Master Thesis

Master's Programme in Embedded and Intelligent Systems, 120 credits

Energy Predictions of Multiple Buildings using Bi-directional Long short-term Memory

Computer science and engineering, 30 credits

Halmstad 2020-11-23

Anton Gustafsson, Julian Sjödal

HALMSTAD UNIVERSITY

ABSTRACT

The process of energy consumption and monitoring of a building is time-consuming. Therefore, an feasible approach for using transfer learning is presented to decrease the necessary time to extract required large dataset. The technique applies a bidirectional long short term memory recurrent neural network using sequence to sequence prediction. The idea involves a training phase that extracts information and patterns of a building that is presented with a reasonably sized dataset. The validation phase uses a dataset that is not sufficient in size. This dataset was acquired through a related paper, the results can therefore be validated accordingly. The conducted experiments include four cases that involve different strategies in training and validation phases and percentages of fine-tuning. Our proposed model generated better scores in terms of prediction performance compared to the related paper.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

S2S Sequence-to-Sequence

RNN Recurrent Neural Network

MLR Multiple Linear Regression

SVR Support Vector Regression

GLR Generalized Linear Regression

BPNN Backpropagation Neural Network

RBFNN Radial Basis Function Neural Network

GRNN General Regression Neural Network

ANN Artificial Neural Network

CNN Convolutional Neural Network

TSNE T-distributed Stochastic Neighbor Embedding

RMSE Root Mean Squared Error

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

LSTM Long Short Term Memory

ReLu Rectified Linear Unit

Tanh Hyperbolic Tangent

1 INTRODUCTION

1.1 INTRODUCTION

Energy prediction in buildings is a topic that has grown substantially in importance throughout the last decade. This is because of the increasing threat of global warming and the increasing demand for energy consumption. In 2018, there was an overall increase of 2.9% in energy consumption, which is the fastest growth since 2010 [1]. Buildings are one of the largest factors of this overall increase. This increasing energy usage is not only due to the increasing demand for comfort but also for the number of new technologies that is embodied in everyday life. In 2009, the European Union concluded that the average annual energy consumption in Europe is 220 kwh/m^2 with residential buildings having 200 kwh/m^2 and non-residential 300/m^2. [2] According to the EU, the top three largest energy-consuming areas are buildings, transport, and industrial with 41%, 32%, and 25%, respectively [2]. This information only emphasizes the importance of acknowledging the weak points and streamlining the use of energy overall but most importantly in buildings. A vast amount of studies and projects have, therefore, tried and are trying to find these weak points in order to find possible enhancements of efficiency or unnecessary usages of energy.

Previously conducted projects have been trying various types of machine learning and artificial intelligence methods to find possible efficiency enhancements in energy consumption. These algorithms have been applied to the problem due to the high complexity and the vast amounts of information that needs to be processed. Furthermore, possible algorithms to imitate real-world scenarios are many but often accurately mediocre. Finding good models that with ease and precision imitates the real-world is difficult. It is difficult due to finding relationships between data and target values as the features are often many and data is often fluctuating or is not as abundant as required. This area of study is heavily explored and many perspectives and strategies can be found.

1.2 MOTIVATION

As mentioned, the energy demand and the environmental threat from global warming are both increasing. Much of the energy demand is satisfied through environmentally unethical ways, which increases the importance of streamlining energy usage substantially. To tackle both of these problems, considering that the majority of energy consumption is from buildings, understanding energy consumption patterns and increase energy efficiency in buildings is relevant. Furthermore, models in machine learning and deep learning requires a trainable dataset. There are possibilities to acquire data from various sources, such as virtual computations and simulations, but the best way to imitate the real-world is to monitor the consumption of buildings in real-time. This strategy is very time-consuming due to the need for representation of different seasons throughout the year. If accuracy is a priority, it is important to note that multiple years might need to be monitored.

1.3 PROBLEM FORMULATION

The long process of data collection through monitoring a building is essential for creating an accurate model. This study aims to examine the idea of using transfer learning between data sets to predict energy consumption. In theory, this will decrease the necessary time spent on monitoring the building and accumulating data. Our study uses data acquired from another building and applies the information onto the energy prediction for a target building, which does not have a reasonable amount of data to create an accurate model for longer time periods. As of now, research has been made on buildings in close vicinity to each other but these studies have handled that data as one dataset. In our study, we could apply our methodologies to two different datasets and could obtain promising results.

More specifically, the proposed model is using S2S bidirectional LSTM that first uses the larger, more complete, dataset to train, and later fine-tunes on the smaller dataset. Bidirectional LSTM is a model that has shown promise in both generalization and accuracy. Also, it is adapted for complex modeling. This project approaches the problem from different angles that include both different time-steps and also different percentages of fine-tuning on the target set. Therefore, using S2S prediction seems to be a good approach not only due to the flexibility it brings but also due to its handling arbitrary data well [3] [4]. Important to note is that the two datasets are not connected, which means that the monitored buildings are in different locations. However, both sets have similar features.

The following are the contribution of this study.

- Improving the estimation of energy consumption in one building and simultaneously comparing the results with previous study.
- Giving evidence for using an additional dataset with low correlation to target dataset in training, showing the amount of needed data of target building can be low to none.
- Proposing a S2S Bidirectional-LSTM Model structure that predicts sequentially that keeps the interaction between data points.

The project code can be found in the following GitHub repository: `https://github.com/Drev-Source/Master-Thesis.git`

2 RELATED WORK

2.1 REGRESSION METHODS FOR ENERGY CONSUMPTION

The literature review contains an in-depth analysis of the current state of the art. The section presents a variety of model-complexities with some more simple than others. It starts with MLRs, GLRs, and SVRs and transitions into more complex models such as ANNs, RNNs, and lastly LSMTs.

Energy consumption prediction is a topic that has several different perspectives and approaches. Regression is one of the models that is used commonly. MLR, GLR, and SVR are also used frequently in past studies with better outcome in some cases and higher application in others. The MLR strategy is, historically, the most extensively tested regression model for energy prediction problems [5] [6] [7] [8] [9]. One reason for the extensive application of MLR can be its simplicity compared to the other models. It is a model based on linear regression that adds variables that gives the algorithm the option to create more sophisticated approximations than a straight line as the basic linear regression does.

GLR is another moderately used regression method [10] but it is a valid approach due to the acceptable results it can produce. Since GLR does not have any restriction on the distribution of the data, it is a more flexible function compared to MLR. SVR, a variation of support vector machine, is a very successful model when it comes to performance [11] [12] [13] [14] [15] [16]. It is applicable to many different regression problems and is widely used in this area of research. The reason is that it is quite well suited for generalization purposes and is an effective tool for real-value function estimation. Furthermore, SVR trains with a symmetrical loss function that penalizes both high- and low- error estimations. The combination of the penalization and Vapnik's (epsilon)-insensitive approach creates an adaptive generalization model, which is well suited for energy forecasting [17] [18].

ANNs is found to be very useful and reliable when it comes to predicting future values of energy consumption. This architecture has a considerable amount of different alterations when it comes to approaching and tackling prediction problems. Many of these alterations, of the original ANN model, have been successful and are used immensely throughout the community. BPNN, RBFNN, and GRNN

are examples of ANN alterations that have come to be useful in generating better results in the prediction problem of energy consumption. The BPNN is a heavily used method that updates its weights through backpropagation, hence the name. Furthermore, the term ”backpropagation” refers to the process where the derivatives of a network error, with respect to the network, are fed back to the network and is used to adjust the weights so that the model can predict more accurately and reduce the errors to minimum. Therefore, BPNN offers a method of minimizing the errors between the obtained outputs and desired target values [19][20][21][22][23][24][25].

The main function of a typical GRNN is to estimate a linear or nonlinear regression surface on independent variables. More specifically, the network computes the joint probability density function of input vectors and output. The architecture of the GRNN is divided into 4 groups, input layer, hidden layer, summation layer, and output layer. This structure gives the method more generalizable outputs that are beneficial in many studies [19][26].

A core part of RBFNNs is that it only uses a single hidden layer compared to other ANNs that may use multiple. Also, instead of using commonly used activation functions, such as ReLu and Tanh, this model uses radial basis functions. The single hidden layer application together with the combination of different activation functions in the RBFNN makes it comparably more efficient in terms of computational speed than BPNN. Its performance and accuracy is also as high as BPNN [27][28].

RNN is a type of ANN that is able to store information inside the neuron itself. The neurons in an RNN is connected back into itself which enables it to perform a memory like feature. This is quite useful in regression and time-series problems in general and does produce a good performance in most cases. Furthermore, LSTM is a developed RNN that enhances the memory aspects of the typical RNN into a more sophisticated version. It can store and send more data inside the layer, in between nodes, which is not only very useful for energy consumption prediction but also other areas of prediction problems [29][30][31][32][33][34][35][3].

LSTM is a RNN-architecture [30]. Bidirectional LSTM and CNN based LSTM are two extensions of LSTM. The latter is a developed method that embraces the good features of both methods in order to acquire the complex patterns that energy consumption sometimes possesses [36]. The CNNs are most often used in image analysis but show promise in combination with more time-series focused methods [36] [33].

The article ”Evaluation of Bidirectional LSTM for Short- and Long-Term Stock Market Prediction” [37] conducts a study that tries to evaluate and compare different LSTM architectures that involve short- and long-term prediction. It considers both bidirectional and fully-stacked LSTM models. The authors have done this on a stock market prediction problem to test their models on one of the hardest real-world applications for time-series prediction. They have used the LSTM structure because of the arbitrary sizes in time-steps without the vanishing gradient problem. When compared the mentioned models to a non-altered LSTM and to a Multi-layer perceptron artificial neural network, the bi-directional LSTM and the stacked LSTM performed the best with a slight edge to the the bi-directional LSTM. The the bi-directional LSTM performed better and converged faster compared to the other models [37].

A more closely related work to ours compared to this is an article called the Sequence-to-Sequence model for building energy consumption prediction [4]. This article uses an encoder/decoder structure while handling an hourly data resolution. Important to note is that the encoder is using a one-layer bidirectional LSTM while the decoder is a basic one-layer decoder. The input sequence is weekly while the output is daily. The dataset contains information on, for example, temperature, humidity, and outdoor irradiation, amongst others. The dataset only contained data from a total of three months and is clearly shown in the results as the differences between target and prediction values are a bit too significant. Although, the authors see potential in the model and the measurable error can be seen as acceptable due to the lack of data [4].

Another related work is the article from the authors Marino et.al. with the title of ”Building energy load forecasting using Deep Neural Networks” [38]. This article compares the performance of an LSTM network to an S2S-LSTM. Both of the models were tested against two different time-step resolution data, one-minute and one-hour. Interestingly, the LSTM network performed well on the one-hour set while falling behind on the one-minute set. The S2S-LSTM, on the other hand, performed well at both of them which indicates that the S2S-LSTM seems to handle different time-steps better. The authors mention later that this performance difference is due to the S2S-LSTM being more flexible than the LSTM in handling arbitrary number of previous measurements as input to estimate an arbitrary number of future time-steps [38].

Previously mentioned state of the art energy consumption predictions are more suitable for different scenarios. Some situations may require more complex models and others do not. Conventional regression methods will generally perform well when there is a small amount of data or if complexity is low. However, not considering previous energy consumption use for predictions, might hinder the model in certain cases. The bi-directional LSTM do take advantage of such information and is one of the key points of using this model for the project. As this model does require more data than conventional regression models, the transfer learning method of adding more data and information from other datasets is necessary.

The paper ”Data driven prediction models of energy use of appliances in a low-energy house” [39] which is going to be discussed, creates a dataset and produces results from predicting the usage of appliances. The authors of this article have conducted a study to improve the knowledge base of the energy consumption of appliances in a residential house. They have accumulated data themselves and have used several different machine learning algorithms that try to model the energy consumption. MLP, SVR, random Forest, and gradient boosting machines (GBM) were tested and every model used cross-validation as the validation function. GBM were determined to perform the best and was able to explain 97% of the variance in the training set and 57% in the testing set. As mentioned, the authors accumulated the data set themselves and this was done from January to June 2016. The term ”appliances” include various different equipment such as kitchen devices in the kitchen such as refrigerator and microwave, entertainment instruments like TV and computer, and ordinary apparatuses such as a hair dryer. The authors lastly highlight the importance of the occupancy in the building and its effect on energy consumption. Air pressure had higher prediction potential than anticipated and that could be explained by the correlation it has with other weather features that could, depending on good or bad weather, increase the occupancy in the house. The results from the GBM method are given in terms of root mean squared error (RMSE), R-squared (R^2), mean absolute error (MAE), and mean absolute percentage error (MAPE). The best models values for the four values were 66.65, 0.57, 35.22, and 38.29, respectively. Another important part to note is that this study is testing on data points that are distributed throughout the dataset instead of future points. An increase in data would probably have resulted in better scores and overall better performance. Another limitation mentioned by the authors is that this study was only conducted on one house and would have been interesting and may have resulted in a more generalizable model if

more houses were monitored [39]. This study is highly related to this project because of using the same dataset and for comparison purpose.

3

METHOD

Bi-directional LSTM is the method used for this study. In theory, to train neurons with past and future data points should benefit accuracy of the model. Transfer learning was practiced for adaptability of information between models together with S2S method for handling time intervals of different lengths. The structure of the model is shown in figure 1.

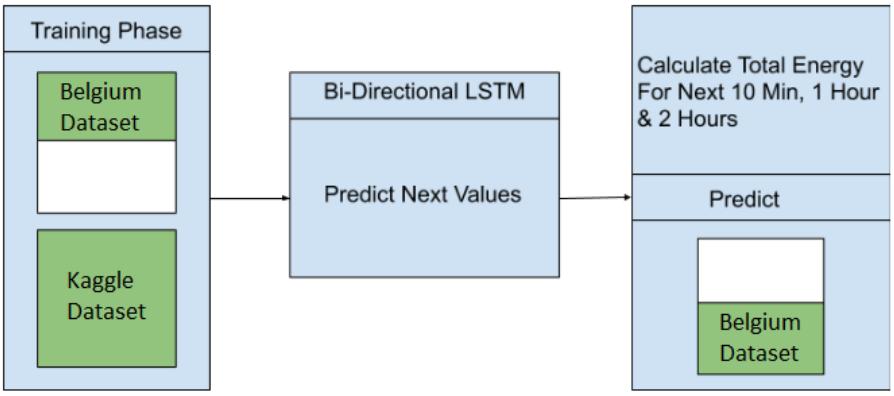

Figure 1: The proposed model structured first by two datasets where Kaggle Dataset is completely used and Belgium Dataset partially. The data is prossesed through a Bidirectional-LSTM module. The model is lastly tested on the remaining part of Belgium Dataset in with different time-intervals.

3.1 MODEL

We being with presenting the RNN and its LSTM structure. Then, the LSTM will be explained in detail together with its bi-directional aspects. At the end, S2S will be described and its application will be justified.

3.1.1 *RNN*

A neural network is a combination of processing units called neurons. These neurons receive input and apply a nonlinear activation function in order to model nonlinear relationships. However, the neurons can also model time-series relationships. This relationship model can be improved by adding a recurrent link.

The recurrence is a feedback connection that has been added in order to send back previous information and re-receive it. This is represented in Figure 2. RNNs can be stacked together and be expanded to further improve modelling of the problem while keeping the already improved time-series predictions. This expanded version of RNN is represented in Figure 3.

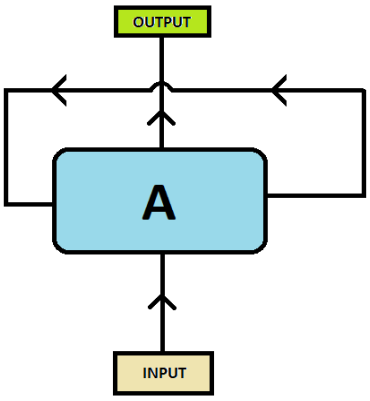

Figure 2: A simple RNN structure with one hidden layer is shown.

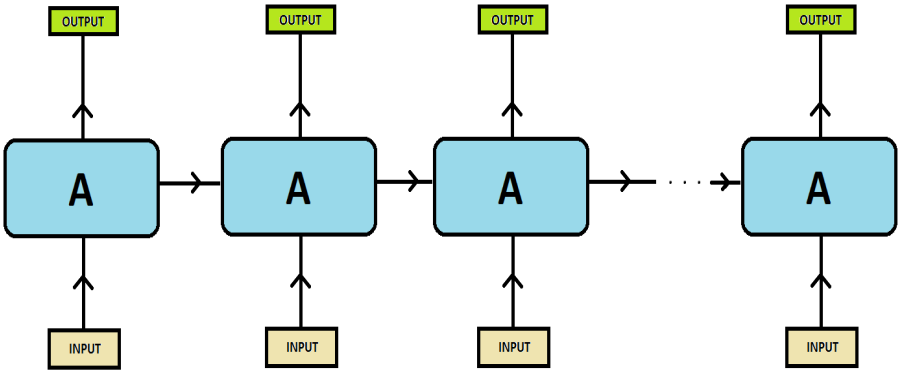

Figure 3: The expansion of the RNN is shown. The figure shows an RNN with one hidden layer unfolded in time.

3.1.2 *LSTM*

Even though RNNs are used in time series problems with success, improvements can still be made. The two major flaws of RNN is the handling of the context in, for example, text analysis. The RNN only contacts the previous text while the following is ignored. The second

flaw is the learning time correlation [40]. This is solved with LSTM [30] which adds three types of "gate-like" structures to the basic RNN. These "gate" layers are called forget gate, input gate, and output gate. The LSTM is shown in Figure 4 and its structure is discussed in the following.

The following Figure, 4, has 4 different modules represented. The sigma and the tanh modules refers to that the activation functions sigmoid and tanh, respectively, will be applied to the signal. Figure 4 depicts the structure of one LSTM neuron with its elated gates. The addition and multiplication sign refers that a pointwise addition or multiplication occurs.

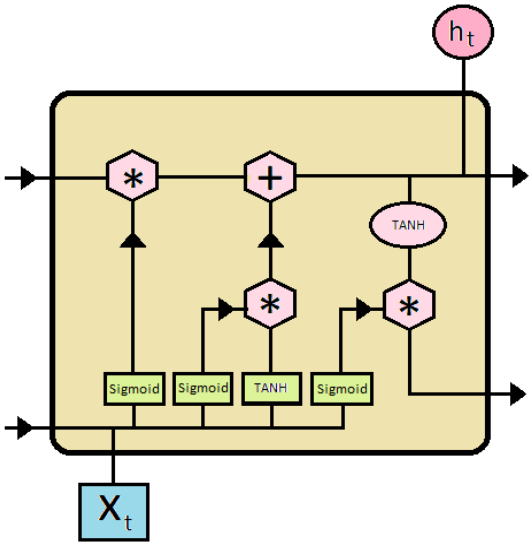

Figure 4: This image visualizes a single LSTM neuron.

The first output out of the sigmoid layer is the forget gate. This gate stores the last moment of historical data and then computes a value between 0 and 1 from previous information and the new data point. The value determines whether the previous information is useful or not.

The second gate which stores the information is the input gate. The signal is passed through two layers of sigmoid and tanh. The first one decides on the value that should be updated. The latter creates a vector that is combined with the output of the sigmoid layer to create an update of the state.

The last gate out of the sigmoid layer is the output gate. The signal passes through a multiplication module and a tanh activation

function. This module is controlling the output of this neuron which means it establishes the need of sending this neuron's information forward. [41]

To conclude, three gates are added to the basic concept of an RNN neuron in order to, more in-depth, control where and what information is being sent. The forget gate controls what information is being sent from previous neurons, the input gate establishes whether information from the input is useful and the output gate controls the output.

3.1.3 *Bidirectional LSTM*

The bidirectional property allows the LSTM neurons to send information back and forth within the model. This enhances the ability to handle sequential data sets, such as text files, speech recordings, and time-series problems. The ability to send back and send forth information about the data creates a model that actively adds the previous context with future ones. This simplifies the process of modeling the relationships within these sequential data sets [42][43]. A simple visualization of the structure can be seen in Figure 5.

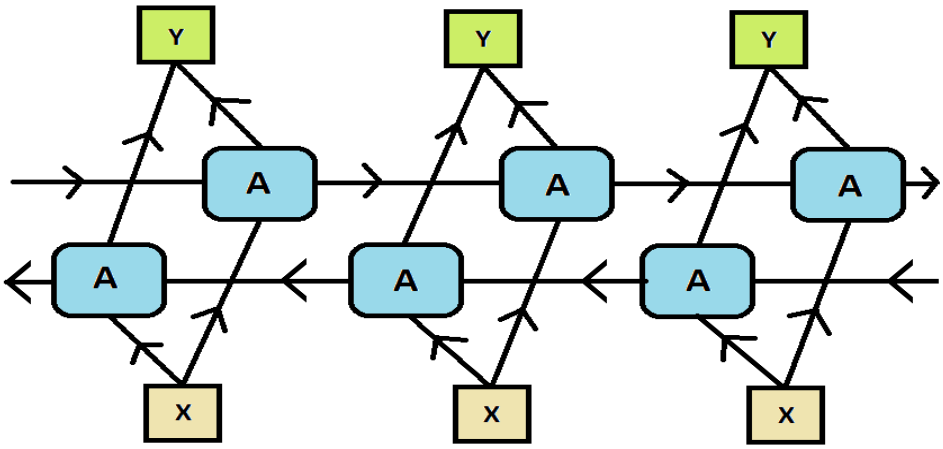

Figure 5: This simple visualization represents a bidirectional LSTM and shows the forward and backward routes the information can take. Here, the "A" represents the LSTM neurons, the "Y" represents the output and the "X" represents the input. The forward and backward paths has not been marked due to unnecessary nature of its direction.

3.1.4 *Sequence-to-Sequence*

For the model to be able to get a contextual perspective as well as to simplify the handling of different time-steps, the S2S structure is implemented. S2S is based on an encoder-decoder strategy that uses a

sequential input to produce a sequential output. More specifically, the system receives a sequence of data points, a period in time, and calculates the following statistical output. The contextual perspective is highlighted through these sequence. To elaborate, in language translation problems, for example, the whole sentence needs to be reviewed in order to get an accurate translation. The same justification can be applied in energy consumption problems. These contexts can be referred to possible daily patterns that might be present [4].

3.2 LOSS FUNCTIONS

Metrics used for evaluation are MAE, MAPE, RMSE and R^2. The MAE describes the mean error of the model's prediction, showing how good the model performs on the majority of data. MAPE is just the percentage of MAE and it is an easier metric for understanding how well the model's performance is. For example, being given a percent value of 10% means the model is on average predicting values with a 10% error. RMSE is a good measure when looking at the prediction of outliers, since RMSE origin from MSE which gives large weights to outliers [44]. R^2 is a measure that describes how well the model can represent the data and how good it fits the data [45]. Naturally, you would want the MAE, MAPE, and RMSE to be as close to zero as possible and R^2 to be as close as possible to one.

3.2.1 *Mean Squared Error*

The Mean Squared Error function (MSE, 1) gives large weights towards outliers. MSE targets the low variance in an estimator because of the least square method. Since large errors is not desirable in the model the MSE is used for such comparison.

$$\mathrm{MSE}(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - \hat{y_i})^2 \tag{1}$$

where, $n_{samples}$ is number of samples, y_i is the true value, $\hat{y_i}$ is the predicted value.

3.2.2 *LogCosh*

The LogCosh function, equals to $\frac{((\hat{y}-y_i)**2)}{2}$. It is small when the difference between $\hat{y}$ and y_i is small and it is large when it equals $|(\hat{y} - y_i)| - \log(2)$. In other words, it is similar to MSE. The difference between LogCosh and MSE is that LogCosh is not affected by extreme wrong predictions [46].

$$\mathrm{LogCosh}(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}} \log\left(\cosh\left(\hat{y_i} - y_i\right)\right) \tag{2}$$

3.2.3 *Mean Absolute Error*

The MAE function corresponds to L1-norm loss [47]. The effect of outliers is minimal.

$$\mathrm{MAE}(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} |y_i - \hat{y_i}| \tag{3}$$

3.2.4 *Root Mean Squared Error*

RMSE (4) calculates the Root value of MSE and can be interpreted as the distance error.

$$\mathrm{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - \hat{y_i})^2} \tag{4}$$

3.2.5 *R-squared*

R^2 (5) calculates how well the model describes the variability of predicted data around the mean. The value range is (-∞, 1), having a negative value indicates that the model describe the variability poorly.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{\sum_{i=1}^{n} (y_i - \frac{1}{n} \sum_{i=1}^{n} y_i)^2} \tag{5}$$

4

DATASET

4.1 BELGIUM DATASET

Belgium Dataset was collected and used by the paper "Data driven prediction models of energy use of appliances in a low-energy house" [39]. It was created in a study conducted in 2017 in Belgium and its purpose was to predict the energy consumption of appliances in a low-energy residential house. The data set has a total of 29 different features. Firstly, the set has environmental data from 8 locations throughout the house and includes both temperature and relative humidity from all places. Secondly, outside environmental data that has been accumulated from both house areas and also from the nearby airport which has situated roughly 12 km from the house. This data have the attributes temperature, relative humidity, wind speed, visibility, and dew point in degree Celsius. In order to improve the generalization of the regression model, other attributes related to appliances, the lighting in house and two random variables were used. The lighting features refers to the energy usage for the lighting around the house. The appliances are the energy usage of electronics around the house. A list of appliances used at the house can be found in Table 1. Since the authors of this study has used the same dataset as the one that we have used in our study, the dataset is used for the validation of our proposed model [39].

4.2 KAGGLE DATASET

The second dataset is presented by Taranveer Singh via the website Kaggle [48] and is used during the training phase of this project. The reason for choosing this dataset is because of the similarities between this and the first dataset, from a feature perspective. This dataset has data of the weather which includes temperature, humidity, and dew-point. This is important because of prediction possibilities, that the exogenous is present to enable the modelling of the environment in which the building is standing. Furthermore, the dataset does have consumption features that enable the core purpose of the project, energy consumption prediction. These features are the current energy consumption of specific rooms, items, or the total consumption. This dataset contains data that has been collected over a year which means that information from all four seasons are present. This makes it viable for use as a training phase set, in terms of this project, since it includes data of all seasons and periods throughout the year. Other-

wise, it would have made the prediction of the energy consumption for the missing season's data invalid or impossible

Table 1: This table represents appliances and electrical apparatuses used for data collection.

Office	Garage and Attic	Kitchen	Dining
2 Desktop Computers	Rain Water Pump	Fridge	WIFI Booster
3 Computer Screens	Electric Garage door	Induction Cooktop	ZigBee coordinator
1 Router	Computer	Kitchen hood	Electrical Blinds
1 Laptop	Musical Instruments	Microwave	
1 Copier-Printer	Amplifier	Oven	
Electric Blinds		Dishwasher	
		Coffee machine	

Living	Laundry	Ironing	Room 1
TV 138 cm	Small Fridge	Alarm Clock	Alarm Clock
Hard Drive enclosure	Upright Freezer	Radio	Radio
DVD Player	Winecellar for 160 bottles	Electrical Blinds	Electric Blinds
Cable Box	Dryer	Iron	2 Lamps
Laptop	Washing Machine		
Ink-jet Printer	Internet Router		
Electrical Blinds	Internet Hub		
	Network Attached Storage		

Room 2	Room 3	Game	Bathrooms
Desktop Computer	Laptop	TV 93 cm	4 Electric Toothbrushes
Monitor	Alarm Clock	Internet Router	Hair Dryer
Alarm Clock		DVD Player	
Electrical Blinds		PlayStation	

5 DATA ANALYSIS

5.1 BELGIUM DATASET

For our study, we obtained the power consumption data in unit watt (W) from the energy data with a unit of (Wh) by using the formula $\frac{x_i}{(1000*3.6)}$. The two features are then summed into one feature called "HouseConsum".

Calculating the correlation between each feature and plotting them in a heat-map (Figure 6), gives us features importance. An example of negative correlation, can be seen between the two features RH_out (Humidity) and T_out (Temperature) in Figure 7. When the feature T_out rises the feature RH_out drops. High correlation is represented in Figure 8, between the features Tdewpoint (Dewpoint) and T_out (Temperature). The features are almost identical.

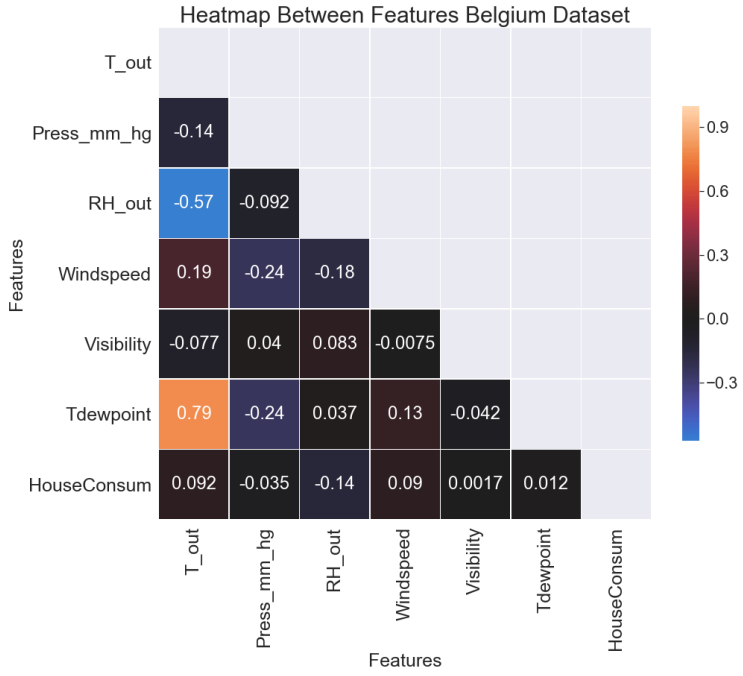

Figure 6: Heatmap of correlation between features used in Belgium Dataset

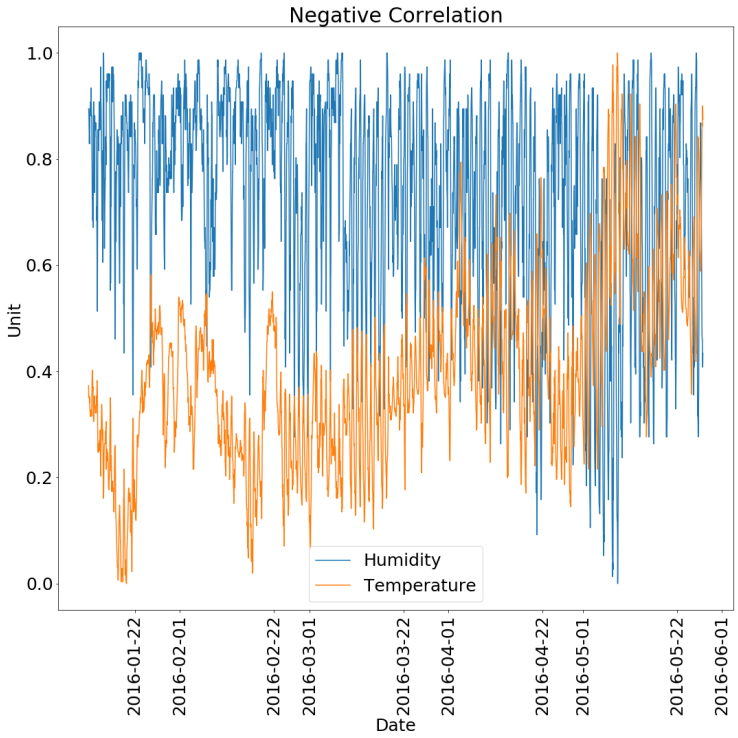

Figure 7: Example showing the negative correlation between Humidity and Temperature

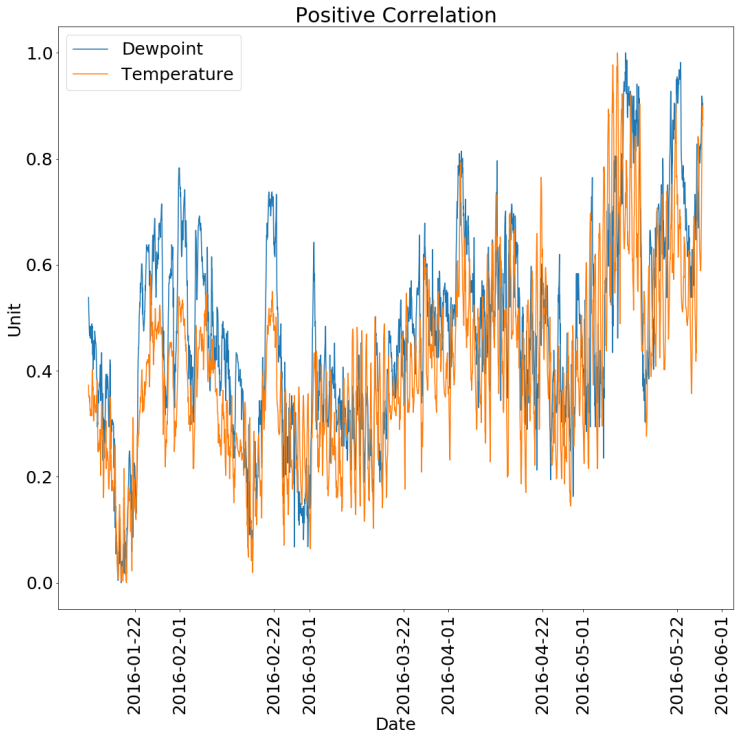

Figure 8: Example showing the positive correlation between Dewpoint and Temperature

Random Forest Regressor has a feature importance function that determines the contribution for each feature when doing regression on the usage of appliances (Figure 9). The two most important features are air pressure and Dew-Point temperature.

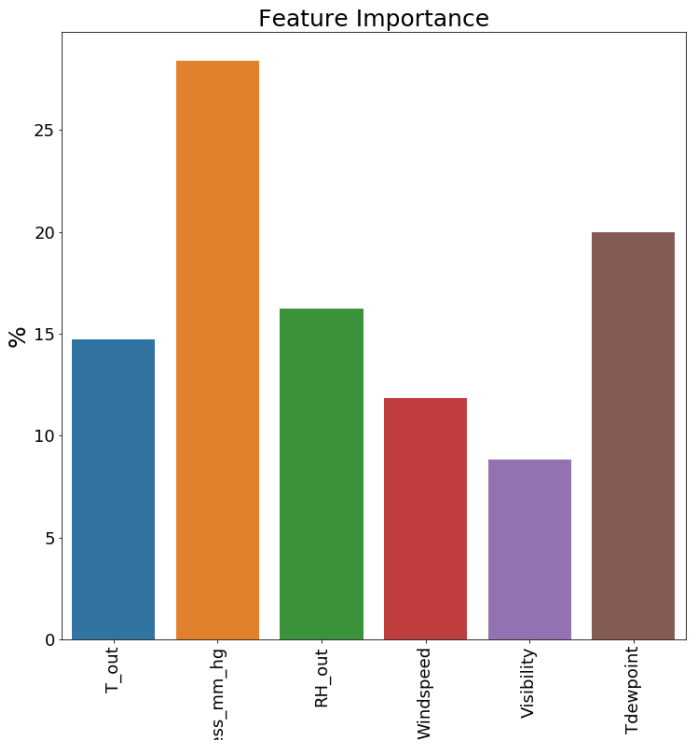

Figure 9: Feature importance given by Random Forest function

Table 2 shows dataframe consisting of variables such as T_out and their respected attributes. Count represent the number of samples of the variables. Mean, standard deviation (std), minimum value (min), quantiles and maximum value (max) are also given. As it is shown in Table 2, the attributes have different range which make the modeling of the solution difficult. Therefore, a pre-processing of the samples of the attributes is essential.

Table 2: Calculated attributes for each variable in Belgium Dataset.

	T_out	Press_mm_hg	RH_out	Windspeed	Visibility	Tdewpoint	HouseConsum
count	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000
mean	7.411665	755.522602	79.750418	4.039752	38.330834	3.760707	0.028194
std	5.317409	7.399441	14.901088	2.451221	11.794719	4.194648	0.028995
min	-5.000000	729.300000	24.000000	0.000000	1.000000	-6.600000	0.002778
25%	3.666667	750.933333	70.333333	2.000000	29.000000	0.900000	0.013889
50%	6.916667	756.100000	83.666667	3.666667	40.000000	3.433333	0.016667
75%	10.408333	760.933333	91.666667	5.500000	40.000000	6.566667	0.027778
max	26.100000	772.300000	100.000000	14.000000	66.000000	15.500000	0.308333

5.2 KAGGLE DATASET

A pre-process was made on the features windSpeed, temperature, dewPoint, pressure and visibility where a unit conversion was done. The following formula used for each conversion are listed below. After the conversion, the dataset was transformed from a 1 min period to a 10 min period by taking the mean of every 10 data points.

- WindSpeed, $x_i * 0.44704$, where x_i is windSpeed.
- Temperature, $\frac{(x_i - 32)}{1.8}$, where x_i is temperature.
- DewPoint, $\frac{(x_i - 32)}{1.8}$, where x_i is dewPoint temperature.
- Pressure, $(\frac{x_i}{133.322368}) * 100$, where x_i is air pressure.
- Visibility, $\frac{x_i}{1.609344}$, where x_i is visibility.

Calculating the correlation between each feature in the second dataset yields the heatmap in Figure 10. In both heatmaps shown in Figure 6 and Figure 10, the correlation between temperature and dewpoint are high. Another noticeable similarity of both datasets is the usage of appliances have low correlation with all other features.

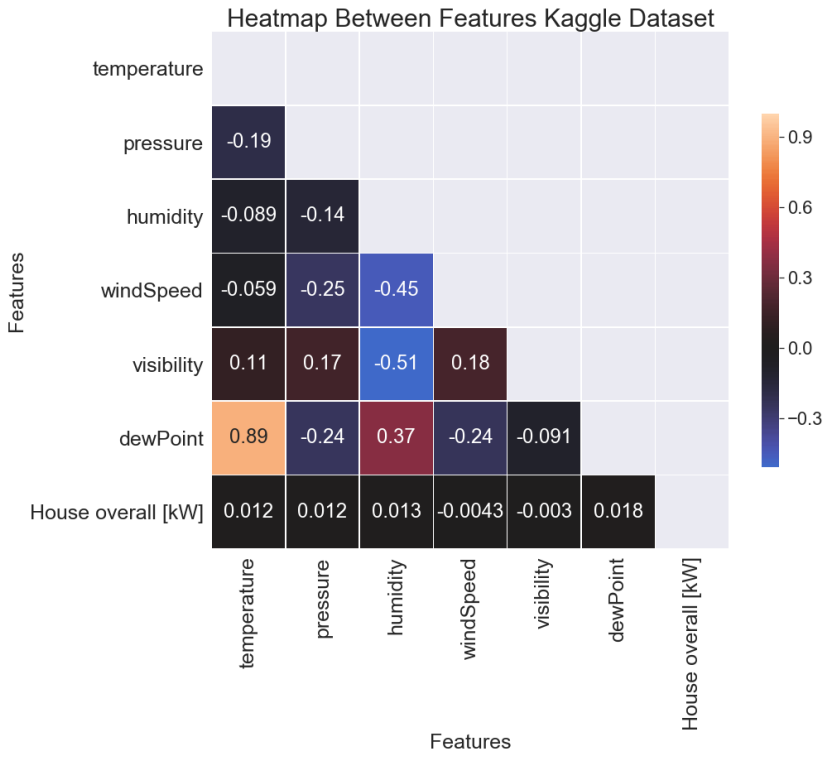

Figure 10: Heatmap of correlation between all features used in Kaggle Dataset

By applying Random Forest Regressors, important features of this dataset was also found which is shown in Figure 9. Similarly, air pressure and dewpoint were found to be important features for the dataset. It has a high similarity to the previous Figure 9. What differs is the importance in air pressure and humidity has a lower value in Figure 11 and the importance in temperature, windspeed and dewpoint has a higher value.

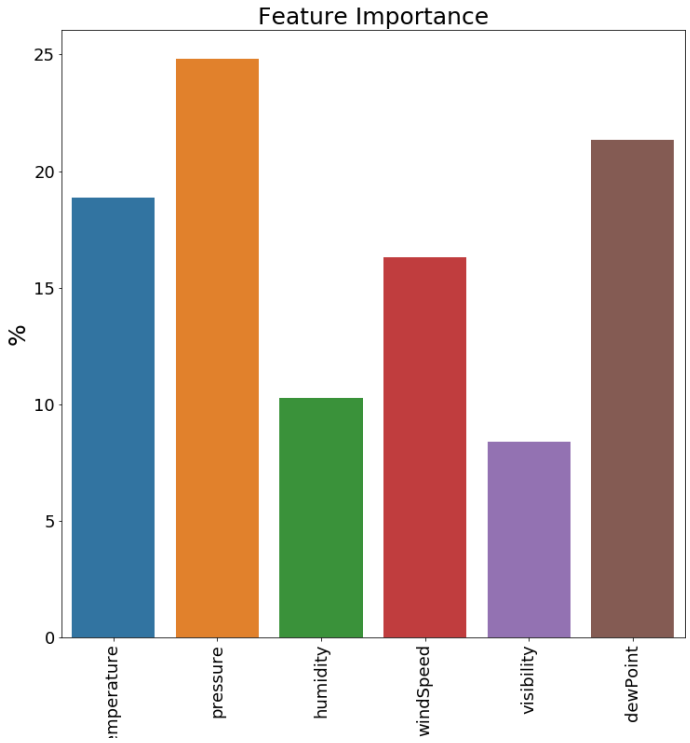

Figure 11: Feature importance given by Random Forest function

Table 3 shows dataframe consisting of variables such as T_out and their respected attributes. Count represent the number of samples of the variables. Mean, standard deviation (std), minimum value (min), quantiles and maximum value (max) are also given. As it is shown in Table 3, the attributes have different range which make the modeling of the solution difficult. Therefore, a pre-processing of the samples of the attributes is essential. Differences between Table 2 and Table 3 are the samples per feature, and its corresponding energy consumption (House Overall [kW]) has a higher maximum value.

Table 3: Calculated attributes for each variable in Kaggle Dataset.

	temperature	pressure	humidity	windSpeed	visibility	dewPoint	House overall [kW]
count	50391.000000	50391.000000	50391.000000	50391.000000	50391.000000	50391.000000	50391.000000
mean	10.412186	762.288910	0.664085	2.972787	5.749824	3.718896	0.858962
std	10.617192	5.921434	0.194233	1.776022	0.998889	10.603960	0.928474
min	-24.800000	739.860846	0.130000	0.000000	0.167770	-32.911111	0.000800
25%	2.094444	758.533632	0.510500	1.636390	5.853317	-4.117222	0.398141
50%	10.183333	762.460205	0.680000	2.646477	6.213712	3.905556	0.613882
75%	19.033333	766.173010	0.836000	3.992067	6.213712	12.661111	1.002376
max	34.288889	781.909304	0.980000	10.241686	6.213712	24.161111	11.224978

5.3 NORMALIZATION

For a RNN to work efficiently, a dataset should not have large numbers or different scales. However, as it can be seen in Table 2 the feature "Press_mm_hg" has its values circulate around 755 while the feature "windspeed" never reaches values above 14. In the case of having two datasets, the scales should match. Shown in the target feature "HouseConsum" in Table 2 and the corresponding feature "House Overall [kW]" in Table 3, the max values respectively are 0.3 and 11. Using min-max normalization, which gives each features a scale ranging from 0 to 1, eliminates the scale problem. Having the two datasets separated while normalizing keeps the information that exists in each feature while giving the same scale. If the normalization was global then the information about Belgium Dataset would blend into Kaggle Dataset. The normalized data can be seen in Table 4 and 5.

Table 4: Calculated attributes for each variable in Belgium Dataset after normalization.

	T_out	Press_mm_hg	RH_out	Windspeed	Visibility	Tdewpoint	HouseConsum
count	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000	19735.000000
mean	0.399089	0.609828	0.733558	0.288554	0.574321	0.468810	0.083179
std	0.170978	0.172080	0.196067	0.175087	0.181457	0.189803	0.094892
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.278671	0.503101	0.609649	0.142857	0.430769	0.339367	0.036364
50%	0.383173	0.623256	0.785088	0.261905	0.600000	0.453997	0.045455
75%	0.495445	0.735659	0.890351	0.392857	0.600000	0.595777	0.081818
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Table 5: Calculated attributes for each variable in Kaggle Dataset after normalization.

	temperature	pressure	humidity	windSpeed	visibility	dewPoint	House overall [kW]
count	50391.000000	50391.000000	50391.000000	50391.000000	50391.000000	50391.000000	50391.000000
mean	0.595919	0.533386	0.628335	0.290263	0.923273	0.641818	0.076457
std	0.179682	0.140824	0.228509	0.173411	0.165216	0.185799	0.082721
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.455152	0.444078	0.447647	0.159777	0.940391	0.504517	0.035400
50%	0.592046	0.537460	0.647059	0.258402	1.000000	0.645089	0.054622
75%	0.741820	0.625758	0.830588	0.389786	1.000000	0.798501	0.089234
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Calculating the correlation between the features in the two datasets require a subsampling of the larger Kaggle Dataset. It is done by using dataframes sample function. Using a randomstate set to 0, number of samples to extract set to 19735. The correlation is presented as a heatmap in Figure 12. What can be concluded from the heatmap is that the correlation of the two datasets are low. Although this correlation is weak, some information can be extracted. This information increases the interest of the study and presents an opportunity for using non-correlating data to create an accurate model.

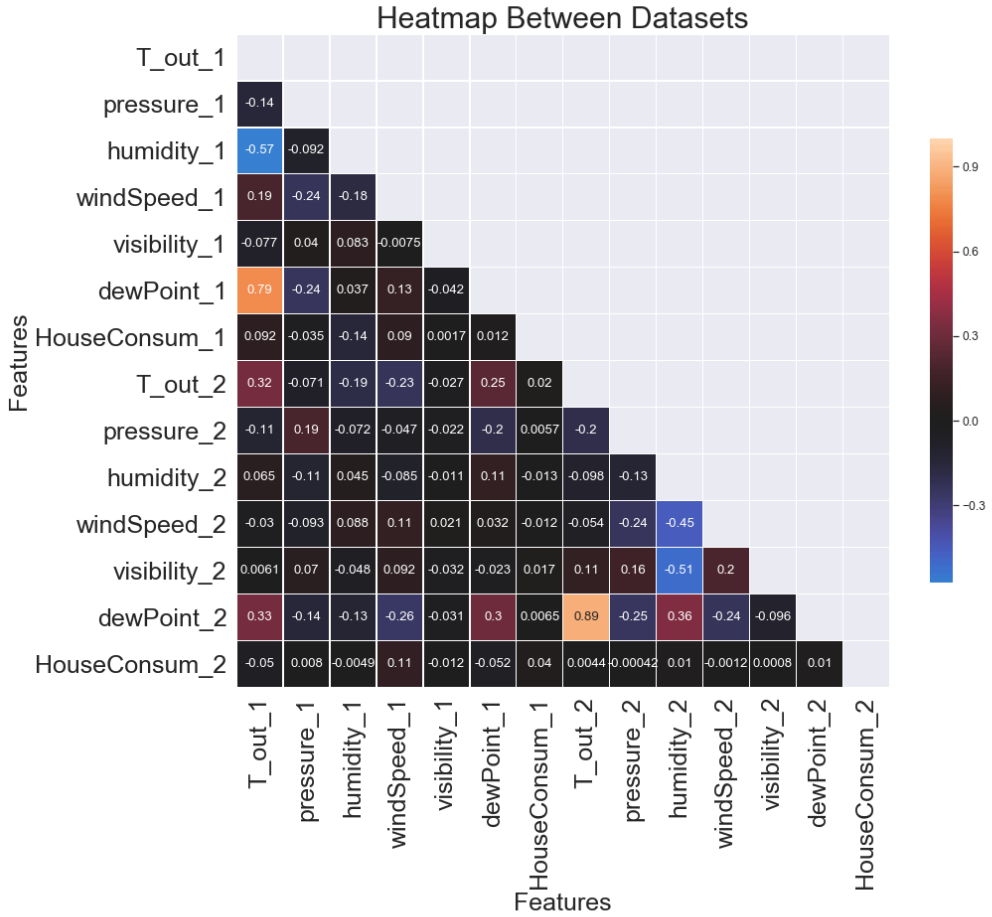

Figure 12: Correlation between the two datasets presented in a heatmap.

6 RESULTS

6.1 EXPERIMENT SETUP

6.1.1 *Hardware*

The experiments has been run on a computer running the windows operating system. The specifications of the computer are as follows:

- GPU: ASUS Nvidia GTX 1060 3 GB
- CPU: AMD Ryzen 5 2600
- RAM: Corsair Vengeance LPX Black 16 GB
- Mother Board: ASUS ROG Strix B450-E Gaming

6.1.2 *Software*

The programming language used is Python. The experiment is run through a virtual environment created with Anaconda using the python version 3.7.6 and the following packages listed below are used. The code has been written and executed in a Jupyter Notebook. The network was trained using the above listed GPU Nvidia GTX 1060 that has 3 GB VRAM with CUDA version 11.0.

- Tensorflow 2.1.0
- Keras 2.3.1
- scikit-learn 0.22.1
- Pandas 1.0.1
- Numpy 1.18.1
- Matplotlib 3.1.3
- Seaborn 0.10.0

6.1.3 *Settings*

The problem is the creation of a model by applying S2S LSTM methodology for accurately predicting the energy consumption of a building through the application of transfer learning. By extracting information of a building with a larger dataset and applying it to a target building. Experiments includes predictions of energy consumption in periods of ten minutes, one hour and two hours. With the data of historical information of appliance use and weather conditions, this time-series regression problem, if successful, can be used as evidence for using transfer learning in energy consumption prediction.

6.2 CASES

There are four cases defined where each case is a unique way of training the model. The idea is to train the model with different amounts of information about Belgium Dataset. The four cases are described bellow. The validation data which was 30% of Belgium Dataset was used for all of the four cases. The cases uses the normalized data that was described in Table 4 and Table 5. Each case uses the evaluation metrics RMSE, R^2, MAE and MAPE.

Case 1 presented in Table 6 uses Belgium Dataset as training set and test set. This gives us an upper-bound in evaluating performance in future cases.

Table 6: Training approach Case 1, training and testing with Belgium Dataset.

	Training	Testing	Validation
Case 1	Belgium Dataset	Belgium Dataset	30 % Of Belgium Dataset

In Case 2 (Table 7), the model only has information about Kaggle Dataset in both training and testing. Given the fact that the distribution of data in these datasets are very different, shown in section 5, this experiment gives us a lower-bound comparison and validation.

Table 7: Training approach Case 2, training and testing with Kaggle Dataset.

	Training	Testing	Validation
Case 2	Kaggle Dataset	Kaggle Dataset	30 % Of Belgium Dataset

In Case 3 (Table 8), the model tries to generalize the information gained in Kaggle Dataset while continuously tests its prediction capabilities against a part of Belgium Dataset. This experiment gives understanding of whether the generalization is good or not and gives feedback to the system in the training phase.

Table 8: Training approach Case 3, training with Kaggle Dataset and testing with Belgium Dataset.

	Training	Testing	Validation
Case 3	Kaggle Dataset	Belgium Dataset	30 % Of Belgium Dataset

In Case 4 (Table 9), the model trains in the same way as Case 2 by only using Kaggle Dataset as train and test, then it is fine-tuned in the last 2 layers (Table 11 and 12) for 1000 epochs. The fine-tuning phase is added due to the models probable enhancement in performance.

Table 9: Training approach Case 4, training with Kaggle Dataset and tuning with Belgium Dataset.

	Training	Testing	Tune Training	Tune Testing	Validation
Case 4	Kaggle Dataset	Kaggle Dataset	Belgium Dataset	Belgium Dataset	30 % Of Belgium Dataset

The Figures 13 and 14, show the distribution of each data split for the Belgium datazet and the Kaggle dataset respectively, the validation set is added in both plots to be able to show how similar it is to other splits. In the Figure 13 the splits show the same distributions. In Figure 14, the splits 1,2,3 and 6, does not contain a significant amount of values above 0.4. However the splits 4 and 5 contain plenty of data reaching above 0.4.

The seventh data split in both Figures 13 and 14 is our validation set where the evaluation metrics for the models are calculated. The validation set consists of data points from the end of the Belgium Dataset, it amounts to 30%. Both Kaggle Dataset and the remaining of the Belgium Dataset is then divided into 6 sub-datasets to be used to calculate a statistical significance. Because the problem has a time constraint, an ordinary cross validation method, where random points are used in each fold, can't be implemented in this problem. Its crucial to conserve the time series, therefore, the folds does not contain random samples but instead each fold increases the amount of training data. Each fold has a unique test set, for the next fold the previous fold's test set is added into the current fold's training set. Visual of how the process proceed can be seen in Table 10.

Table 10: Table shows the workings of the fold training method.

	Sub-Set 1	Sub-Set 2	Sub-Set 3	Sub-Set 4	Sub-Set 5	Sub-Set 6
Fold 1						
Fold 2						
Fold 3						
Fold 4						
Fold 5						

Train-set
Test-set

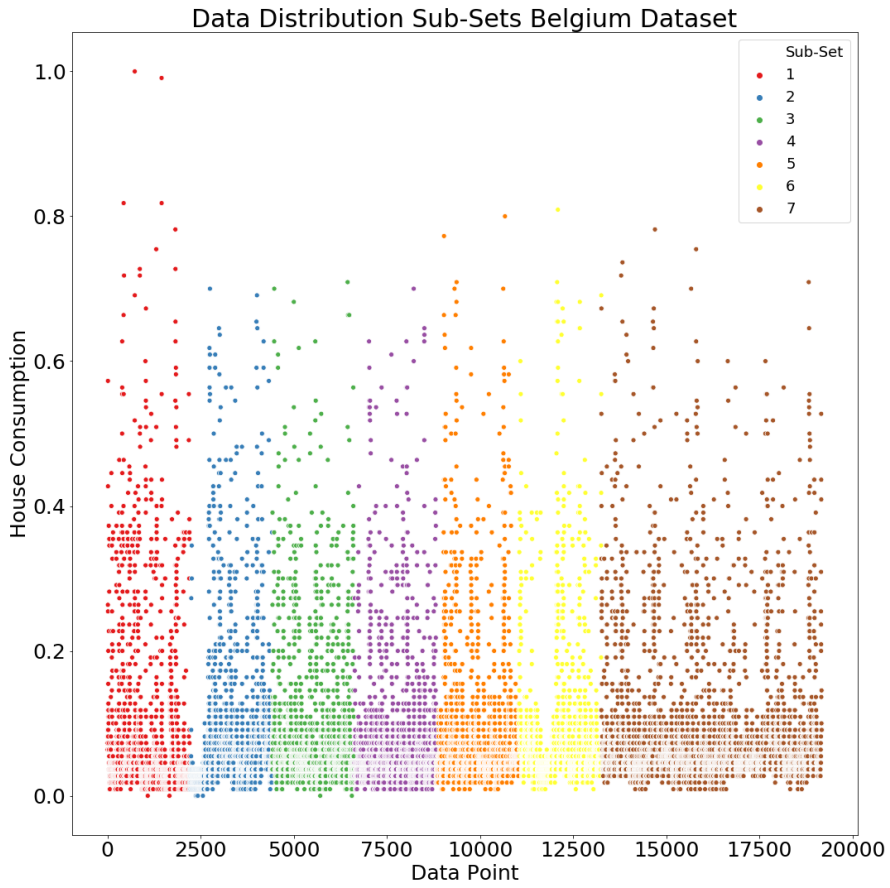

Figure 13: Figure shows a catplot over the Belgium Dataset with last fold (brown) being the validation fold that is used to evaluate the models performance.

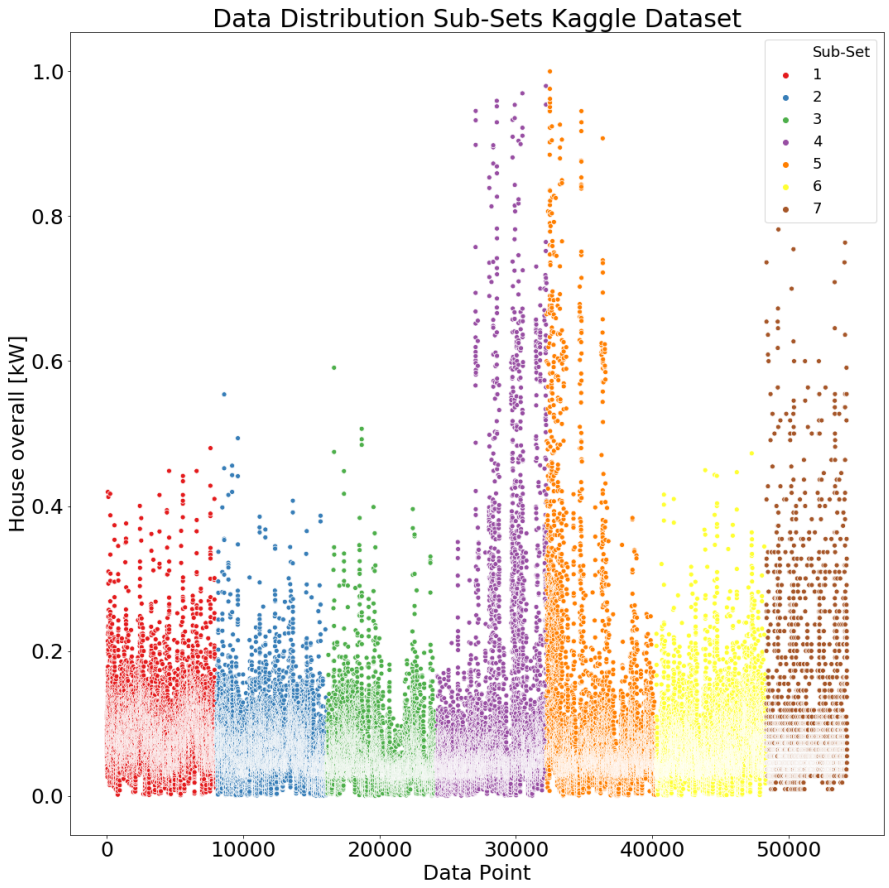

Figure 14: Figure shows a catplot over the Kaggle Dataset with last fold (brown) being the validation fold that is used to evaluate the models performance.

Using TSNE to visualize our data in a 2D space. It converts similarities between data points to joint probabilities and tries to minimize the relative entropy between the joint probabilities of the low dimensional embedding and the high dimensional data. If the plot contain no clusters, it shows that the sub-sets of data is similarly distributed. The TSNE plot showing the sub-sets from the Belgium Dataset has only one obvious cluster, being the brown colored points, while the other colors have data points spread out across the space, seen in Figure 15. The TSNE plot for the Kaggle Dataset has more clusters than the Belgium Dataset. It is visually shown in Figure 16, it is only the purple and red colored points that is clustered together. Seen in previous catplot for the Kaggle Dataset in Figure 14, the data splits 4 and 5 represented as red and purple in the TSNE plot has a pattern that is similar. These mentioned observation is interesting to note because of the potential change in prediction performance. That the different folds differ more from each other when the test subset is represented by the clustered subsets.

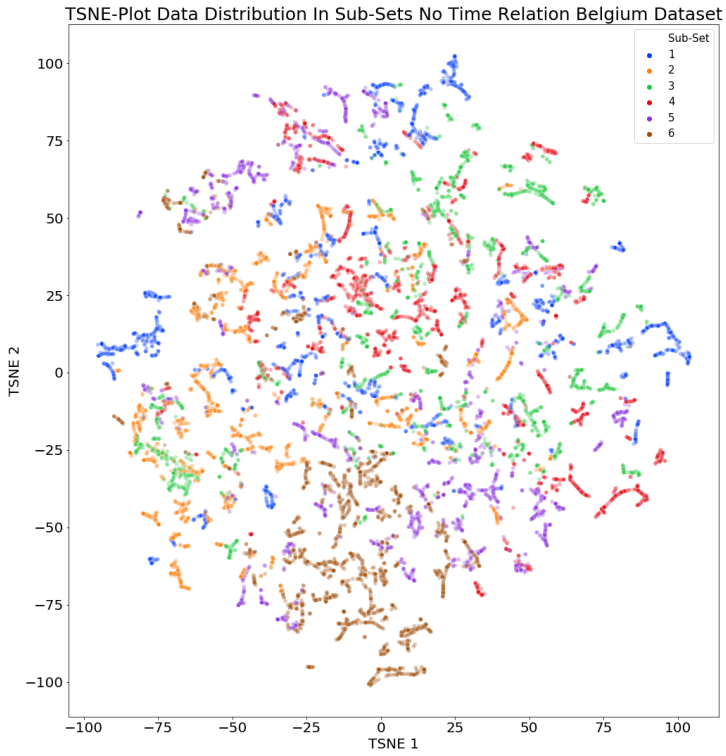

Figure 15

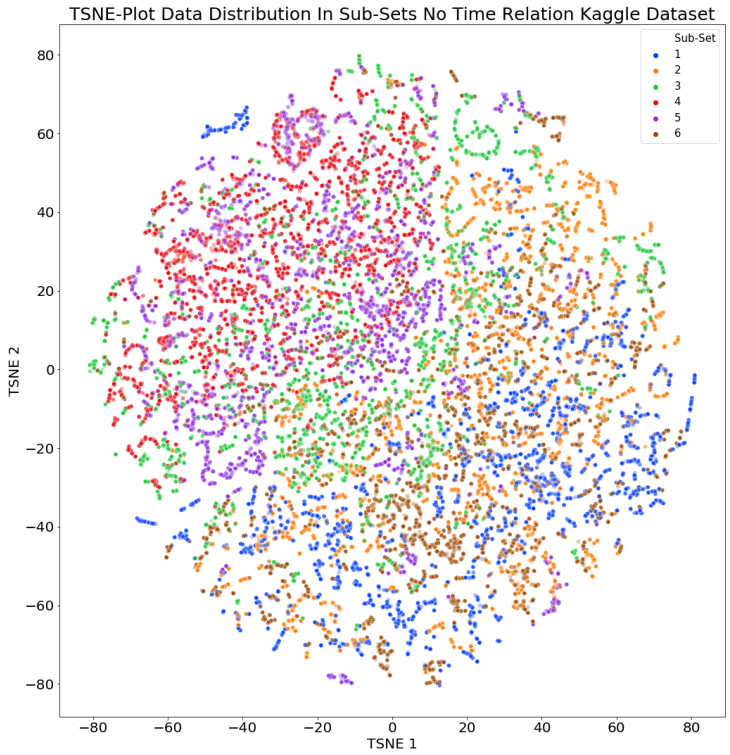

Figure 16

6.4.1 *Loss Functions*

Choosing loss function was made by creating two arrays that has 20000 data points each, where one is ranging from (-10000, 10000), and the other having a constant value of 100. A global normalization to the array was added to make them range between (0,1). The error between the two arrays are calculated using RMSE, MAE, LogCosh and MSE. In the Figure 17 the errors are visualized, the x axis is prediction and the left y axis is the MAE and RMSE loss and right y axis is MSE and LogCosh loss. The equation used are presented in Section 3.2.

The optimizer tries to find the weights that gives the smallest error, which is at $x = 0$ in Figure 17. When the optimizer converges towards the smallest error using the RMSE function, the gradient is constant, which makes the optimizer unreliable as it approaches a local minimum causing the optimizer to overshoot. Using the MSE which provides a smoother curve with a changing gradient, helps the optimizer to find the local minimum. The LogCosh function has a similar shape to MSE, however, the gradient is more flat than the one MSE produces, causing the training to be longer in order to reach the same minimum as MSE. R^2 is not shown in the plot of loss functions in Figure 17 because it ranges from minus infinity making it hard to be shown with other loss functions on the same plot. It is designed to provide one score for one array.

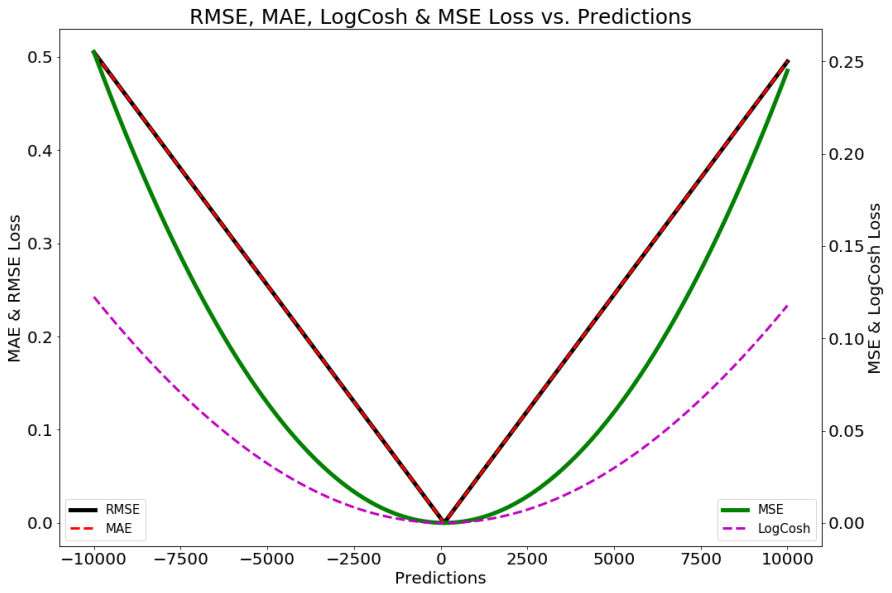

Figure 17: RMSE, MAE, LogCosh & MSE loss for predictions, visualized.

6.4.2 *Activation functions*

In order to optimize the model we have tested a few different combinations of activation functions. These tests is done through four combinations with Relu or Tanh as activation function on the hidden layers while the output layer is tested with Linear and Sigmoid. These combinations were chosen due to the graphical movement of each function. They represent the movement of either a stable derivative, such as the Relu and Linear functions, or a changing derivative, Tanh and Sigmoidal functions. These tests were done with a simple model of one hidden layer and one output layer to emphasize more on the activation function. Two models are chosen to be further investigated with more complex structured models. Comparing the two models, shown in Figure 18, using Sigmoid as an activation function, the version that uses Tanh is preforming the best in all three time intervals. The two models that uses the Linear function, the version with Relu preforms the best in two out of the three time intervals.

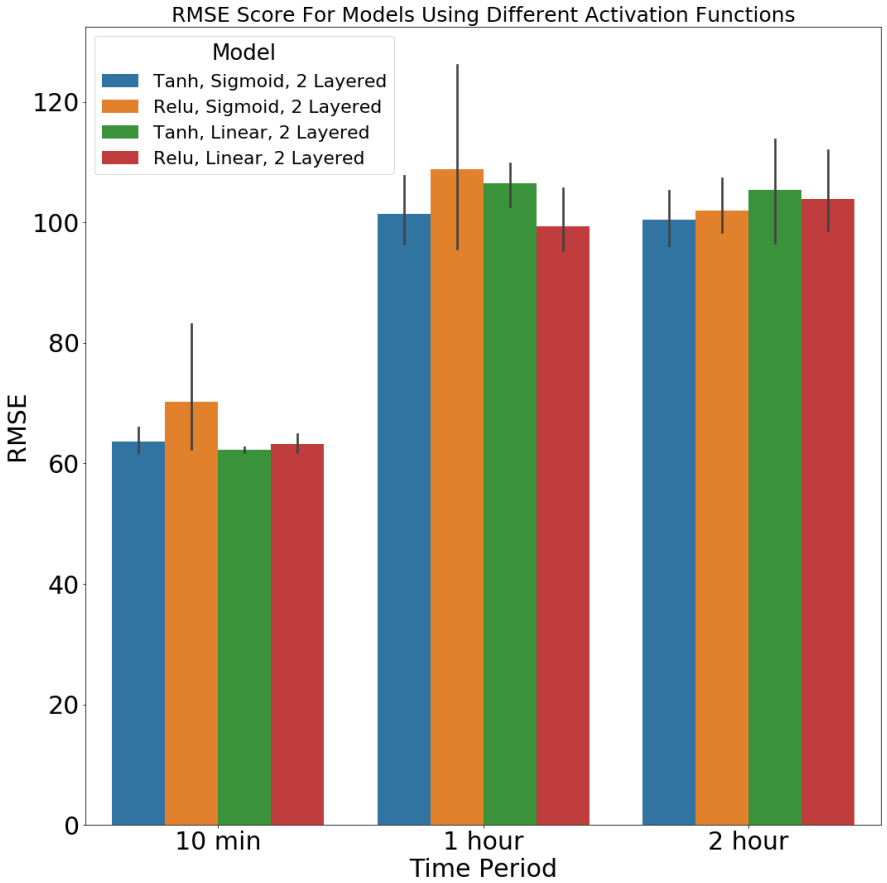

Figure 18: Figure showing the RMSE score for models using Tanh or Relu activation function in hidden layer together with Sigmoid or Linear activation function in output layer.

The performance of evaluations of the activation funcitons are shown in Figure 19 and Figure 20. In both figures, the results show stability in the case of the Sigmoid function. The Sigmoidal model with two layers show a hint of instability. The Linear activation function on the other hand do not have a great stability as it can be seen in the case of ten-minute tests. However, the Linear function with 4, 5, and 6 layers show great performance in the case of two-hour which may overlooked due to the instability of the results. Since the Sigmoidal functions is stable, it will be used in models and tests.

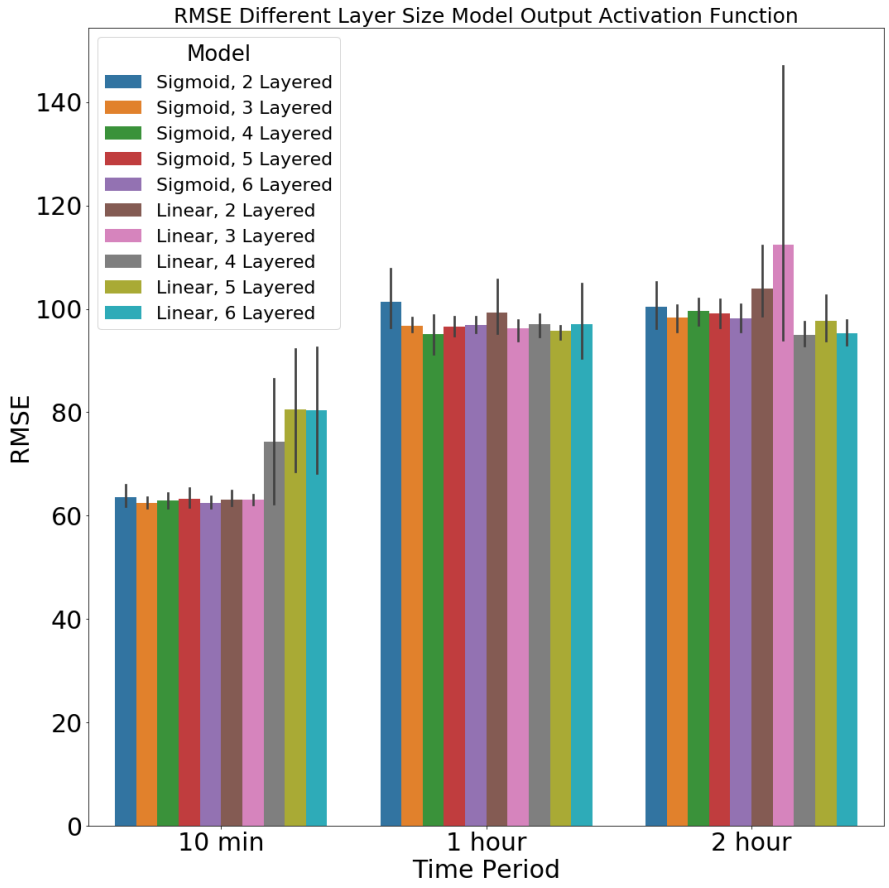

Figure 19: Figure showing the RMSE score for models using different amount of layers using tanh in hidden and Sigmoid in output or relu in hiddden and Linear in output activation function in output layer.

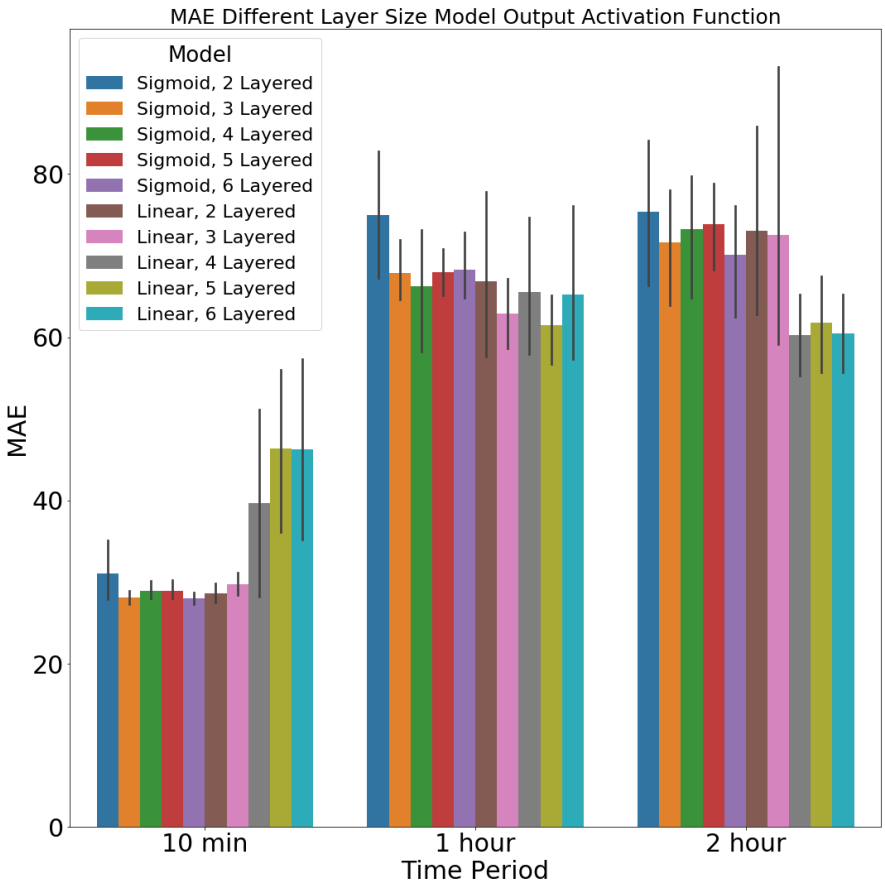

Figure 20: Figure showing the MAE score for models using different amount of layers using tanh in hidden and Sigmoid in output or relu in hiddden and Linear in output activation function in output layer.

6.5 MODEL STRUCTURE

6.5.1 *8L-Model*

The 8L-Model consist of the model chosen from previous experiments, conducted in Section 6.4.2, and setting the amount of nodes in each hidden layer to seven. In addition to the seven hidden layers, there are a batch normalization layer and a dropout layer with a rate of 0.5, added before the output layer of the model. The 8L-Model sums up to a total of 5888 number of parameters. The structure can be seen in Table 11.

Table 11: This is the 8L-Model structure

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Input (Bidirectional)        (None, 1, 14)             840
_________________________________________________________________
Bi-LSTM-1 (Bidirectional)    (None, 1, 14)             1232
_________________________________________________________________
Bi-LSTM-2 (Bidirectional)    (None, 1, 14)             1232
_________________________________________________________________
Bi-LSTM-3 (Bidirectional)    (None, 1, 14)             1232
_________________________________________________________________
Bi-LSTM-4 (Bidirectional)    (None, 1, 14)             1232
_________________________________________________________________
Normalization (BatchNormaliz (None, 1, 14)             56
_________________________________________________________________
dropout_1 (Dropout)          (None, 1, 14)             0
_________________________________________________________________
Output (LSTM)                (None, 1)                 64
=================================================================
Total params: 5,888
Trainable params: 5,860
Non-trainable params: 28
```

The training settings for the 8L-Model are listed below.

- Optimizer: ADAM
- Optimizer Loss Function: MSE
- Optimizer Learn Rate: 0.001
- Early Stop Patience: 100 Epoch
- Batch Size: 1000
- Epoch Size: 1000

6.5.2 *11L-Model*

The model was also tested with a larger amount of parameters. The number of nodes in the hidden layers was raised be 7. Then, the number of nodes was decreased by 7 while adding 3 extra hidden layers. For the purpose of testing a model with larger amount of trainable parameters. Increasing the number of nodes in hidden layers by seven each layer, then decrease the number of nodes by seven, adding three extra hidden layers. The 11L-Model sums up to a total of 69056 number of parameters. The structure is shown in Table 12. Like the 8L-Model there are a batch normalization and a dropout layer, with a rate of 0.5, before the output layer.

Table 12: This is the 11L-Model structure

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Input (Bidirectional)        (None, 1, 14)             840
_________________________________________________________________
Bi-LSTM-1 (Bidirectional)    (None, 1, 28)             3248
_________________________________________________________________
Bi-LSTM-2 (Bidirectional)    (None, 1, 42)             8400
_________________________________________________________________
Bi-LSTM-3 (Bidirectional)    (None, 1, 56)             15904
_________________________________________________________________
Bi-LSTM-4 (Bidirectional)    (None, 1, 56)             19040
_________________________________________________________________
Bi-LSTM-5 (Bidirectional)    (None, 1, 42)             13104
_________________________________________________________________
Bi-LSTM-6 (Bidirectional)    (None, 1, 28)             6384
_________________________________________________________________
Bi-LSTM-7 (Bidirectional)    (None, 1, 14)             2016
_________________________________________________________________
Normalization (BatchNormaliz (None, 1, 14)             56
_________________________________________________________________
dropout_2 (Dropout)          (None, 1, 14)             0
_________________________________________________________________
Output (LSTM)                (None, 1)                 64
=================================================================
Total params: 69,056
Trainable params: 69,028
Non-trainable params: 28
```

The train settings for the 11L-Model are listed below.

- Optimizer: ADAM
- Optimizer Loss Function: MSE
- Optimizer Learn Rate: 0.0005
- Early Stop Patience: 100 Epoch

- Batch Size: 1000
- Epoch Size: 1000

Using the model structures presented in Section 6.5, with each models setting, training them accordingly to each case presented in Section 6.2. The scores in the following section has been converted back from its normalized values to the original values, in order to be comparable to the GBM model from "Data driven prediction models of energy use of appliances in a low-energy house" [39]. Each training of the models uses the same fold system presented in Section 6.3.

Studying the loss function after the ten-minute period training of the 11L-Model, there was an instability in the training. The first 300 epochs in the early folds of the training phase showed an instability that led to an early stop, preventing the 11L-Model from learning anything, resulting in bad results. This was not an issue for the 8L-Model. Training a model with more parameters with such small amount of data in the early folds, causes this instability. By increasing the early stop patience for the 11L-Model from 100 to 300, the model has enough time to overcome the instability and to learn as the 8L-model. This was a problem occurring during the training of the 11L-Model in all the four cases predicting for the ten-minute period.

Given the knowledge from previous Chapter 5, there was small or no correlation between the variables of the dataset could e the cause of the bad results in the case of the model of Case 2. However results further down proves otherwise. Surprisingly, the Kaggle Dataset contains information that the model can use to generalize and score good results in the evaluation metrics. However, the models could not for the first 3 folds, predict any values that reached above 0.4. It is not until fold 4 and 5 where the model can predict any high values. Referring back to the distribution of the data that was presented in previous section 6.3 in Figure 14, it can be seen that the first splits contains no data valued above 0.4. It is not until the later splits that the model is introduced with higher values. This was also a problem for the models when trained accordingly to Case 3. When introducing the transfer learning from Case 4, the models did improve the predictions in the early folds.

6.6.1 *Case 1*

In this case the models has limited amount of data to work with. Both models perform good predicting ten-minute period with a small error bar. When studying the loss during training, for the early folds with small amount of data, it was seen that the 11L-Model stopped before it stabilized. Therefore an increase in the early stop patience from 100 to 300 provided better results as it gave the model time to stabilize. As the time period increase the results become worse for both models. In Figure 21, for the one-hour and two-hour intervals, the error bar is increasing for both models. This is due to the lack of data. The 8L-Model seem to be able to handle the problem better than the 11L-Model, mainly because it has less parameters. The MAE score of the models, follows the same pattern that can be found in the Figure 21.

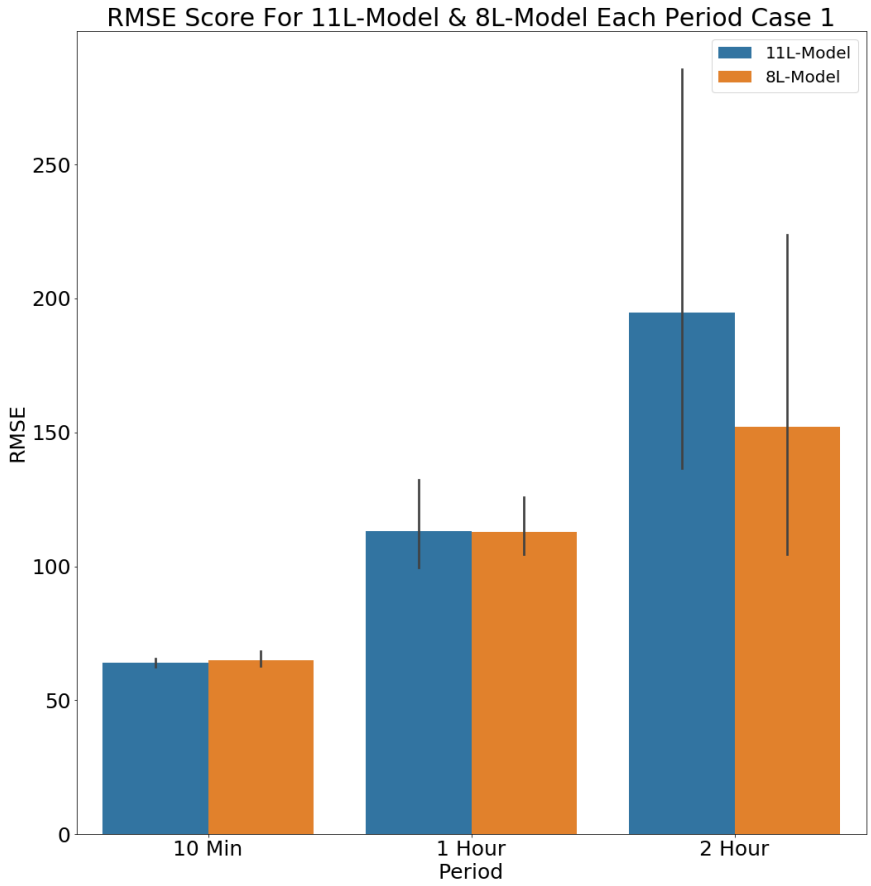

Figure 21: RMSE score of 11L- and 8L-Model showing an error bar calculated from each fold. Models using Case 1.

Investigating the ten-minute period results from the fifth fold where the models have all data available for training and comparing the results. The prediction scores, RMSE, R-Squared, MAE and MAPE, in Table 13, provide the information that the 11L-Model has a slightly better performance in predicting the higher values but does not predict the majority of the data as good as the 8L-Model. However the results are not significantly different. The visual of the 11L-Models prediction in Figure 22 compared to the visual of the 8L-Models prediction in Figure 23 looks similar. Both models predict no values higher than 0.5.

Table 13: Table showing results from fold 5, predicting ten-minute timestep. Training the models according to approach Case 1.

Timestep/Evaluation	RMSE	R^2	MAE	MAPE
11L-Model	62.09	0.54	26.44	22.68
8L-Model	63.34	0.53	26.20	20.98

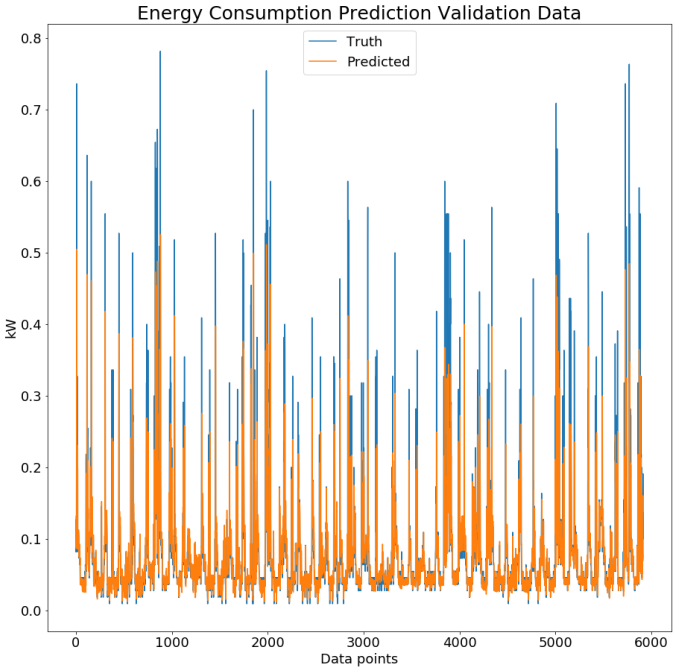

Figure 22: Visual of the predicted value from the 11L-Model trained with Case 1 and predicting ten-minute timestep.

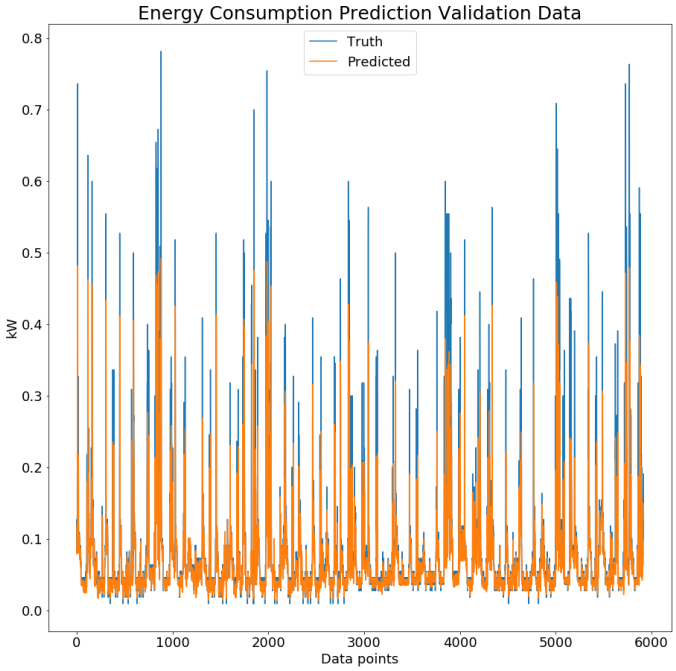

Figure 23: Visual of the predicted value from the 8L-Model trained with Case 1 and predicting ten-minute timestep.

6.6.2 *Case 2*

The fact that Case 2 uses a dataset with no relation to the target dataset, gives the assumption of poor performance from the two models. However, the results proves otherwise. The 11L-Model suffered from the same unstable behavior as it did in Case 1 6.6.1, when the 11L-Model was trained in early folds with small amount of data. It was in need of an increase of early-stop patience from 100 to 300 for the ten-minute prediction. Both 11L- and 8L-Model succeeded to find information in Kaggle Dataset that made it possible to generalize for Belgium Dataset, the target dataset. Compared to previous results from Case 1 6.6.1, it shows that the performance for both the one-hour and two-hour improved. The error bar for one-hour is almost eliminated and the average is now below 100 in RMSE score, as it can be seen in Figure 24. The two-hour error bars show a significant statistical error rate however, it is not as large error as the error given in Case 1 6.6.1. The MAE score for the models, follows the same pattern seen in the Figure 24, as found in Case 1 6.6.1 where the models MAE score pattern is similar to the RMSE pattern.

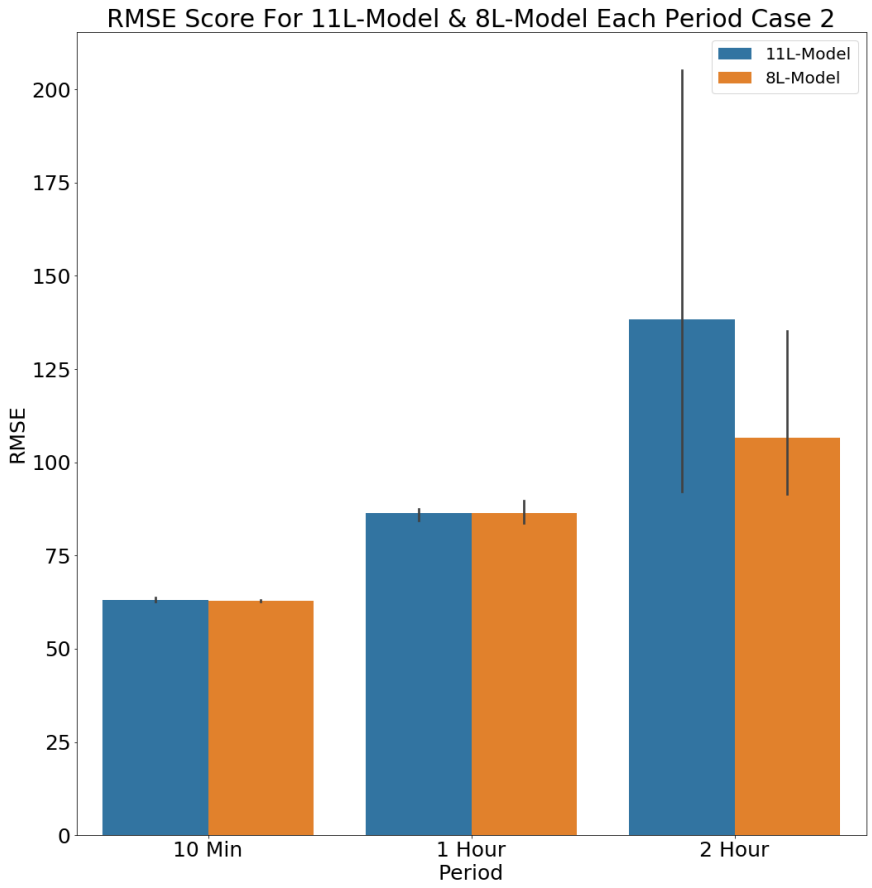

Figure 24: RMSE score of 11L- and 8L-Model showing an error bar calculated from each fold. Models using case 2.

The prediction plots of the 11L-model of Figure 25 and that of 8L-Model of Figure 26 show almost identical performances. The peak located at x value 4000 show a clear example where the 8L-Model predict higher than the 11L-Model.

Table 14: Table showing results from fold 5, predicting ten-minute timestep. Training the models according to approach Case 2.

Timestep/Evaluation	RMSE	R^2	MAE	MAPE
11L-Model	62.71	0.54	28.26	26.54
8L-Model	62.67	0.54	28.54	27.51

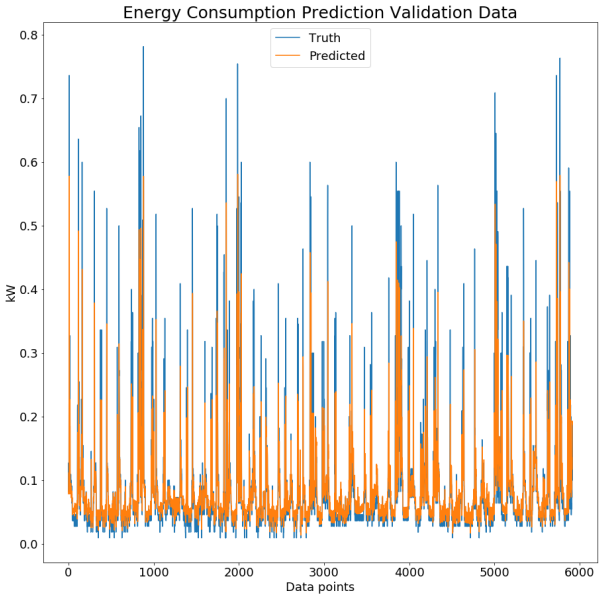

Figure 25: Visual of the predicted value from the 11L-Model trained with Case 2 and predicting ten-minute timestep.

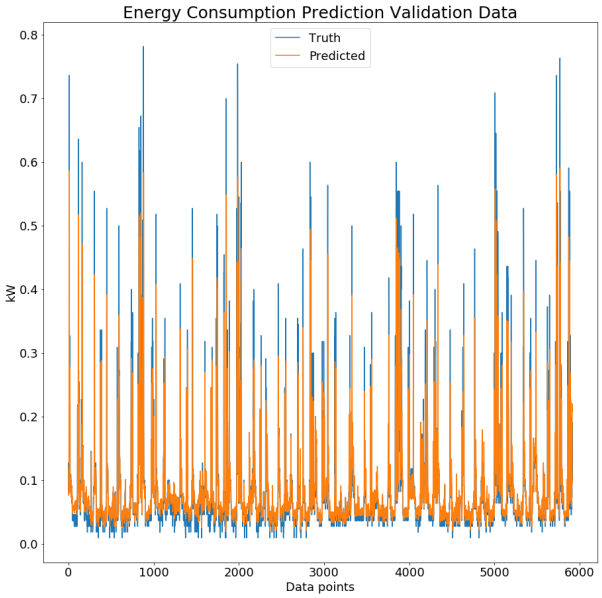

Figure 26: Visual of the predicted value from the 8L-Model trained with Case 2 and predicting ten-minute timestep.

6.6.3 *Case 3*

The models in Case 3 performance compared to previous results from Case 1 6.6.1 and Case 2 6.6.2 has increased. The stability increased for the models in the two-hour time period while also their average. The early-stop patience is once again increased from 100 to 300 as the 11L-Model is unstable for the ten-minute prediction, and for the early folds with small amount of data. The RMSE score for both models in the two-hour interval, shown in Figure 27, has a shorter error and their averages decreased significantly compared to previous Case 2. Providing the models with more data allows the models to learn form the Kaggle dataset and to generalize towards the Belgium dataset. The expectation slightly improved compared to the Case 2 because the models were provided with information of the target dataset. However, since Kaggle Dataset actually provided good information and also providing information from target dataset better results was expected for the lower time periods. Once again the MAE score pattern is following the same pattern given by the RMSE score in Figure 27.

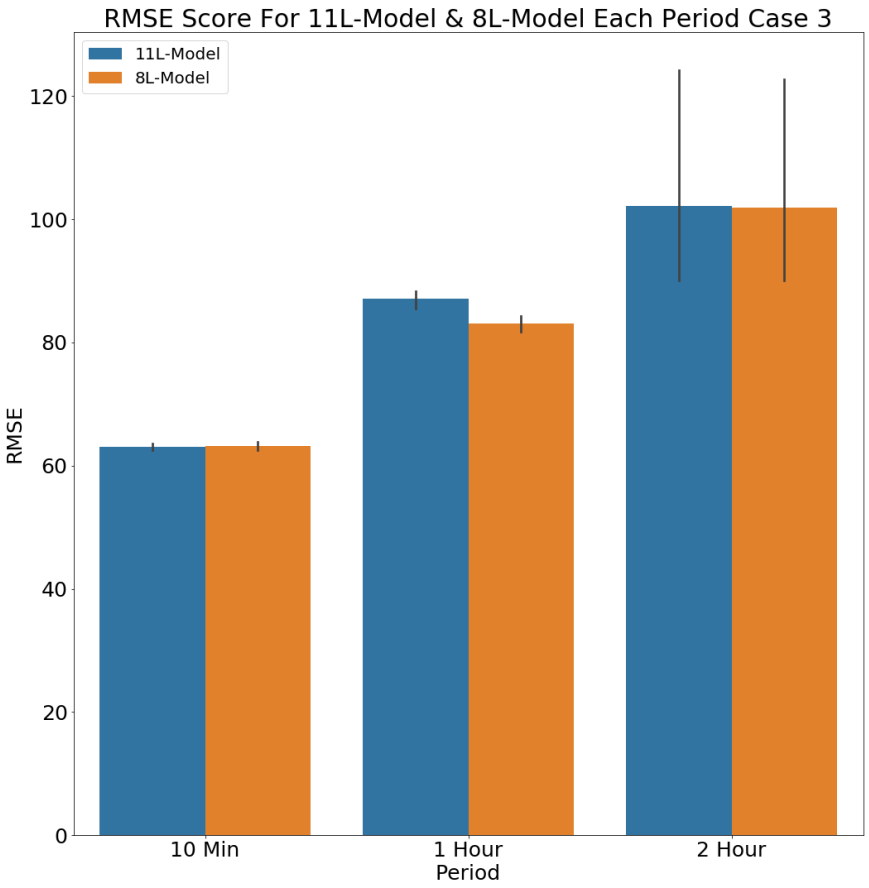

Figure 27: RMSE score of 11L- and 8L-Model showing an error bar calculated from each fold. Models using Case 3.

The visual comparison of the fifth fold in Figure 28 and Figure 29 show little to almost no difference between them. Table 15 indicates a slight increase in performance. Visually, the Case 3 performs similarly to Case 2 but do have a slight edge when it comes to larger targets since it do reach the 0.6 values in the 11L-Model.

Table 15: Table showing results from fold 5, predicting ten-minute timestep. Training the models according to approach Case 3.

Timestep/Evaluation	RMSE	R^2	MAE	MAPE
11L-Model	63.89	0.52	34.38	37.37
8L-Model	62.93	0.53	28.53	27.40

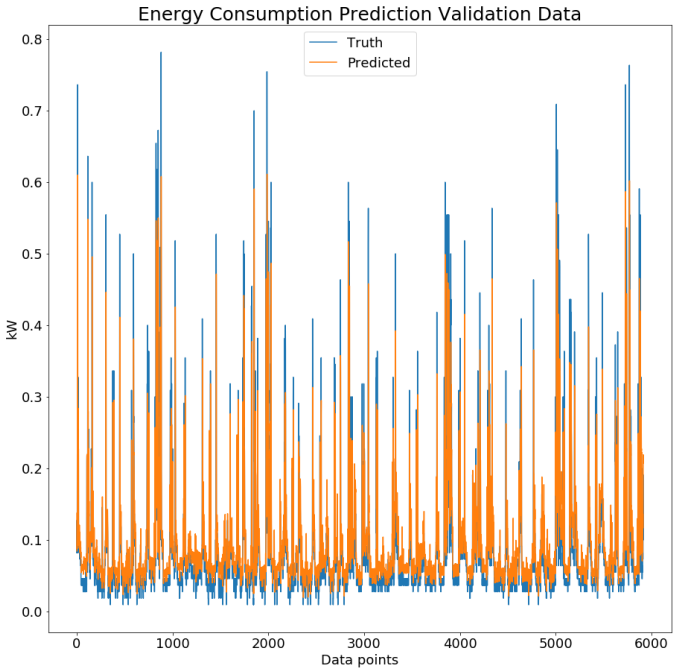

Figure 28: Visual of the predicted value from the 11L-Model trained with Case 3 and predicting ten-minute timestep.

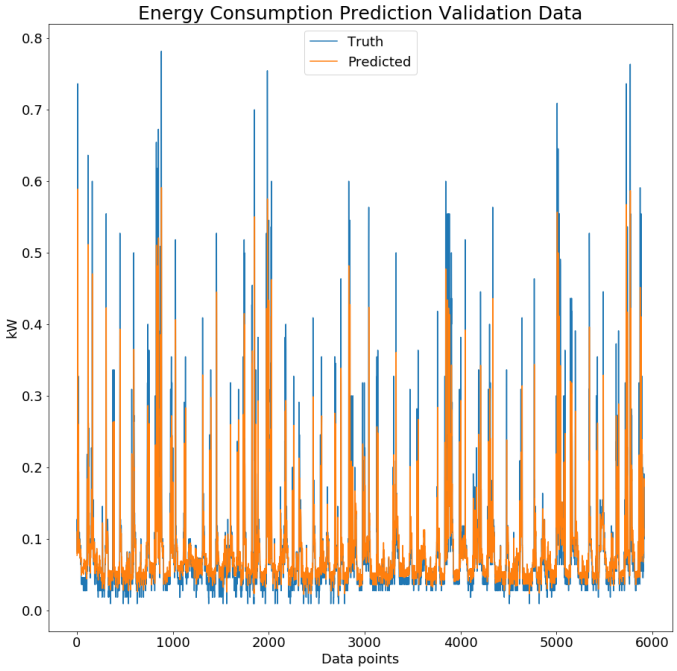

Figure 29: Visual of the predicted value from the 8L-Model trained with Case 3 and predicting ten-minute timestep.

6.6.4 *Case 4*

Training the models according to Case 4 should utilize the data learned from the larger Kaggle Dataset better together with information given from the target Belgium Dataset. As for the previous three cases, the 11L-Model needed an increase of the early-stop patience, for the ten-minute period. The less amount of data given in the early folds required longer learning time before the model stabilized. Compared to previous Case 3 6.6.3, the 8L-Model benefits more from the transfer learning than the 11L-Model. The results are almost indistinguishable from Case 2 6.6.2 and Case 3 6.6.3. It was expected that Case 4 would provide the best results out of all four cases. The transfer learning is composed of three components. First the training of the whole network on the larger Kaggle Dataset. The network is then freezed in all layers except the dropout layer and output layer. It is re-trained on the smaller target Belgium Dataset, then all layers are unfreezed except the batch normalization layer. The last training occurs using a learning rate of 0.000001. Either the transfer learning method implemented is too simple or the data does not contain information that previous cases could not find. The MAE pattern given by this case, follows the RMSE pattern found in Figure 30.

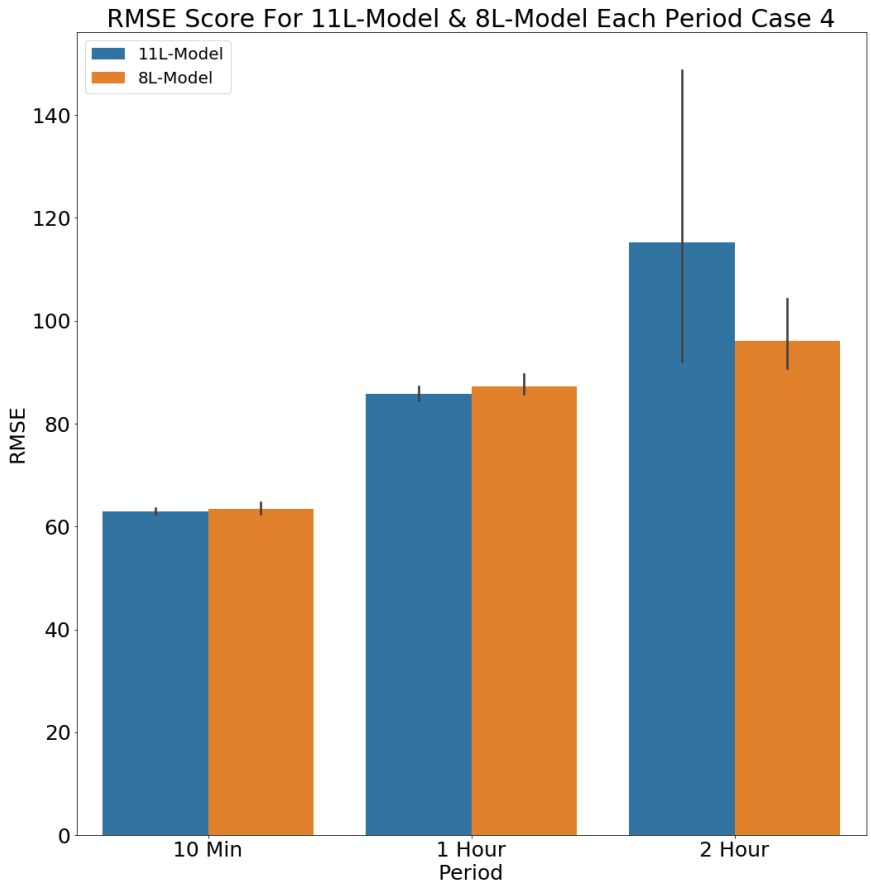

Figure 30: RMSE score of 11L- and 8L-Model showing an error bar calculated from each fold. Models using Case 4.

The fifth fold of Case 4 is presented in Table 16 which indicates that the 11L-Model preform good at the larger values but do not capture the overall structure of the target values comparing to the 8L-Model. Figure 31 and Figure 32 show a slight difference in the prediction of the larger values and lower values. The 11L-Model do have a better performance in larger values but do have more difficulty in addressing the lower values. The 8L-Model produces the opposite results. Visually it is hard to differentiate between the models on the lower values end but easier for the larger values. The 11L-Model has the best high value prediction. It minimizes the error given in the prediction of the peaks but may overshoot on some of the lower values.

Visually, Case 4 11L-Model produces the best results, it has a more stable coverage of all peaks while keeping coverage of the lower predicted values. Case 3 11L-Model, Figure 28, has a good coverage of the peaks but performs poorly on the low variables. Both model of Case 1 shown in Figures 22 and 23 provide the best coverage of the lower values but lacks in peak predictions.

Table 16: Table showing results from fold 5, predicting ten-minute timestep. Training the models according to approach Case 4.

Timestep/Evaluation	RMSE	R^2	MAE	MAPE
11L-Model	63.35	0.53	29.97	28.73
8L-Model	65.50	0.49	28.91	23.54

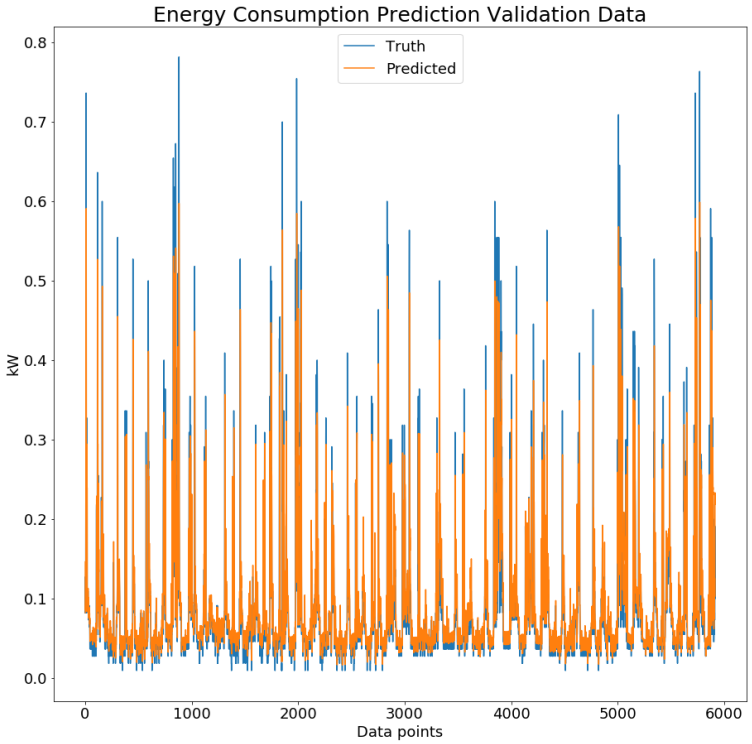

Figure 31: Visual of the predicted value from the 11L-Model trained with Case 4 and predicting ten-minute timestep.

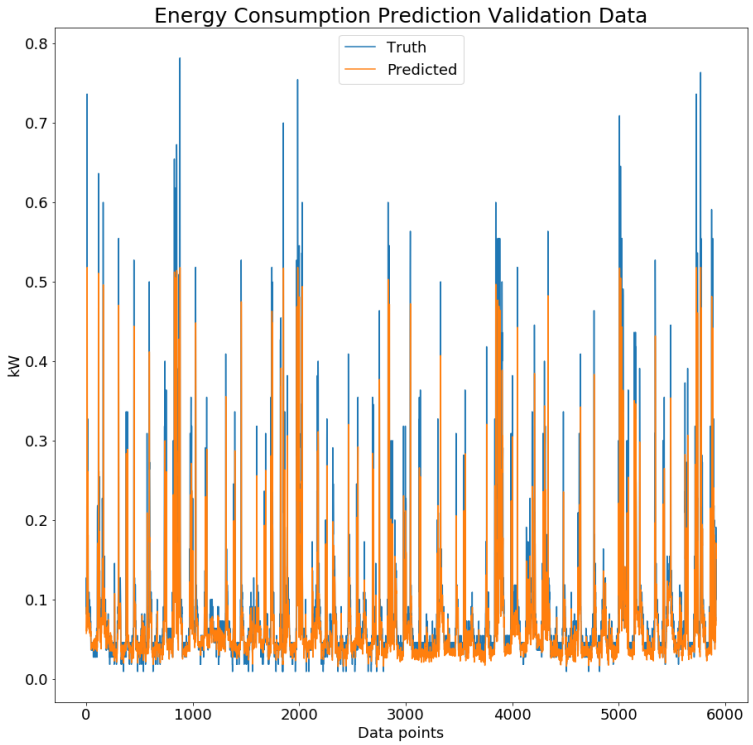

Figure 32: Visual of the predicted value from the 8L-Model trained with Case 4 and predicting ten-minute timestep.

6.6.5 *Summation of the cases.*

When comparing the results given from the models trained on each case for the prediction of ten-minute periods, it is hard to see major differences. As seen in Figure 33, the RMSE score in the ten-minute period is the same. It is the same for the MAE score shown in Figure 34. There are not huge differences when comparing the models performance fold-wise either.

The 11L-Model for ten-miunte period when trained according to Case 1 has larger errors in the early folds compared to the later folds. For the other three cases the erorr increase in the later folds as we kept adding more data from the larger Kaggle dataset. Interesting pattern could be observed in the loss graphs. The loss graph after training yielded 11L-Model, Case 1, had an instability for the first 400 epochs. This instability was lower for the other three cases, but, still in need of an increase in early-stop patience, from 100 to 300, as mentioned earlier in the results 6.6. Without the increase of patience the first folds would not provide results close to current, in fact sometimes the model would not learn at all and only predicted a straight line. The instability in loss graph could be found for the smaller 8L-Model, however, it was not affecting the model to a point where the early-stop patience had to be increased. Case 4 for the 8L-Model had a problem where fold 4 and fold 5 gave worse results than the three first folds.

Increasing the time period to one-hour the models, when trained according to Case 1, starts to fall behind the models trained accordingly to the other three Cases. Results presented in Figure 33, for the one-hour period, Case 1 for both 8L-Model and 11L-Model falls behind in the prediction scores comparing to the other cases. The differences for the other three cases can only be seen when looking fold-wise on the results. The difference is found in the MAE score fold-wise. When the time period increased the models struggle to predict high values. The prediction performance improves when the data contains the majority of the values as we could observe when we applied the MAE to each fold. This is also reflected in the graph of the predictions in Figure 34. For each fold the MAE score is getting lower, however, when reaching the third fold there is an increase of error and this pattern also exists in the RMSE score, can be observed in Figure 33. Studying the loss graph for the 11L-Model in the one-hour period, the line is jagged for the Case 1. The other Cases provides a smoother line. The 8L-Model does not show the same jagged line that was found in the 11L-Models loss graph when trained accordingly to Case 1.

For the largest time period, two-hour, the differences between the models becomes larger. The 8L-Model provides the best average of RMSE and MAE. However, the 11L-Model provides the best MAE score in one fold. When the complexity increased it was expected that the larger 11L-Model would perform better than the smaller 8L-Model. When studying the results given for each fold, it can be seen that the 11L-Model when given all data, improves to be just as good as the 8L-Model. There is not enough data to train the 11L-Model, in the early folds, to achieve same results as the 8L-Model. The smaller 8L-Model can better generalize towards the smaller amount of data given in the early folds. Although, the results provided from either model is not usable in the sense that it estimates only values that lies in the range of the majority of data points. As it can be seen in the Data Analysis chapter 5 in Table 4, the value of 75% of data points from Belgium Dataset are no higher than 0.081818. Since there is more emphasize on the higher values in this study, the performance is not acceptable.

From the study of the loss graphs given from the training of the 8L-Model and 11L-model, an interesting pattern was found for all time periods for both models, is that Case 3 always provides a test loss higher than its train loss. Another interesting behavior was found for the ten-minute period where the two models 8L-Model and 11L-Model in fold 3 and fold 4 had its test loss to be higher than its train loss, but, for the other folds the test loss was lower than the training loss.

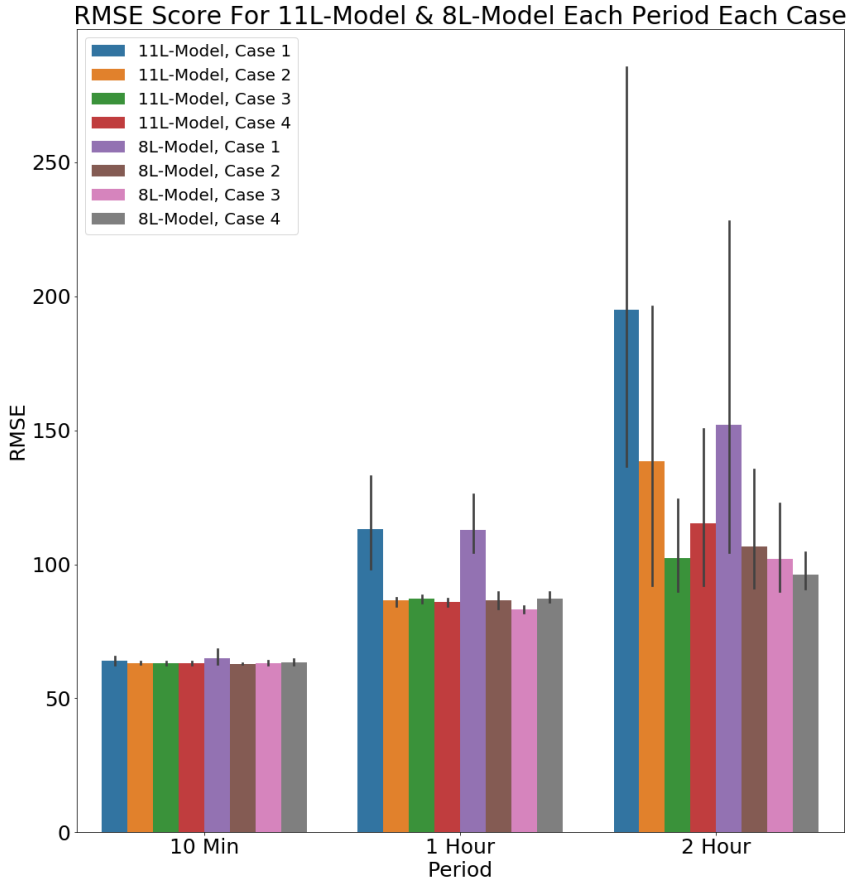

Figure 33: Figure shows the RMSE score for the two models, between each case, each time period, with an error bar calculated of the scores from each fold.

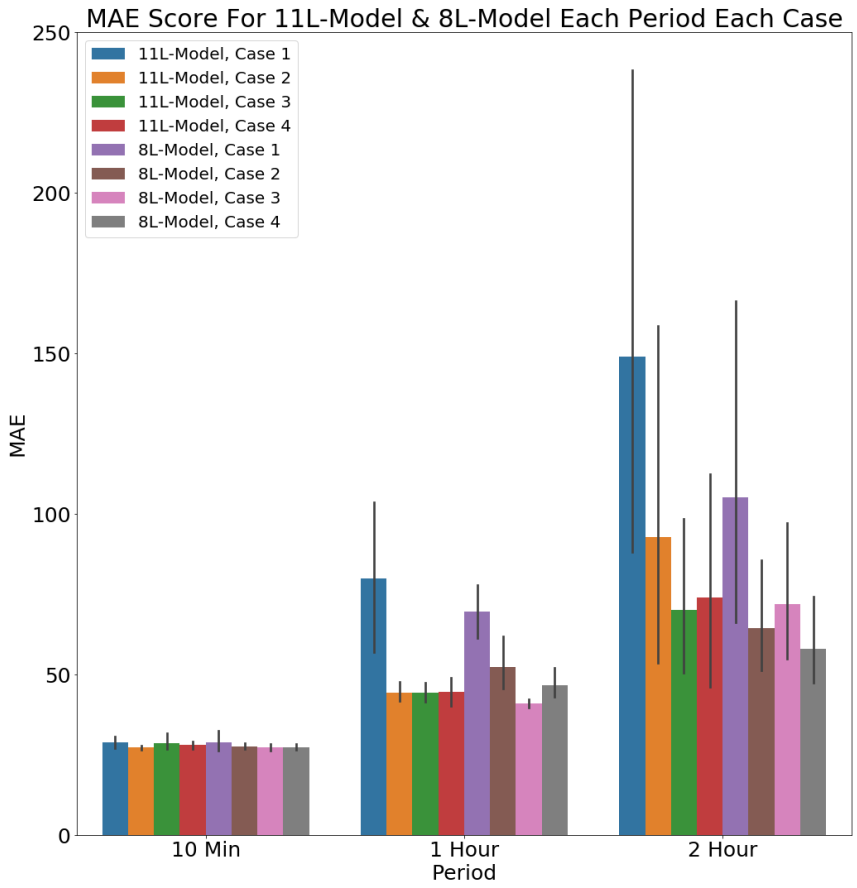

Figure 34: Figure shows the MAE score for the two models, between each case, each time period, with an error bar calculated of the scores from each fold.

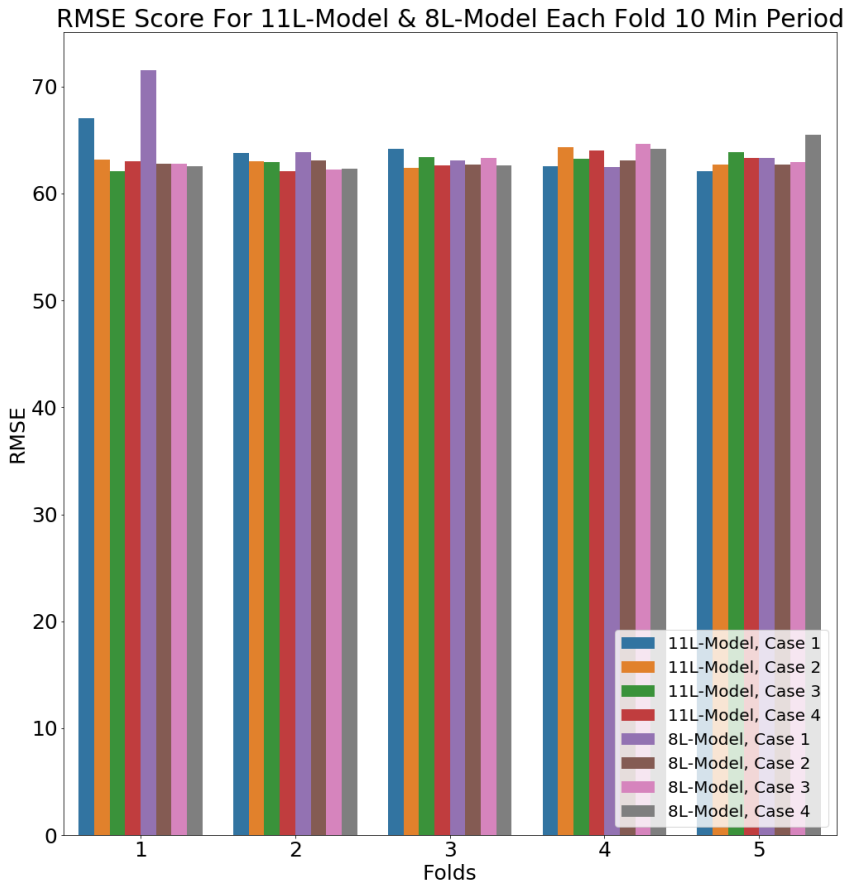

Figure 35: Figure shows the RMSE score for the two models, between each case, each fold and is presenting the ten-minute time interval.

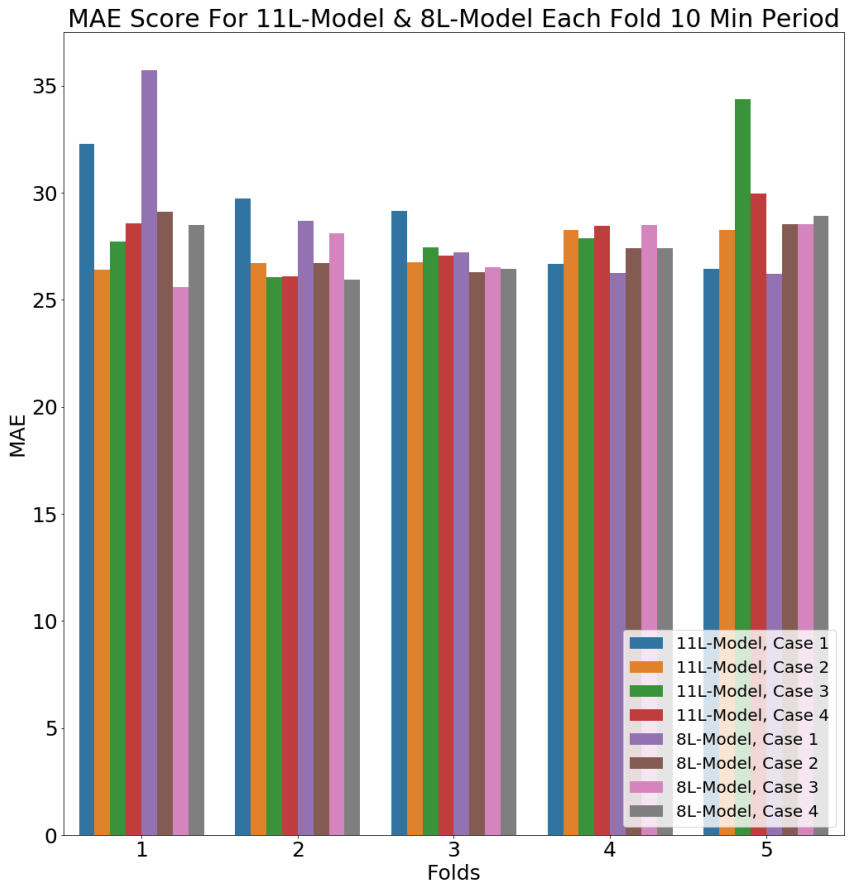

Figure 36: Figure shows the MAE score for the two models, between each case, each fold and is presenting the ten-minute time interval.

Table 17 and Table 18 presents the averages from five folds for both 11L- and 8L-Model and each case. These results show interestingly good performance of the 8L-Model Case 2 which produce a good RMSE score however not as good of a MAE score. The pattern of producing good RMSE and R-Squared and MAE and MAPE, or vice versa, seems to be common for the test results. That being said, the best 11L-Model case for RMSE and R-Squared seems to be Case 4 but for MAE and MAPE the Case 2 11L-Model preform the best results. When testing with the 8L-Model, Case 2 seems to produce the best RMSE and R-Squared. However, the best MAE and MAPE is from Case 4. It is very unexpected to see that Case 2 produces such a good results. However, it is important to note that when case 2 produces good in RMSE and R-Squared the MAE and MAPE is most often ranked 3rd, and vice versa. This means that Case 2 is either predicting good high values or predicting the majority of variables well while the opposite scores are poor. Interesting to notice is the difference in scores between the cases and the comparison study's GBM mentioned in Section 2.2.

Table 17: Table showing the average of five folds from 11L-Model and the standard deviation (SD) for RMSE and MAE predicting ten-minute timestep.

Model Case/Score Metric	RMSE	R^2	MAE	MAPE	SD RMSE	SD MAE
11L-Model, Case 1	63.90	0.52	28.85	25.36	1.92	2.41
11L-Model, Case 2	63.12	0.53	27.27	25.00	0.75	0.91
11L-Model, Case 3	63.10	0.53	28.71	27.85	0.68	3.25
11L-Model, Case 4	63.00	0.53	28.03	25.74	0.73	1.49
GBM	66.65	0.57	35.22	38.29	-	-

Table 18: Table showing the average of five folds from 8L-Model and the standard deviation (SD) for RMSE and MAE predicting ten-minute timestep.

Model Case/Score Metric	RMSE	R^2	MAE	MAPE	SD RMSE	SD MAE
8L-Model, Case 1	64.86	0.50	28.82	24.54	3.75	3.99
8L-Model, Case 2	62.87	0.53	27.61	25.86	0.22	1.20
8L-Model, Case 3	63.16	0.53	27.46	24.75	0.89	1.32
8L-Model, Case 4	63.44	0.52	27.45	24.09	1.37	1.27
GBM	66.65	0.57	35.22	38.29	-	-

The standard deviation of the RMSE and MAE for all cases are displayed in Tables 17 and 18. These metrics are interesting to notice due to the structures of the cases and how the data is handled. For example, Case 2 for the 8L-Model gets the lowest SDs out of the four which is not expected. This unanticipation is due to the low correlation of the two datasets also because and the case 2 trains and tests only on the Kaggle dataset and validates on the validation subset from the Belgium dataset. We have seen that the Kaggle data point distribution differentiate somewhat from the validation subset and also fluctuates internally as well, as it was seen in section 6.3 and shown in Figure 14. Case 1 do have a relatively high SD and this is due to the low amount of data in the first fold and since it preform significantly better when more fold are added, the SD is therefore increased. The SD for this analysis is used as a metric that represents the overall stability between folds. Figures 35 and 36 show where the difference in performance occurs and on which folds it happens. From those figures, for example, one may observe the difficulty of the first fold prediction of Case 1. The majority of the cases have a decrease in performance in

fold 4 and 5. This is because of the change in the distribution of the data in the Kaggle dataset.

7

SUMMARY AND CONCLUSION

The real-time collection of useful data in a building is a time consuming task. Therefore, this project has studied a possible solution including transfer learning in order to reduce the time spent on monitoring a target building. The aim is to train on a previously collected dataset of a building and fine-tune the model on a target building with smaller dataset. This study, proposes an algorithm that uses a bidirectional structure with LSTM nodes for predicting energy consumption in building. The model uses transfer learning between datasets that has a low correlation, in order for the strategy to be integrated in every day situations. The mentioned contributions allows for a decrease in time spent monitoring a target building since different datasets can be used in the training phase of a model and still produce acceptable results.

Transfer learning is applied in a form of parameter-based by freezing the parameters of the trained bidirectional LSTM model. A flexible model that can learn daily pattern is needed to be able to generalized to the validation strategy containing different time constraints. Therefore, the model has been using sequence to sequence prediction to be able to handle more data and to be more flexible. The results of predicting energy consumption have been validated against a previous study and the proposed model is shown to be more successful and have an overall better performance. The results of the previous study has been reported through a test with data points scattered throughout the set and this study has predicted only future points. This argues that the proposed model performs well in a generalization perspective. Important to note, the previous study has a limited amount of data in both training and testing which might explain the lacking performance.

The analysis in Section Data Analysis 5 shows interesting results, the different correlation heatmaps show no particular correlation between the datasets, which means that it has little to no similarity. Therefore, the transferability between the datasets seems not obvious, although, the validation phase tells a different story. That is the model has found a ”hidden” correlation, with multiple features in both datasets. The results in section 6.3 show that transfer learning performed better if training data cover a larger distribution of the target feature (energy consumption). As it has shown in section 6.3, Belgium Dataset has a rather acceptable distribution of the consump-

tion, in comparison to Kaggle Dataset. This can explain the difficulty of the model in Case 2 to predict as it only uses the Kaggle Dataset to train. Interestingly, the t-SNE maps show that there is some structure in the data that is specific to a time frame and data split. This is most identifiable in the t-SNE map for the Kaggle Dataset. This confirms an upset of differences between the datasets and their respective data distribution.

The structure of the proposed model was produced through a couple of tests. The selection of the activation and loss function is also performed experimentally and explained in Section 6.4. Among the most common loss functions, MSE were the most appropriate for this problem. The activation function were decided through a series of tests choosing between four combinations. Two types of hidden layer activation functions, ReLU and Tanh, and two types output layer activation function, Linear and Sigmoid. First test picked the two combinations with the best results from a five fold training over all timesteps. Secondly, a test with the same structure but with more focus on the complexity which identified Sigmoid-Tanh as the best activation function for the predictor.

The results was unexpected, for Case 2 that only uses a different Kaggle dataset and has low correlation to Belgium Dataset, had a performance that was close or equal to other cases. It also managed to score better evaluation metrics than the evaluation paper, "Data driven prediction models of energy use of appliances in a low-energy house" [39], as so did all cases. The R^2 metric is the only metric that the evaluation paper performed better at. Important to note, is that the results compared towards, used more features than our model. However the evaluation paper did not use information about previous energy consumption but included what type of day and dates. As Case 4 6.6.4, did preform better than the rest of the cases for the two-hour interval prediction, this can be considered the most useful information due to the application perspective for the real-world that it provides. That means that a prediction of ten-minutes might not be as useful for applications while a longer time period prediction is more beneficial and arguably more interesting. This being said, the results bring an opportunity for using the models in a real-world situation and for possible commercialization. It would have been preferable to have a larger training dataset, in order to have a substantial amount of data to train the models since the results from one hour and two hours, are similar. It is with great likelihood that the problem is a lack of data. We propose the use of the 8L-Model, since it is the model providing the best scores and is not in need of extra tuning in its training settings for different time periods. As of now the larger 11L-Model has stability problems in the ten minute period and

require more careful use. The study show that for the data available, more complex model is not necessary.

In future work, it is necessary to expand the potential use of the model to decrease the limits in using more diverse features in training datasets. Another interesting future work would be to include data from multiple buildings and see if the performance can be maintained or improved. This would allow for generalization improvements. Another potential interesting research would be to use the information about the usage of appliances to predict occupancy in the building. For such prediction to be possible additional data need to be gathered about the presence of people in the building. It would also be interesting to see if information about occupancy could improve the prediction of appliances use.

As final remarks for this thesis, with the knowledge of performing better than earlier studies we recommend using the proposed model with caution. It generalizes well within problems that include a sufficiently large dataset in the training phase. Even though predictions are somewhat acceptable regarding the overall volume of the data, to increase the accuracy for high values fine tuning is needed. Also, with a larger training dataset an increase in performance of the model for the longer time periods as of now the amount of data is too small. We believe that the proposed model have potential and with more research and testing can be improved to the point of application and possibly commercialization.

BIBLIOGRAPHY

[1] PB plc. *BP Statistical Review of World Energy*, 68 edition, 2019.

[2] EU, Leopold Quarter, Belgium. *Energy Efficient trends in the EU*. accessed 2019-11-25.

[3] Seungmin Rho Eenjun Hwang Jihoon Moon, Sungwoo Park. A comparative analysis of artificial neural network architectures for building consumption forecasting. *International Journal of Distributed Sensor Networks*, 15, 2019.

[4] Nasoo Kim YuJin Song Cheol Sik Pyo Marie Kim, JongAm Jun. Sequence-to-sequence model for building energy consumption prediction. *2018 International Conference on Inoformation and Communication Technology Convergence (ICTC)*, 2018.

[5] Lynne E. Parker Richard E. Edwards, Joshua New. Predicting futre hourly residential electrical consumption: A machine learning case study. *Energy and Buildings*, 49:591–603, 2012.

[6] Shengwei Wang Cheng Fan, Fu Xiao. Development of prediction models for next-day building energy consumption and peak power demand using data mining technuiques. *Applied Energy*, 127:1–10, 2014.

[7] Llorenc Burgas Joaquim Melendez Joan Colomer Joaquim Massana, Carles Pous. Short-term load forecasting in a non-residential building contrasting models and attributes. *Energy and Buildings*, 92:322–330, 2015.

[8] Takashi Ikegami Kazuhiko Ogimoto Yumiko Iwafune, Yoshie Yagita. Short-term forecasting of residential building load for distributed energy management. *IEEE Internatioanl Energy Conference*, pages 1197–1204, 2014.

[9] Perdo J.Mago Heejin Cho Kyungtae Yun, Rogelio Luck. Building hourly thermal load prediction using an indexed arx model. *Energy and Buildings*, 54:225–233, 2012.

[10] Dac-Khuong Bui Jui-Sheng Chou. Modeling heating and cooling loads by artificial intelligence for energy-efficient building design. *Energy and Buildings*, 82:437–446, 2014.

[11] Lv Jinhu Xu Gang Li Jibin Li Xuemei, Ding Lixing. A novel hybrid apporach of kpca and svm for building cooling load prediction. *2010 Third International Conference on Knowledge Discovery and Data Mining*, 2010.

[12] Llorenc Burgas Joaquim Melendez Joan Colomer Joaquim Massana, Carles Pous. Short-term load forecasting for non-residential buildings contrasting artificial occupancy attributes. *Energy and Buildings*, 130:519–531, 2016.

[13] Patricia J. Culligan John E. Taylor Rishee K. Jain, Kevin M. Smith. Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy. *Applied Energy*, 123:168–178, 2014.

[14] Ivan Fernandez Juan Prieto Oscar Bretos Cruz E. Borges, Yoseba K. Penya. Assessing tolerance-based robust short-term load forecasting in buildings. *Energies*, 6:2110–2129, 2013.

[15] Wil L. Kling Mohamed Elmitri Bruno Lacarriere Oliver Le Corre Subodh Paudel, Phuong H. Nguyen. Support vector machine in prediction of building energy demand using psuedo dynamic apporach. *ECOS 2015-In: Proceedings of the 28th International Conference on Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy Systems*, 2015.

[16] Luke Seewald Katarina Grolinger Miriam A. M. Capretz. Energy consumption prediction with big data: Balancing prediction accuracy and computational resources. *2016 IEEE International Congress on Big Data*, 2016.

[17] Vladimir N. Vapnik. *Statistical Learning Theory*. New York: Wiley, 1998.

[18] Khanna R. Awad M. *In: Efficient Learning Machines.* Apress, Berkeley, CA, 2015.

[19] Brian Zelle Minda Nelson Gang Sun, Steven Hoff. Development and comparison of backpropagation and generalized regression neural network models to predict diurnal and seasonal gas and pm10 concentrations and emissions from swine buildings. 2008.

[20] Jiejin Cai Hiroshi Yoshino Akashi Mochida Qiong Li, Qinglin Meng. Predicting hourly cooling load in the building: A comparison of support vector machine and different artificial neural networks. *Energy Conversion and Management*, 50:90–96, 2009.

[21] Lv Jinhu Xu Gang Li Jibin Li Xuemei, Ding Lixing. Building cooling load forecasting model based on ls-svm. *2009 Asia-Pacific Conference on Information Processing*, 2009.

[22] Qinglin Meng Qiong Li, Peng Ren. Prediction model of annual energy consumption of residential buildings. *2010 International Conference n Advances in Energy Engineering*, 52:2555–2564, 2010.

[23] Ilknur Erlalelitepe Uygun Kenan Evren Ekman Gulden Gokcen Akkurt Cihan Turhan, Tugce Kazanasmaz. Comparative study of a building energy performance software (kep-iyte-ess) and ann-based building heat load estimation. *Energy and Buildings*, 85:115–125, 2014.

[24] U. Teoman Aksoy Betul Bektas Ekici. Prediction of building energy consumption by using artificial neural networks. *Advances in Engineering Software*, 40:356–362, 2009.

[25] Andrew Baldwin Md. Uzzal Hossain Shazia Farzana, Meng Liu. Multi-model prediction and simulation of residential building energy in urban areas of chongqing, south west china. *Energy and Buildings*, 81:161–169, 2014.

[26] Eric W.M. Lee Simon S.K. Kwok. A study of the importance of occupancy to building cooling load in prediction by intelligent approach. *Energy Conversion and Management*, 52:2555–2564, 2011.

[27] A. Y. Abdelaziz. Static security using radial basis function neural network. 2005.

[28] Qijuin Chen Dandan Liu. Prediction of building lighting energy consumption based on support vector regression. *2013 9th Asian Control Conference*, pages 1–5, 2013.

[29] Madeleine Gibescu Wil L. King Elena Mocanu, Phuong H. Nguyen. Deep learning for estimating building energy consumption. *Sustainable Energy, Grids and Networks*, 6:91–99, 2016.

[30] Jürgen Schmidhuber Sepp Hochreiter. Long short-term memory. *Neural Computation*, 9:1735,1780, 1997.

[31] Milos Manic Daniel L. Marino, Kasun Amarasinghe. Building energy load forecasting using deep neural network. *IECON 2016 - 42nd Annual Conference of the IEEE industrial Electronics Society*, pages 7046–7051, 2016.

[32] Wenjie Gang Shenghan Li Cheng Fan, Jiayuan Wang. Assessment of deep recurrent neural network-based strategies for short-term building energy predictions. *Applied Energy*, 236:700–710, 2019.

[33] Bay vo Eenjun Hwang Seungmin Rho Sung Wook Baik Tuong Le, Minh Thanh Vo. Improving electric energy consumption prediction using cnn and bi-lstm. *Applied sciences*, 9, 2019.

[34] Sung-Bae Cho Tae-Young Kim. Predicting residential energy consumption using cnn-lstm neural network. *Energy*, 182:72–81, 2019.

[35] Flavio Miguel Varejao Thiago Oliveira-Santos Rodrigo F. Berriel Andre Teixeira Lopes, Alexandre Rodgrigues. Monthly energy consumption forecast: A deep learning approach. *2017 International Joint Conference on Neural Networks*, 2017.

[36] Sung-Bae Cho Tae-Young Kim. Predicting residential energy consumption using cnn-lstm neural networks. *Energy*, 182:72–81, 2019.

[37] Salahadin Mohammed Khaled A. Althelaya, El-Sayed M. El-Alfy. Evaluation of bidirectional lstm for short- and long-term stock market prediction. *2018 9th International Conference on Information and Communication Systems (ICICS)*, 2018.

[38] Milos Manic Daniel L. Marino, Kasun Amarasinghe. Building energy load forecastion using deep neural networks. *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016.

[39] Dominique Deramaix Luis M. Candanedo, VÃ©ronique Feldheim. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and Buildings*, 140:81–97, 2017.

[40] Charles Elkan Randall Wetzel Zachary C. Lipton, David C. Kale. Learning to diagnose with lstm recurrent neural networks. *ICLR*, 2016.

[41] Gang Lin Dichen Liu Zhaoyang Dong Hesen Liu Daqian Wei, Bo Wang. Research on unstructured text data mining and fault classification based on rnn-lstm with malfuction inspection report. *Energies*, 406:10, 2017.

[42] Abdel-rahman Mohamed Alex Graves, Navdeep Jaitly. Hybrid speech recognition with deep bidirectional lstm. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013.

[43] Kai Yu Zhiheng Huang, Wei Xu. Bidirectional lstm-crf models for sequence tagging. 2015.

[44] Kenji Matsuura Cort J. Willmott. Advantages of mean absolute error (mae) over the root mean squared error (rmse) in assessing average model performance. *Climate Research*, 30:79–82, 2005.

[45] Frank A.G. Windmeijer A. Colin Cameron. An r-squared measure of goodness of fit for some common nonlinear regression models. *Journal of Econometrics*, 77:329–342, 1997.

[46] TensorFlow. *Loss function LogCosh*. accessed 2020-04-11.

[47] Scikit-learn. *Loss function Mean Absolute Error*. accessed 2020-04-11.

[48] Taranveer Singh. *Smart Home Dataset with Weather Information*. accessed: 2020-05-04.

PO Box 823, SE-301 18 Halmstad
Phone: +35 46 16 71 00
E-mail: registrator@hh.se
www.hh.se

I'm Anton Gustafsson and with a large interest for Deep Learning conducted this thesis together with Julian Sjödal. I have studied to become a Master Engineer in Computer Science. Big thanks to our supervisors Sepideh Pashami and Abbas Orand.

My name is Julian Sjödal and I have studied for five years at the University of Halmstad at the master's programme in Computer Science. I am pleased with our result and want to thank everyone involved.

HALMSTAD UNIVERSITY

PO Box 823, SE-301 18 Halmstad
Phone: +35 46 16 71 00
E-mail: registrator@hh.se
www.hh.se