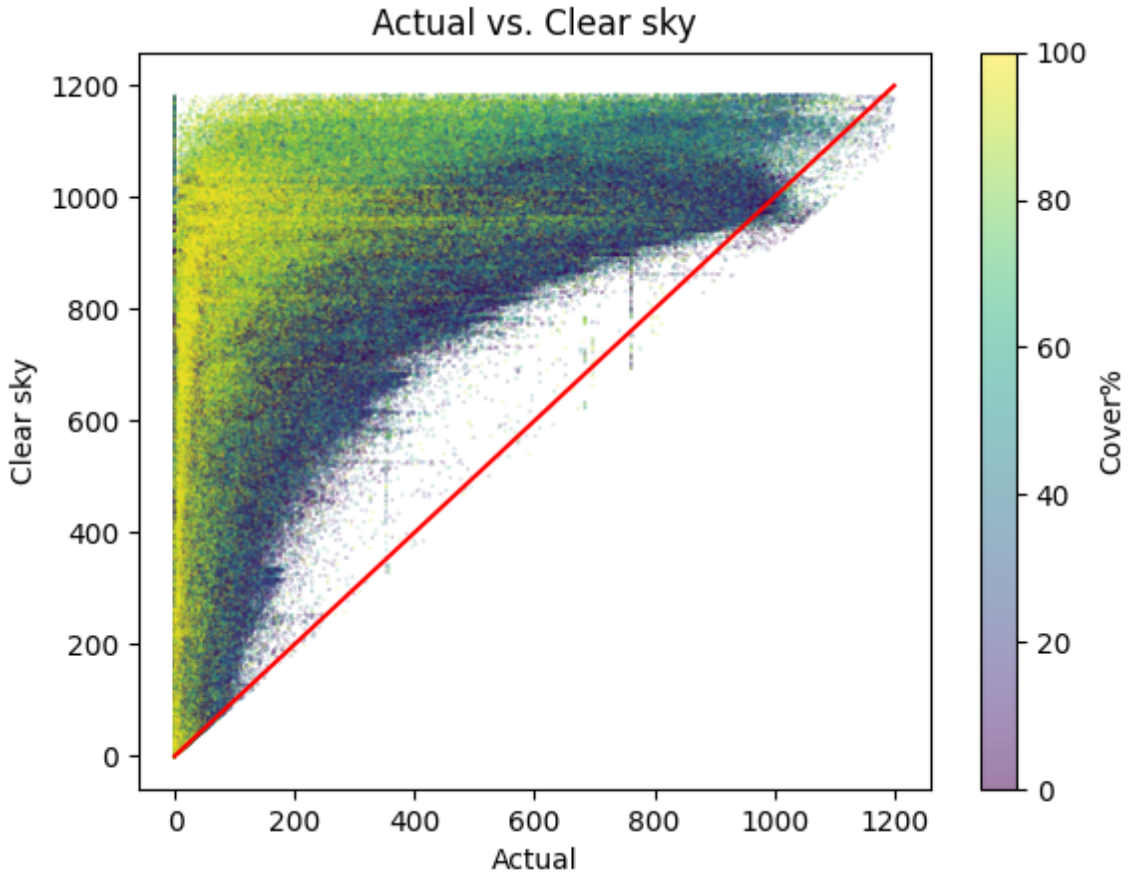


Figure 3.2: Clear sky estimation vs actual readings (in W/m^2), using the Pysolar library, with color coding depending on cloud cover

Here, we focus on the quality control procedures applied to the previously described dataset. We outline the steps taken to identify, assess, and rectify data anomalies, ensuring that the dataset is both accurate and reliable. By implementing rigorous quality control measures, we can mitigate the impact of errors, thereby enhancing the overall robustness of the neural network's final results.

These procedures are mostly heuristic, since most of the input data has already gone through basic quality checks at the stations from

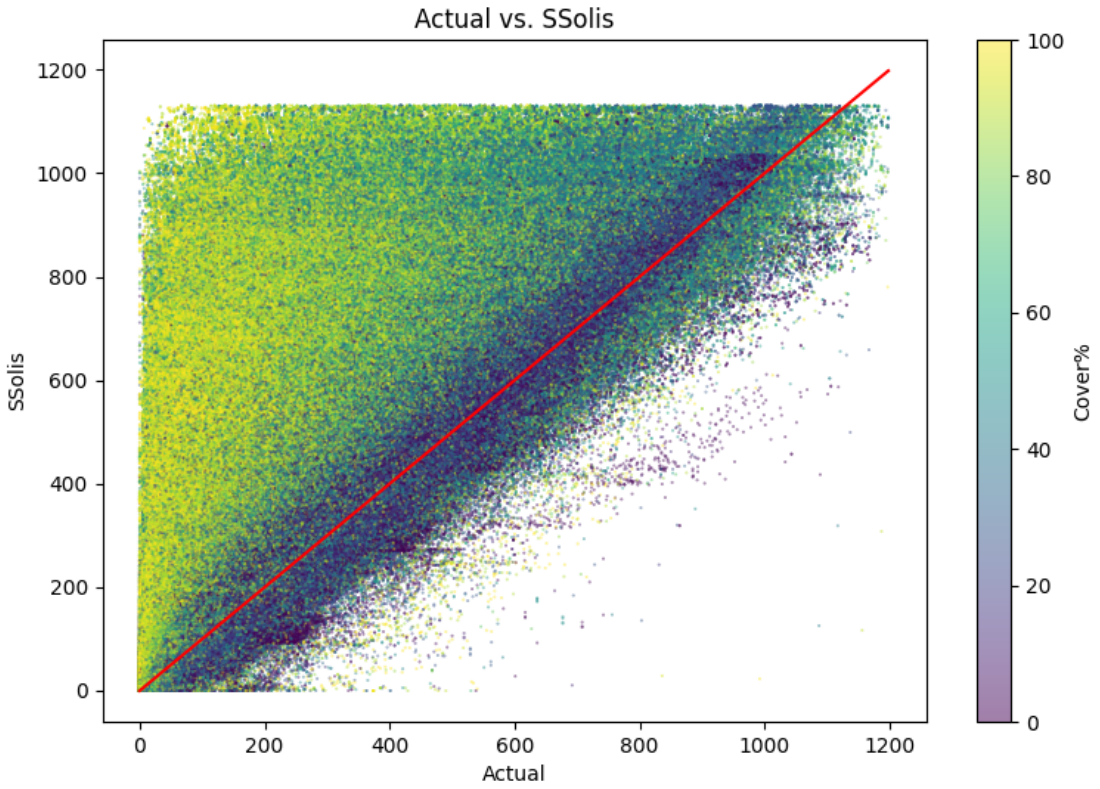


Figure 3.3: Clear sky estimation vs actual readings (in W/m^2), using the Simplified Solis model, with color coding depending on cloud cover

which it was received, barring malfunctions, as well as being cross-referenced through Visual Crossing's system, if it has access to multiple stations. As such, our main goal was to identify any malfunctioning stations that were providing consistently inaccurate data, while avoiding culling good-quality data.

The first method we utilized is a simple one that mostly serves as a sanity check; ignoring all data before each day's sunrise, and after its sunset. Since what this research is interested in is solar radiation, it is reasonable to remove all values that are guaranteed to not be

relevant to it. This happens to have the added benefits of reducing the dataset's size by 30-40%, as well as making each data point "worth more", on average, to the neural network. The sunrise/sunset times are fetched using a slightly modified version of the Suntime library[7], which implements the sunrise/sunset calculation algorithm described in [21].

Secondly, after noticing that some stations were providing constantly repeating data, most likely due to a malfunction, as mentioned earlier, we decided to add a check for exactly this circumstance. This check completely removes a station from consideration if there is a developer-mandated number of continuous values within a developer-mandated distance of each other. For the purposes of this work, the number of continuous values was 5, and the distance between them was 4.

Finally, since we have access to a reasonably accurate clear-sky estimation of solar radiation, as mentioned in section 3.3, we can utilize it to check whether a station either consistently and massively underestimates solar radiation under clear skies, pointing to a solar radiation detection equipment problem, or has problematic cloud cover detection. If either of these assumptions are true, the most cautious course of action to be taken is to, once again, remove the station from con-

[7] https://github.com/SatAgro/suntime

sideration.

With all of this in mind, the dataset we decided to use contained 500 sampling locations, following the distribution seen in Figure 3.1. All of these samples have passed the aforementioned sanitization procedures, and contain a minimum of 100 valid hourly readings, totaling to 7,073,396 readings. As discussed previously, the values contained in these samples can be seen in Table 3.1. We also include some basic statistics regarding the values we utilized in training our models, for which there is practical meaning, spanning all readings, in Table 3.3.

Value (unit)	**Min**	**Max**	**Mean**	**Std. Deviation (σ)**
Cloud cover (%)	0.0	100.0	54.09	34.06
Temperature (°F)	-37.0	114.8	63.06	15.74
Humidity (%)	1.2	100.0	65.74	18.94
Wind speed (mph)	0.0	172.1	8.70	4.94
AQI	1.0	159.0	41.60	21.54
Dew point (°F)	-53.6	94.1	50.02	14.77
Clear sky estimation (W/m^2)	0.0	1199.0	294.42	260.56
Visibility (mi.)	0.0	60.3	10.29	6.52

Table 3.3: Statistics for selected variables.

3.3 Preprocessing

The success of machine learning models, particularly neural networks, hinges on the quality and relevance of the input data. In our case, all of the available variables can be seen above, in Table 3.2, along with the coordinates of the area these readings have been acquired from. However, not all these variables contribute equally to the predictive

performance of a model. Selecting the most suitable elements from a weather reading dataset—those that are most informative and relevant to the prediction task at hand—is a critical step in building an effective neural network.

Our goal in this section is to effectively curate the set of variables to be used as inputs to our neural network, both by removing, as well as potentially introducing some that can be inferred from the ones already in our disposal, in order to enhance accuracy, reduce computational complexity, and improve model generalization.

In order to achieve this, we used the Pearson correlation coefficient between each of these values, as well as the solar radiation target, shown in Figures 3.4 and 3.5. This provided a good initial pointer as to which elements of a weather reading are indeed worth considering for use in our experiment, which we were later able to confirm in testing.

For the purposes of training our NNs, we define 3 subsets of the initial dataset, shown in Table 3.4. Subset 1 simply contains all the items shown in Table 3.1, except for beam and diffuse radiation. Subset 2 contains the elements with the 6 highest Pearson correlation coefficient values in relation to GHI. Subset 3 is a more heuristically curated dataset, which seemed to perform well during initial testing.

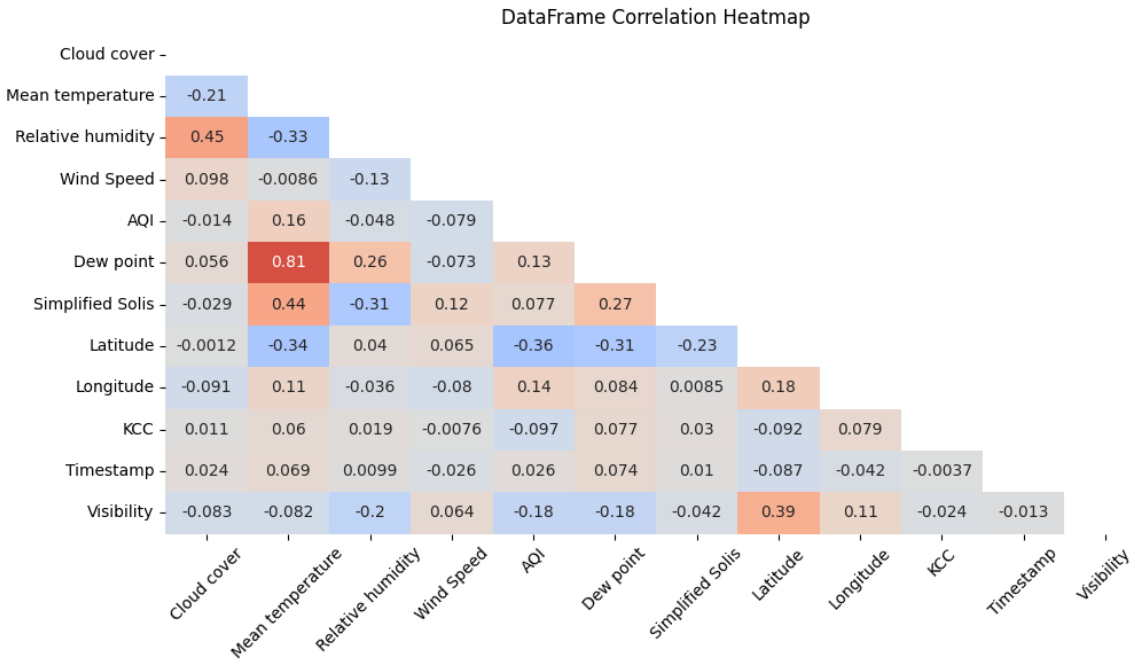


Figure 3.4: Pearson correlation coefficient between each element of a reading

Table 3.4: Dataset subsets

Subset	**Subset elements**
Subset 1	All sample contents, except for beam and diffuse radiation components
Subset 2	Cloud cover, mean temperature, relative humidity, dew point, Simplified Solis estimate, latitude
Subset 3	All elements in Subset 2, also including longitude, UNIX timestamp, visibility

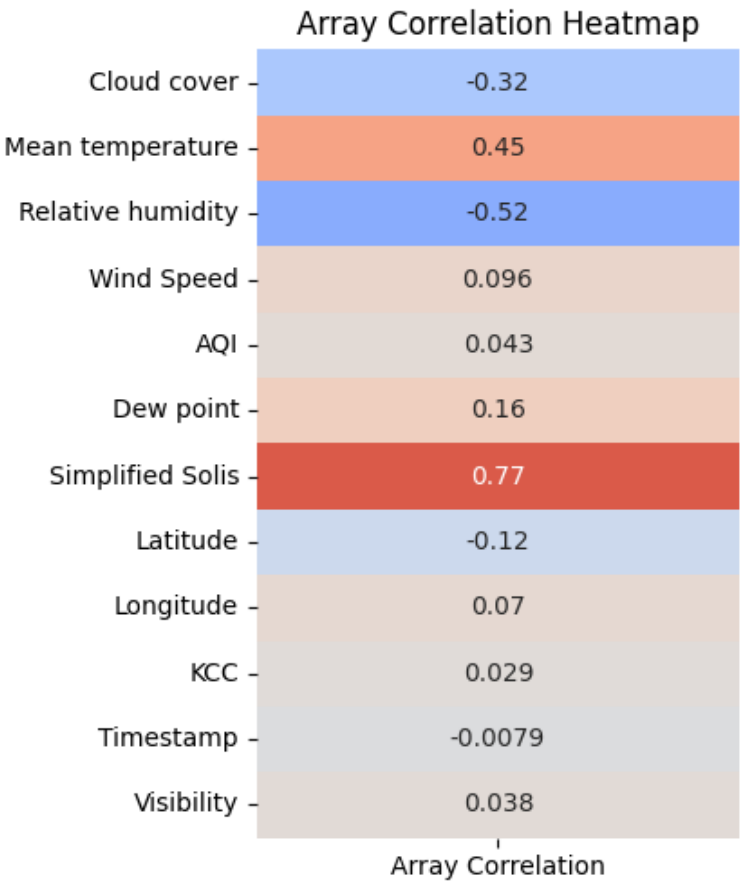


Figure 3.5: Pearson correlation coefficient of each element relating to solar radiation readings

Chapter 4

Implementation of Neural Networks for Solar Radiation Prediction

In this chapter, we describe in detail the process of implementing NNs which predict solar radiation. More specifically, in 4.1, we explain the methodology we followed in order to identify the set of hyperparameters for our models. Afterward, in 4.2, we describe the methodology we followed in order to utilize the aforementioned hyperparameters to train and test our models. The hyperparameters we tested, along with their values, can be seen in Table 4.2, the NN architectures we ended up using are shown in 4.3, and the dataset subsets we utilized to train said architectures were discussed in Section 3.3.

4.1 Designing the NNs

Selecting the appropriate hyperparameters for a neural network is a critical step in the model development process, as these parameters directly influence the network's performance and generalization capabilities. Algorithm hyperparameters, such as learning rate and batch size, and model hyperparameters, such as number of layers and activation functions, determine how the model learns from data, balances the bias-variance tradeoff, and converges during training.

Hyperparameters can have a multitude of different values, and figuring out the optimal setup manually can take a prohibitively large amount of time. For this reason, it is paramount to automate this process, while also making sure to cover enough of the viable hyperparameter space. This can be achieved by performing a grid search within said space, as described in [22], using the R^2 score as our NN evaluation metric, as suggested by [23]. However, instead of doing a single pass through the entire training set, we can perform k-fold cross-validation within the grid search.

In our case, the hyperparameters we tested for, as well as their values, are shown in Table 4.2. The best-performing values, marked with an asterisk, were utilized to train the NN, as will be described in the next section. For this experiment, we performed grid search using

two different partitions of our dataset. The first one contained all the array elements, as shown in Table 3.1, and the second one contained the 6 elements with the highest Pearson correlation coefficient between the elements and GHI readings, as seen in Figure 3.5 and listed in Table 4.1. As a final note, in order to make the time complexity of the grid search reasonable, we used a random sample of 5% of the dataset, which was then split into a training and a testing set, with an analogy of 4:1.

Table 4.1: Elements with the 6 highest Pearson Correlation Coefficient Values

Element
Cloud cover, in %
Mean temperature, in °F
Relative humidity, in %
Dew point, in °F
Simplified Solis clear sky estimation, in W/m^2
Latitude

Table 4.2: Hyperparameter Configurations and Their Corresponding Values Tested Within Grid Search

Hyperparameter	**Values**
Batch size	4[1], 256, 512*
Learning rate	0.001, 0.01*, 0.05, 0.1
Optimizer	Adam*, RMSprop
Number of first layer neurons	4, 128, 256, 512, 1024
Number of second layer neurons	0, 64, 128, 256, 512
Number of third layer neurons	0, 32, 64, 128, 256
Dropout layer (20%)	True, False*
Batch normalization layer	True*, False

[1] Using a batch size of only 4 causes the NN to be prohibitively slow, so this value was rejected very early on.

4.2 Training and Testing the NNs

Having identified the best performing hyperparameters, we can utilize them to build 3 models with different architectures, as seen in Table 4.3, which we can train on our data subsets, as seen in Table 3.4. The training/testing/validation split we used for said data was 6:1:1, or 75%/12.5%/12.5%.

Table 4.3: Neural Network architectures used

Architecture	**Number of Hidden Layers**	**Neurons Per Layer**
Architecture A	2	512, 256
Architecture B	2	1024, 512
Architecture C	3	1024, 512, 256

As far as training is concerned, the loss function used for it is Huber loss, a well-known loss function commonly used in robust regression problems. We opted for training to run through 20 epochs, which seemed to strike a serviceable balance between good optimization and time efficiency, as the addition of further epochs offered statistically insignificant improvement. As a byproduct of this, even though we utilized early stopping, it was fully unnecessary, as it never came up.

Chapter 5

Evaluation of Neural Network Models

The final step in this process is evaluating our models' performance using suitable numerical metrics, as well as making observations regarding comparisons with actual GHI readings.

5.1 Choosing our Metrics

In order to have a fair and wholistic evaluation of our models, we needed to utilize metrics that are robust, and can express both the performance of the models themselves against actual readings, as well as the performance of the networks as far as generalization or other functional factors go, so as to make sure that they have not, for example, overfitted to our dataset.

For this reason, we decided to use metrics very commonly used

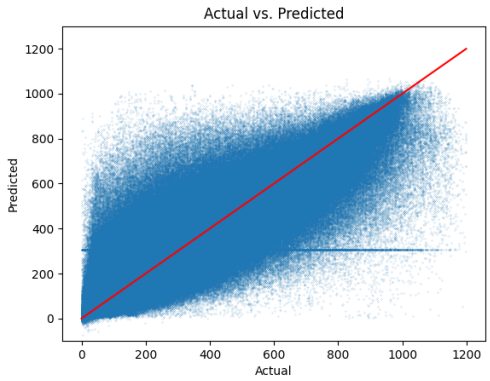
in regression problems. More specifically, to measure the networks' performance on the testing set against actual readings, we used both a filtered version of the mean average percentage error (MAPE), where we filtered all rows with an actual reading of less than $50W/m^2$, since MAPE is very sensitive to small values, and the root mean squared error (RMSE). For the more general evaluation of our models, we utilized R^2 Score, since it works very well for this purpose, as discussed in Section 4.1.

Finally, in Appendix A, we provide predicted versus actual plots for the networks not directly referenced here, so that their results can be studied in a more heuristic approach.

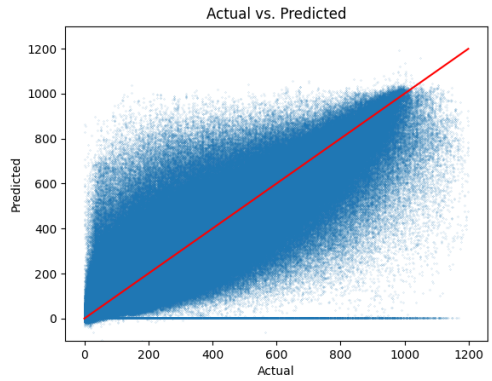
5.2 Evaluation and Commentary

The first point of note is that the inclusion of the Köppen–Geiger climate classification in Subset 1 seems to have introduced some problematic behavior, as seen in Figures 5.1a, 5.1b and 5.1c. Our assumption here is that, considering the somewhat concentrated nature of our dataset, it is likely that only one station from an area characterized by a rare climate classification subgroup with comparatively few values was included, thus creating a false correlation between a very common solar radiation value for that area with its KCC subgroup. For this

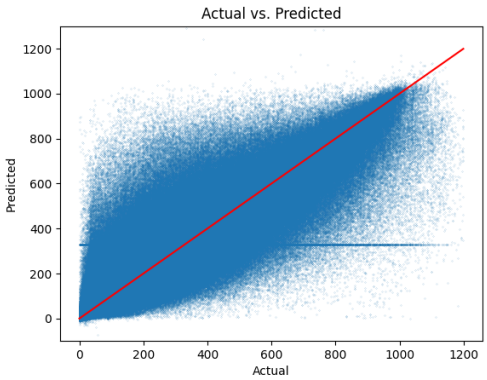
reason, we also include the results excluding KCC for Subset 1, and we utilized these results for evaluation via metrics.



(a) Architecture A



(b) Architecture B



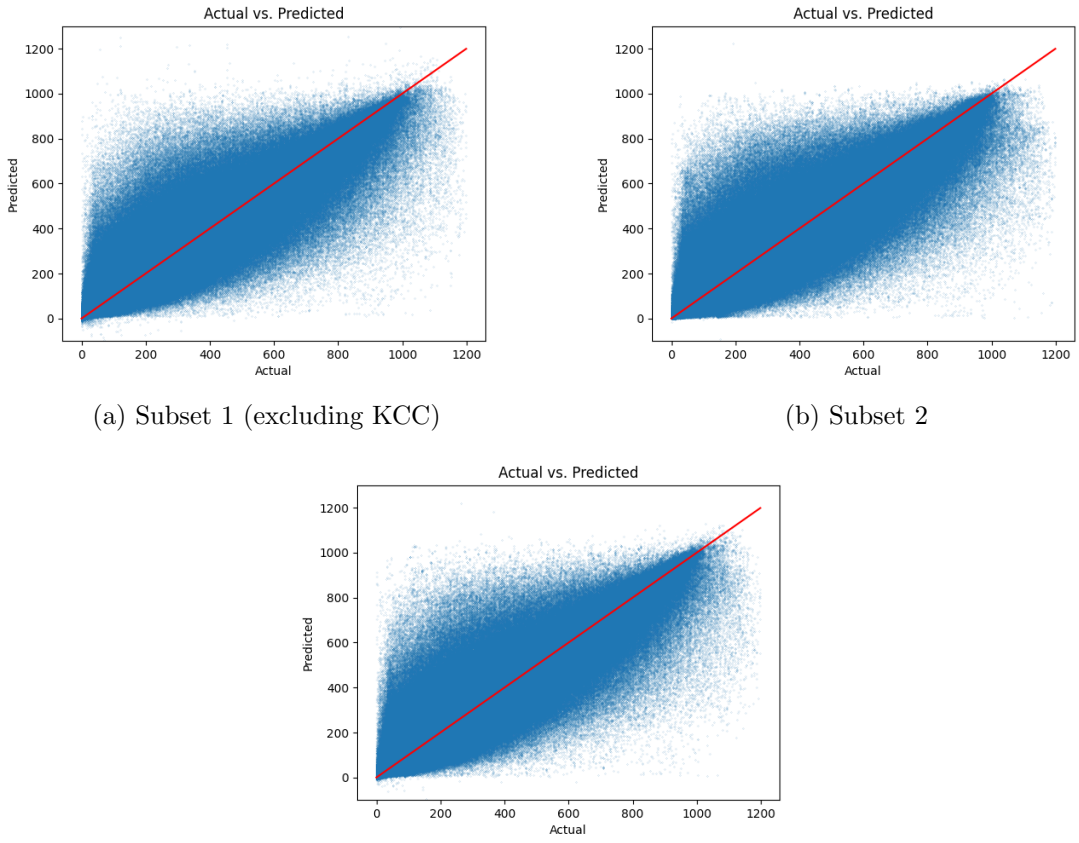
(c) Architecture C

Figure 5.1: Plot of actual vs predicted GHI measurements (in W/m^2) using Subset 1

As is evident in Table 5.1, Architecture C seems to perform marginally better than the other architectures in all categories, and similarly for Subset 1[1], but the final numerical performance of the NNs is very similar, and not quite compelling enough, as the models seem to have trouble with readings including high cloud cover, evident in the higher variance in the lower to middle values in both the plots included in

[1]Excluding KCC.

Figure 5.2, as well as the rest in Appendix A. One potential explanation for this behavior is that the relatively simplistic nature of our models is not enough to process such a large amount of varied data. This leads us to believe that while adding extra "filtering" via simple hidden layers to our architectures can provide slight improvements, a different, more bespoke approach to the creation of the neural networks may be of more use.



(a) Subset 1 (excluding KCC)

(b) Subset 2

(c) Subset 3

Figure 5.2: Plots of actual vs predicted GHI measurements (in W/m^2) using Architecture C for different subsets.

As far as general performance is concerned, as shown in Appendix

B, all of the models don't seem particularly biased against any specific solar radiation values as indicated by the LOWESS (locally weighted scatterplot smoothing) curves (red lines), which attempt to fit the data into a simple curve in order to show any tendencies toward unexpected values in it, but there is definitely a non-trivial number of outlier values, where the NNs either over- or undershoot the prediction massively, a fact that becomes more apparent in larger solar radiation values. However, another thing of note, is that while Subset 1[2] performed better according to our metrics, Subset 3 seems to better limit, or at least balance out, the aforementioned outlier values, as seen, for example, in the LOWESS curves in Figures 5.3 and 5.4.

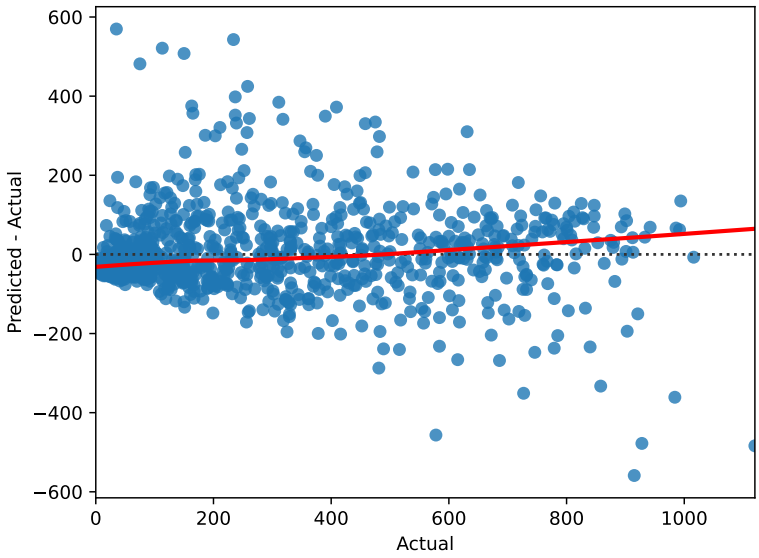


Figure 5.3: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 1 and Architecture B, excluding KCC

[2]Excluding KCC.

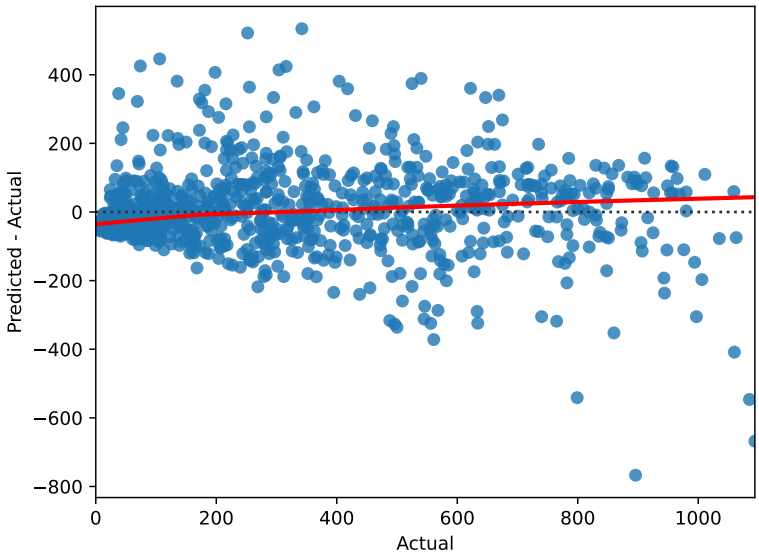


Figure 5.4: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 3 and Architecture B

Comparing to past work, the only other work utilizing similarly standard NN architectures is RENES [7]. As we can see, the even simpler MLP architecture utilized there is not at all suited for the increased volume of our dataset (shown in Figure 5.5).

Table 5.1: Testing performance metrics for Neural Network architectures

Architecture	**Dataset Subset**	**R^2 Score**	**Filtered MAPE(%)**	**RMSE (W/m^2)**
Architecture A	Subset 1*	0.804	36.81	116.19
	Subset 2	0.782	38.39	122.73
	Subset 3	0.792	37.29	122.71
Architecture B	Subset 1*	0.807	35.84	115.47
	Subset 2	0.782	38.95	122.88
	Subset 3	0.786	37.13	122.78
Architecture C	Subset 1*	0.809	35.46	114.36
	Subset 2	0.783	38.83	122.22
	Subset 3	0.802	37.02	121.43

* Excluding KCC.

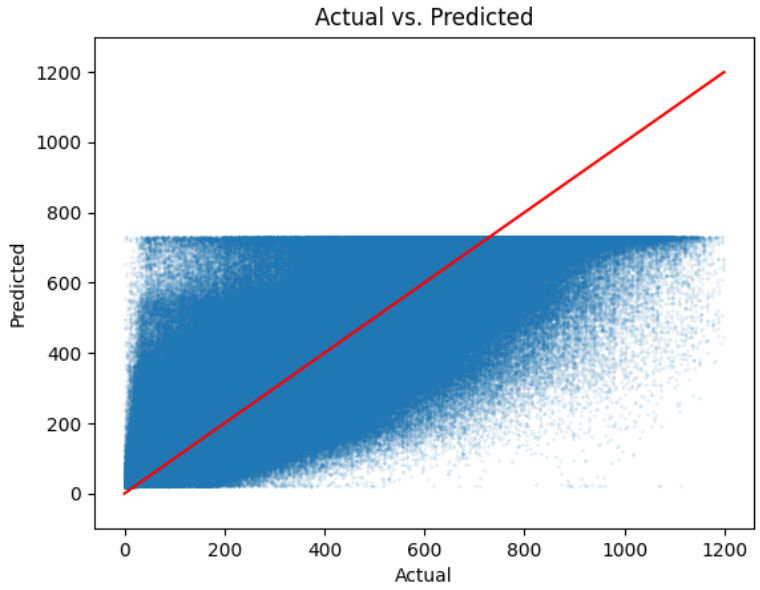


Figure 5.5: Plot of actual vs predicted GHI measurements (in W/m^2), using the MLP architecture and weather reading elements used in RENES [7]

Chapter 6

Conclusion and Future Work

Solar radiation prediction is still an unsolved problem, and there is more than enough room for research in the area, especially when covering a large land mass. However, outside of dedicated research institutes, easily accessible weather data is hard to find, and it is even harder to ensure its quality. We believe that the advent of machine learning can offer a solution to this problem, especially with its ability to "fill in the blanks", so to speak, proving a valuable asset when weather data is incomplete or faulty.

This work presented an initial attempt at addressing the issues with finding a more openly sourced dataset and the methods with which to quality control said dataset. Afterward, the same dataset was used to train and test a number of different MLP architectures, so as to provide some initial expectations for future work in the field. The results were promising, but nowhere close to the more focused, area-

limited works of the past, such as [4].

As far as building upon this project goes, there are a few pointers we can offer as to what to look towards. Firstly, a new dataset can be created, which features areas that had lackluster coverage in the one used in this work, such as South America, Central and North Africa, Asia and Australia. These areas were contained in the initial results fetched from Visual Crossing, but the data received usually did not pass through our filtering process. Perhaps data from these areas can be found via coming in contact with local organizations that keep weather records, but that is beyond the scope of this project.

Secondly, considering the performance of other works discussed in Chapter 2, we firmly believe that a dataset similar to ours can be used to train more sophisticated, hand-crafted models, which are very likely to offer more promising results.

Finally, we believe that an openly available web application which provides and builds upon the functionality of solar radiation prediction, similar to the now-defunct RENES app, described in [7], can be highly useful, especially for areas lacking in professional weather tracking equipment.

Bibliography

[1] Aminmohammad Saberian et al. "Modelling and prediction of photovoltaic power output using artificial neural networks". In: *International journal of Photoenergy* 2014.1 (2014), p. 469701.

[2] Saurabh Saxena, Ritambhara Dubey, and Neda Yaghoobian. "A Model for Predicting Ignition Potential of Complex Fuel in Diurnally Variable Environment". In: *arXiv preprint arXiv:2206.02518* (2022).

[3] Ozan Şenkal and Tuncay Kuleli. "Estimation of solar radiation over Turkey using artificial neural network and satellite data". In: *Applied energy* 86.7-8 (2009), pp. 1222–1228.

[4] D Guijo-Rubio et al. "Evolutionary artificial neural networks for accurate solar radiation prediction". In: *Energy* 210 (2020), p. 118374.

[5] A Sfetsos and AH Coonick. "Univariate and multivariate forecasting of hourly solar radiation with artificial intelligence techniques". In: *Solar Energy* 68.2 (2000), pp. 169–178.

[6] Sujan Ghimire et al. "Deep solar radiation forecasting with convolutional neural network and long short-term memory network algorithms". In: *Applied Energy* 253 (2019), p. 113541.

[7] Athanasios Aris Panagopoulos, Georgios Chalkiadakis, and Eftichios Koutroulis. "Predicting the power output of distributed renewable energy resources within a broad geographical region". In: *ECAI 2012*. IOS Press, 2012, pp. 981–986.

[8] David Elizondo, Gerrit Hoogenboom, and RW McClendon. “Development of a neural network model to predict daily solar radiation”. In: *Agricultural and forest meteorology* 71.1-2 (1994), pp. 115–132.

[9] Tianqi Chen and Carlos Guestrin. “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[10] Xianglong Li et al. “Probabilistic solar irradiance forecasting based on XGBoost”. In: *Energy Reports* 8 (2022), pp. 1087–1095.

[11] Ziming Liu et al. “Kan: Kolmogorov-arnold networks”. In: *arXiv preprint arXiv:2404.19756* (2024).

[12] Sercan Ö Arik and Tomas Pfister. “Tabnet: Attentive interpretable tabular learning”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. 8. 2021, pp. 6679–6687.

[13] Dengchang Ma et al. “Photovoltaic Power Output Prediction Based on TabNet for Regional Distributed Photovoltaic Stations Group”. In: *Energies* 16.15 (2023), p. 5649.

[14] James T Peterson and Edwin C Flowers. “Interactions between air pollution and solar radiation”. In: *Solar Energy* 19.1 (1977), pp. 23–32.

[15] Hoyt C Hottel. “A simple model for estimating the transmittance of direct solar radiation through clear atmospheres”. In: *Solar energy* 18.2 (1976), pp. 129–134.

[16] Wladimir Köppen, Esther Volken, and Stefan Brönnimann. “The thermal zones of the earth according to the duration of hot, moderate and cold periods and to the impact of heat on the organic world (Translated from: Die Wärmezonen der Erde, nach der Dauer der heissen, gemässigten und kalten Zeit und nach der Wirkung der Wärme auf die organische Welt betrachtet, Meteorol Z 1884, 1, 215-226)”. In: *Meteorologische Zeitschrift* 20.3 (2011), pp. 351–360.

[17] Franz Rubel and Markus Kottek. "Observed and projected climate shifts 1901-2100 depicted by world maps of the Köppen-Geiger climate classification". In: *Meteorologische Zeitschrift* 19.2 (2010), p. 135.

[18] Franz Rubel et al. "The climate of the European Alps: Shift of very high resolution Köppen-Geiger climate zones 1800–2100". In: *Meteorologische Zeitschrift* 26.2 (2017), pp. 115–125.

[19] Gilbert M Masters. *Renewable and efficient electric power systems*. John Wiley & Sons, 2013.

[20] Kevin S. Anderson et al. "pvlib python: 2023 project update". In: *Journal of Open Source Software* 8.92 (Dec. 2023), p. 5994. ISSN: 2475-9066. DOI: 10.21105/joss.05994. URL: http://dx.doi.org/10.21105/joss.05994.

[21] United States Naval Observatory. Nautical Almanac Office. *Almanac for Computers*. Nautical Almanac Office, United States Naval Observatory, 1990. URL: https://books.google.gr/books?id=GbDC6AaH-kAC.

[22] Davide Chicco. "Ten quick tips for machine learning in computational biology". In: *BioData mining* 10.1 (2017), p. 35.

[23] Davide Chicco, Matthijs J Warrens, and Giuseppe Jurman. "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation". In: *Peerj computer science* 7 (2021), e623.

Appendix A

Comparisons of model predictions to actual GHI readings

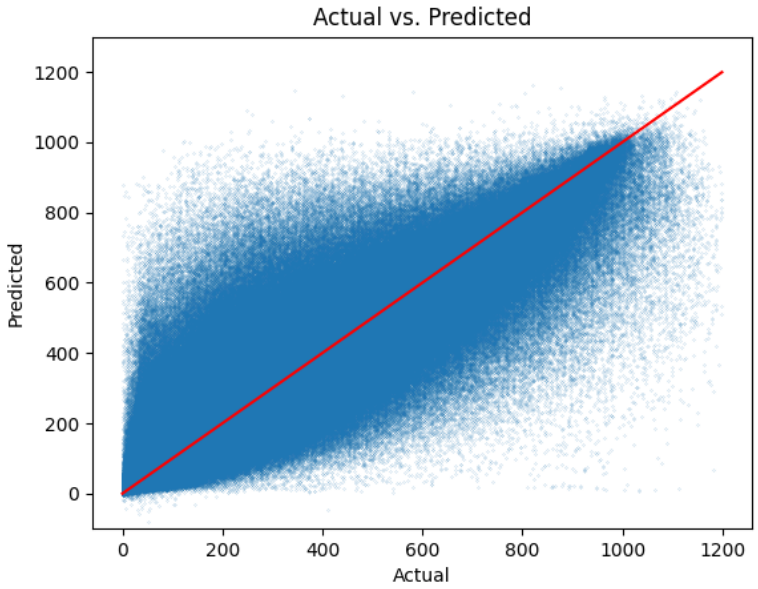


Figure A.1: Plot of actual vs predicted GHI measurements (in W/m^2), using Subset 1 and Architecture A, excluding KCC

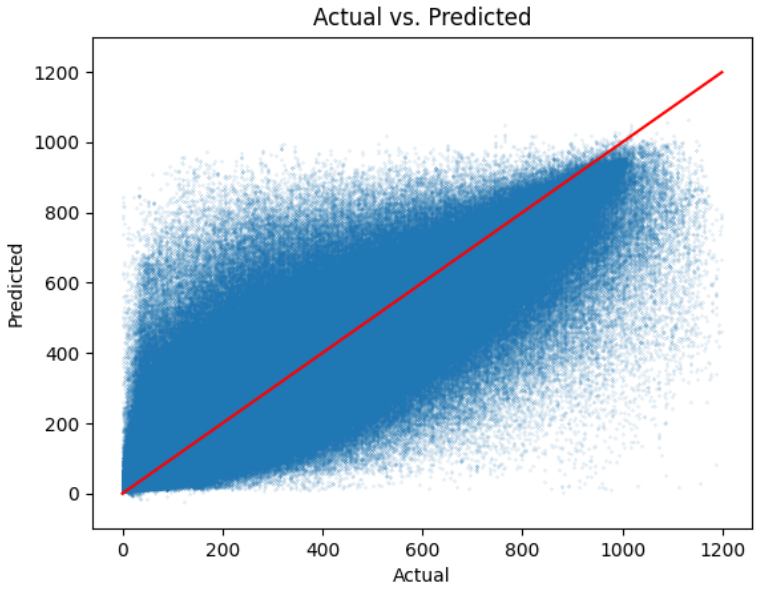


Figure A.2: Plot of actual vs predicted GHI measurements (in W/m^2), using Subset 2 and Architecture A

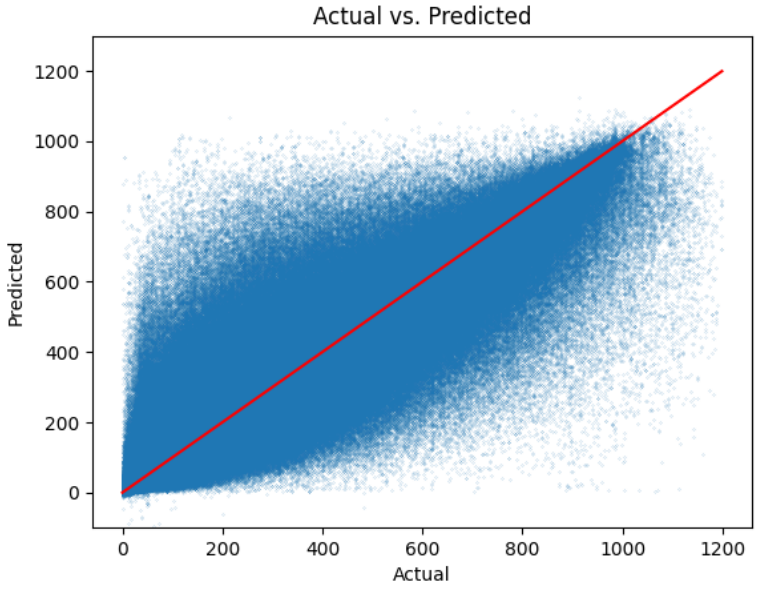


Figure A.3: Plot of actual vs predicted GHI measurements (in W/m^2), using Subset 3 and Architecture A

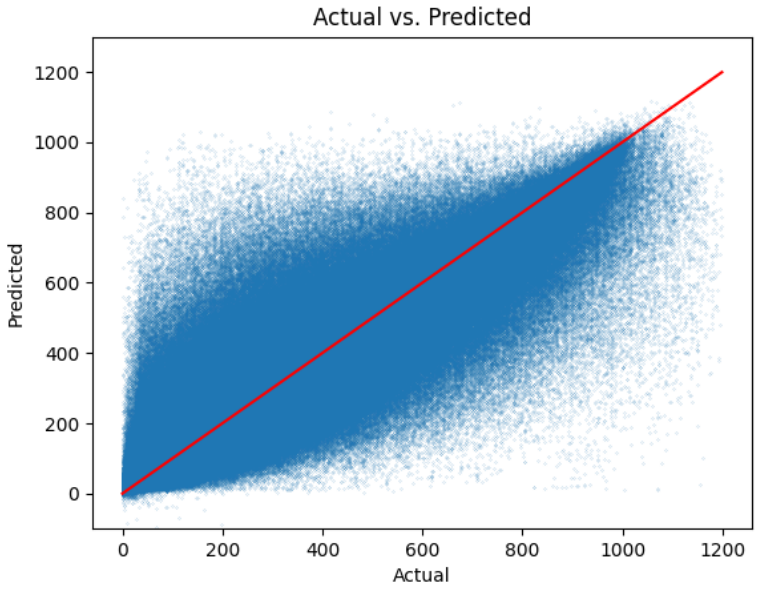


Figure A.4: Plot of actual vs predicted GHI measurements (in W/m^2), using Subset 1 and Architecture B, excluding KCC

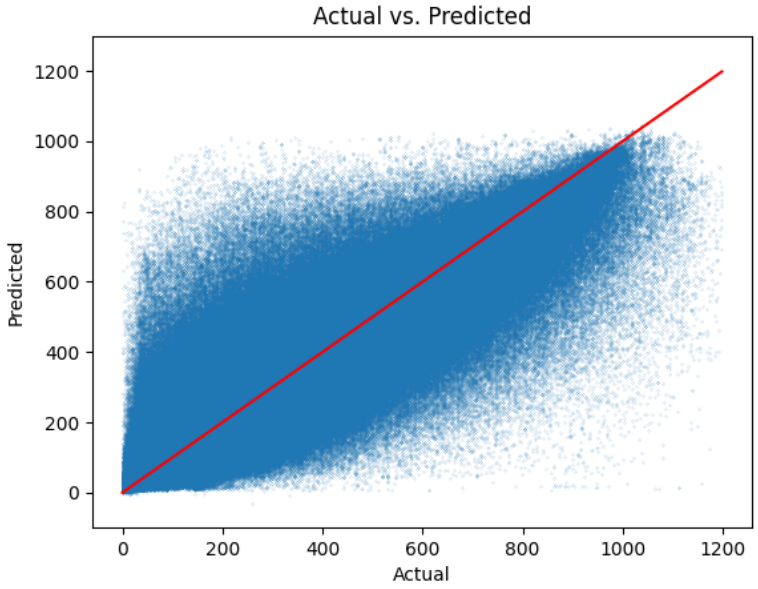


Figure A.5: Plot of actual vs predicted GHI measurements (in W/m^2), using Subset 2 and Architecture B

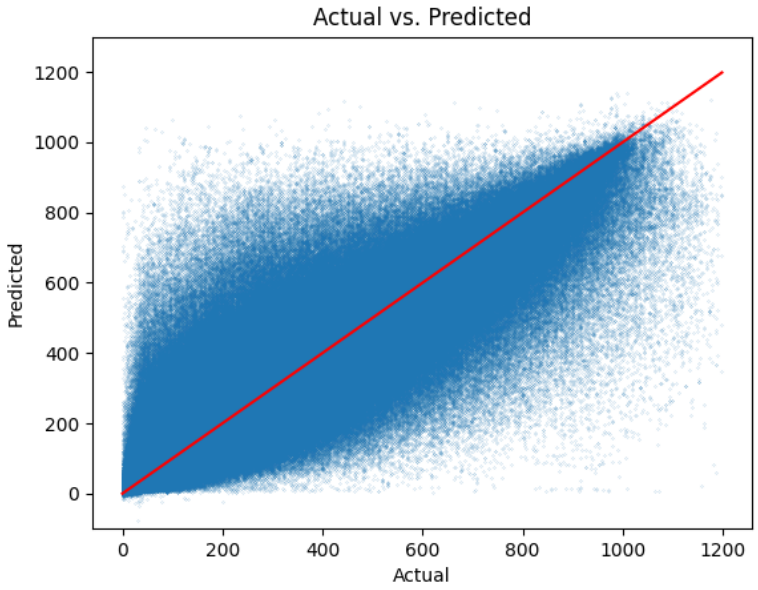


Figure A.6: Plot of actual vs predicted GHI measurements (in W/m^2), using Subset 3 and Architecture B

Appendix B

Residual analysis plots

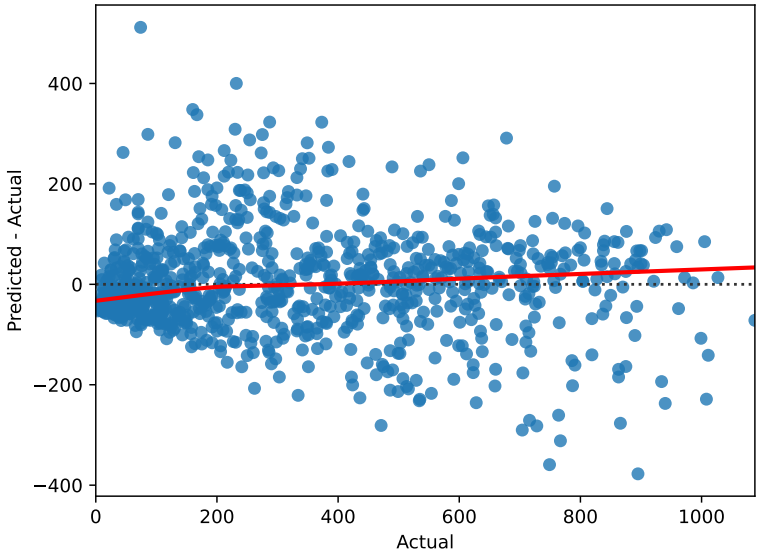


Figure B.1: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 1 and Architecture A, excluding KCC

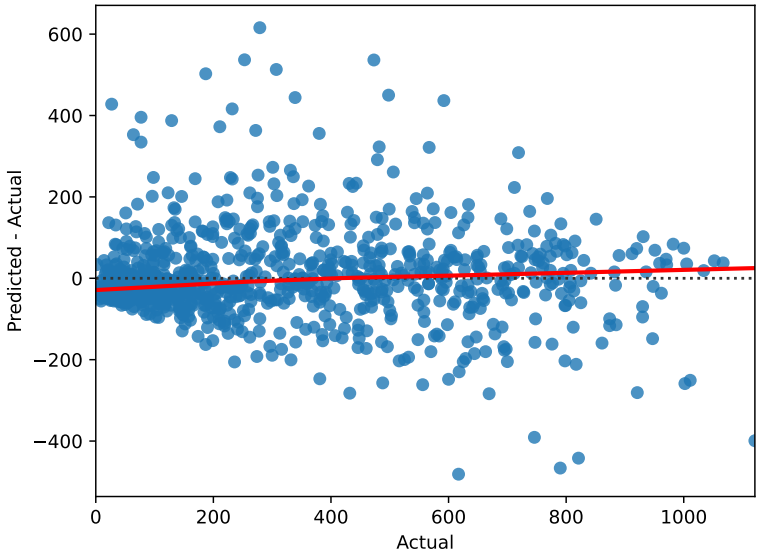


Figure B.2: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 1 and Architecture C, excluding KCC

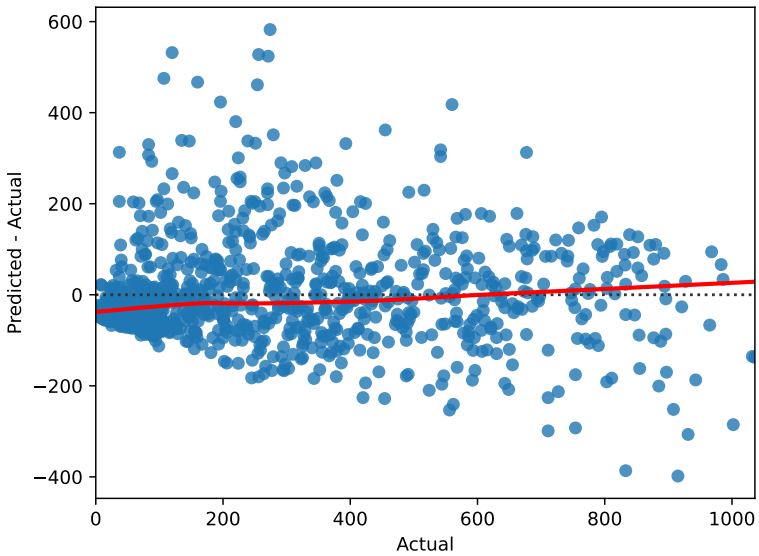


Figure B.3: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 2 and Architecture A

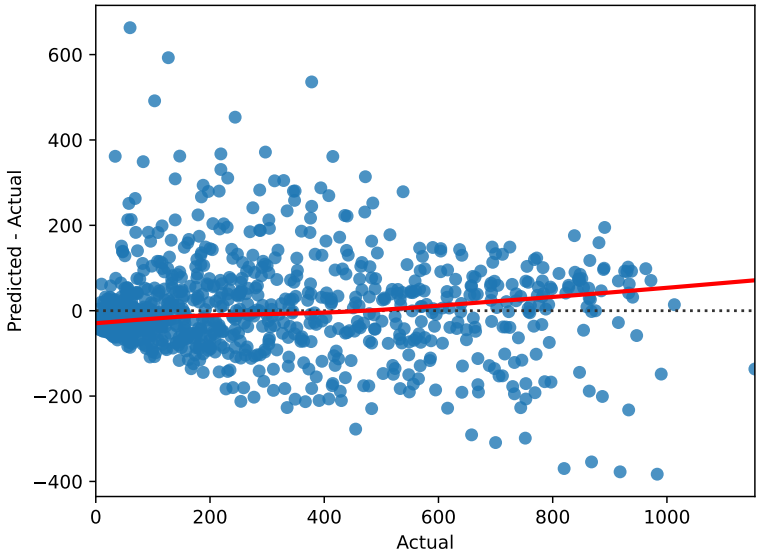


Figure B.4: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 2 and Architecture B

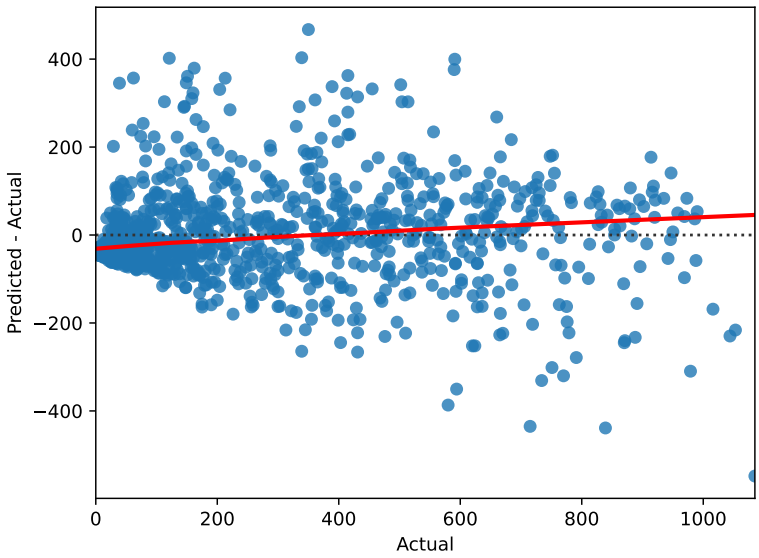


Figure B.5: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 2 and Architecture C

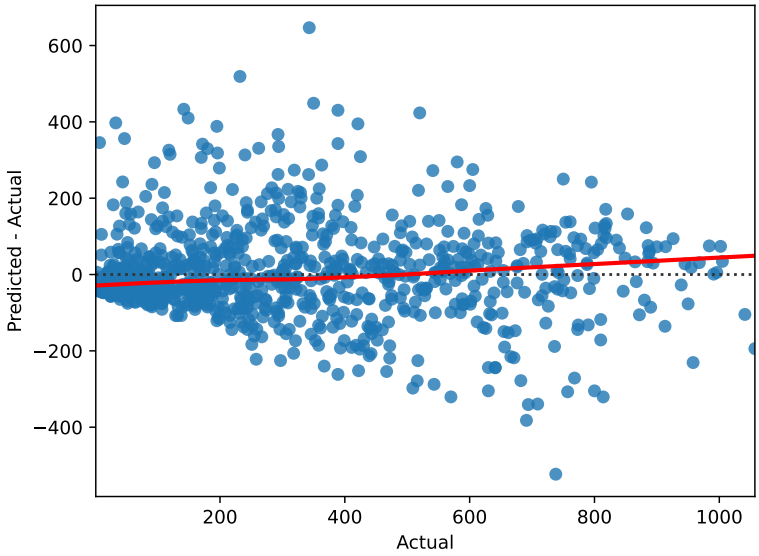


Figure B.6: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 3 and Architecture A

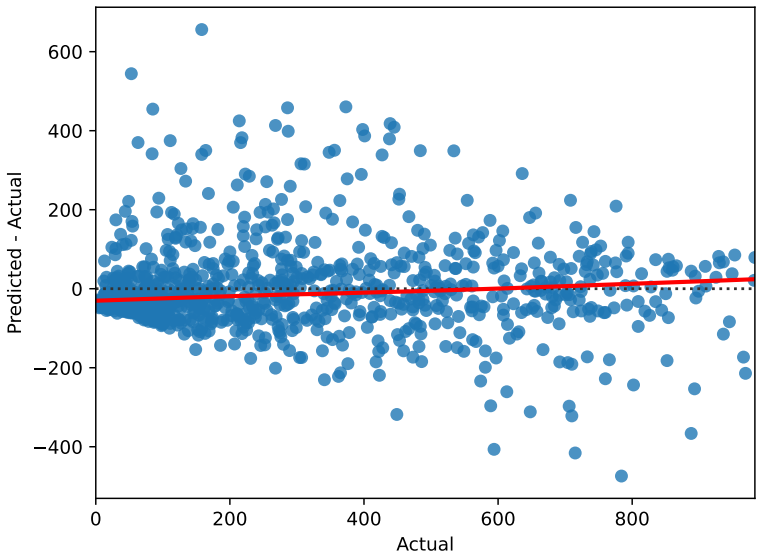


Figure B.7: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 3 and Architecture C

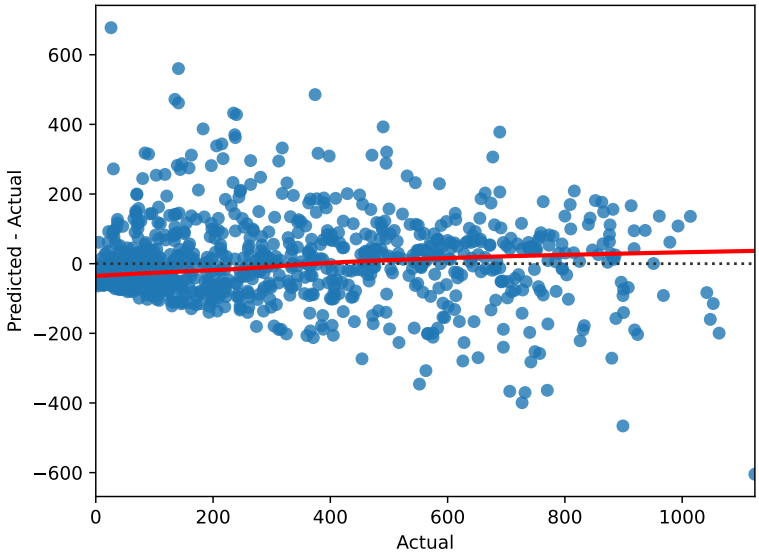


Figure B.8: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 1 and Architecture A

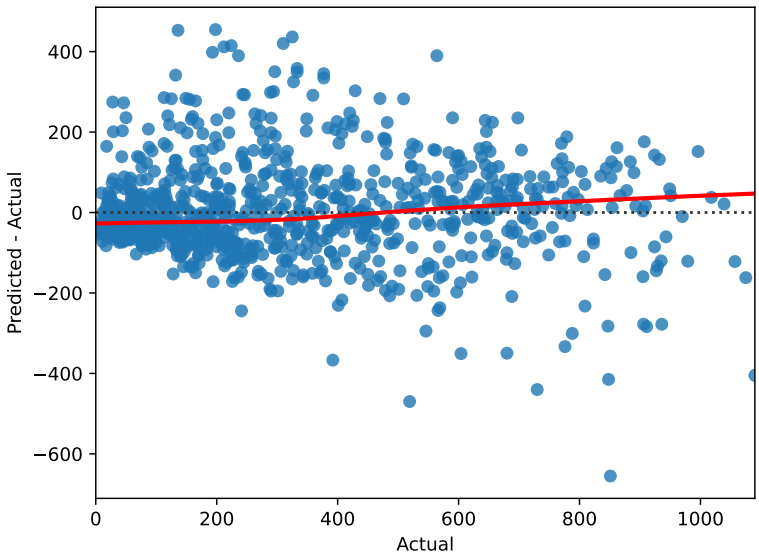


Figure B.9: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 1 and Architecture B

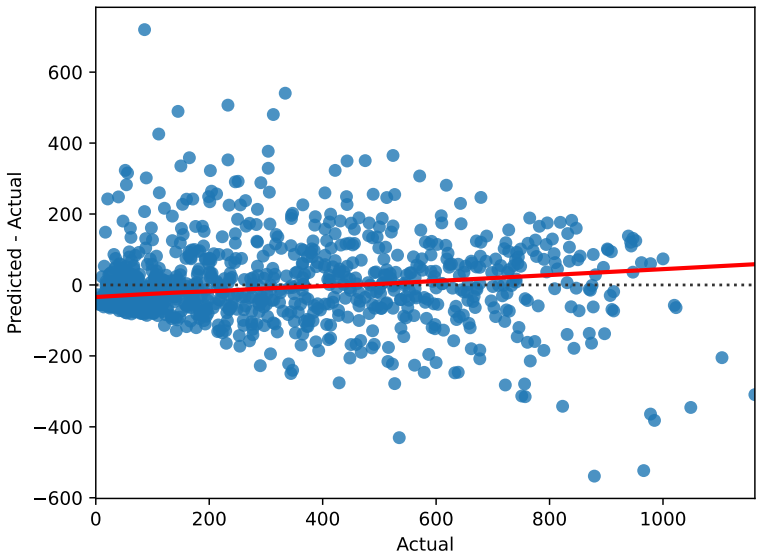


Figure B.10: Residual plot from 1000 randomly selected predicted-actual/actual value pairs, using Subset 1 and Architecture C