buildings

MDPI

Article

Building Energy Consumption Raw Data Forecasting Using Data Cleaning and Deep Recurrent Neural Networks

Junjing Yang [1,*], Kok Keng Tan [1,2], Mat Santamouris [1,3] and Siew Eang Lee [1]

1 Department of Building, National University of Singapore, Singapore 117566, Singapore; 0267338@u.nus.edu (K.K.T.); m.santamouris@unsw.edu.au (M.S.); bdglse@nus.edu.sg (S.E.L.)
2 Institute of System Science, National University of Singapore, Singapore 117566, Singapore
3 Faculty of the Built Environment; University of New South Wales, Sydney 2033, Australia
* Correspondence: bdgyj@nus.edu.sg; Tel.: +65-66012672

Received: 13 August 2019; Accepted: 9 September 2019; Published: 11 September 2019

Abstract: With the rising focus on building energy big data analysis, there lacks a framework for raw data preprocessing to answer the question of how to handle the missing data in the raw data set. This study presents a methodology and framework for building energy consumption raw data forecasting. A case building is used to forecast the energy consumption by using deep recurrent neural networks. Four different methodologies to impute missing data in the raw data set are compared and implemented. The question of sensitivity of gap size and available data percentage on the imputation accuracy was tested. The cleaned data were then used for building energy forecasting. While the existing studies explored only the use of small recurrent networks of 2 layers and less, the question of whether a deep network of more than 2 layers would be performing better for building energy consumption forecasting should be explored. In addition, the problem of overfitting has been cited as a significant problem in using deep networks. In this study, the deep recurrent neural network is then used to explore the use of deeper networks and their regularization in the context of an energy load forecasting task. The results show a mean absolute error of 2.1 can be achieved through the 2*32 gated neural network model. In applying regularization methods to overcome model overfitting, the study found that weights regularization did indeed delay the onset of overfitting.

Keywords: energy forecasting; deep recurrent neural networks; data imputation

1. Introduction

According to the US Department of Energy, Energy Information Administration, the residential and commercial sectors account for 36% of total energy consumption [1]. This substantial level of energy usage has focused research efforts into energy conservation in these two sectors. One energy conservation strategy lies in active demand and supply management which in turn relies on accurate energy consumption predictions.

In recent years, with the introduction of smart metering infrastructure which provided energy consumption data for research, data-driven approaches for energy consumption prediction using various methods have proliferated [2,3] At the same time, the rise of deep learning in the machine learning world led to dramatic improvements in many classical machine learning tasks in speech recognition, visual object recognition and detection, and other domains [4]. Coupled with the availability of large energy datasets which satisfies the key prerequisite for deep learning, its application to energy consumption prediction was matter-of-course and its promise of vastly improved prediction accuracy led to a growing research direction in big data energy analytics [5].

To enable big data analytics, a high-quality data set is necessary as the algorithm would learn from the existing and past data. However, the raw data from energy meters are usually full of outliers and missing values where preprocessing is always required for big data analytics [6]. There lacks a framework for raw data preprocessing to answer the question of how to handle the missing data in the raw data set.

In a survey of the state of the art in load forecasting, Fallah et al. [7] noted that deep learning approaches are known to suffer more from overfitting issues than shallow machine learning models. Powerful deep models with a large learning capacity easily learn incidental patterns in the training data which do not appear in other data, causing it to underperform more simple models. There also lacks a study to explore the use of deeper networks and their regularization in the context of an energy load forecasting task to reduce the overfitting issue.

Existing studies explored only use small recurrent networks of 2 layers and less, and therefore, the question of whether a deep network of more than 2 layers would be performing better for building energy consumption forecasting should be explored. Secondly, the problem of overfitting has been cited as a significant problem in using deep networks. This study aims to explore the use of deeper networks and their regularization in the context of an energy load forecasting task to reduce the overfitting issue.

The structure of the paper is arranged as following, Section 2 will review the state-of-the-art on electrical grid load prediction, Building Energy Consumption Prediction, household energy consumption, and the Deep Learning in Energy Consumption Prediction. Section 3 will state the case building and case data set, followed by Section 4 on Forecasting Methodology. Section 5 will illustrate the results and Section 6 will show the discussion.

2. State of the Art

Studies in energy consumption prediction differ in the scope of the energy consumption predicted. Energy consumption can be predicted for an electrical grid, a building, or an individual household.

2.1. Electrical Grid Load Prediction

At the electrical grid level, to achieve efficiencies in grid operations, electricity suppliers implement demand side management programs to moderate consumption patterns [5]. Demand response (DR) achieved through real-time pricing and load curtailment programs allows electricity suppliers to avoid high generation, transmission and distribution costs at demand peaks, and reduces overall capital investments in network reinforcements and capacity reserves required for maintaining grid reliability. Generation and load characterization and forecasting are key functions in a typical DR system.

Macedo et al. [6] presented a load curve pattern classifier using neural networks which characterized a load according to 4 different profiles. Daily consumer loads are first classified and grouped. The load profiles determine the best control actions in system management at the secondary system level for transformers and also substations at the primary system level. The classifier, a self-organizing neural network of 10 hidden nodes, was trained using learning vector quantization, a supervised learning algorithm.

In the study by Ding et al. [8] on short-term load forecasting model for distribution systems using neural networks, they focused on feature engineering and model parameter selection, having found in a survey that most works, while showing success using neural networks, lacked clear explanation about their choice of input variables and model parameters. Two single-layer feedforward neural network of up to 10 hidden nodes were trained to predict the daily average power and intra-day power variation respectively which were combined to obtain the 24-hour ahead forecast at a 30-minute frequency. Different variable sets from load, temperature, and date/time variables were used in each model. The best neural network model was found to out-perform a variant of the time series decomposition method using additional weather forecast variables in trend regression and spectral analysis for cycle regression.

Liu et al. [9] used long and short-term memory (LSTM) recurrent neural networks (RNN) on hourly frequency grid load data with multiple cyclical components for forecasting a 24-hour load profile. The single layer LSTM network had 256 hidden nodes processing sequences of length 28. Better accuracy was obtained in comparison to a feedforward network.

2.2. Building Energy Consumption Prediction

While energy consumption is obviously aggregated across many individual dwellings at the building level which lessens its inherent volatility, forecasting nonetheless remains a challenging task due to the wide-ranging factors which affect consumption such as the weather, equipment, number of occupants and their energy-use behavior [3]. Notwithstanding the task complexity, prediction models which achieve even a small percentage of improvement can reap large economic and social benefits when applied to high levels of energy consumption in large buildings, wherein lies the attraction of forecasting building energy consumption as a topic for research [10].

Zheng et al. [11] reported using a LSTM network on a school's engineering building energy consumption sampled at 15-minute frequency for forecasting with a forecast horizon of 24 hours using a lookback of 10 days. While the study results showed that the LSTM network outperforms traditional forecasting methods such as support vector regression (SVR) and seasonal auto-regressive integrated moving average (SARIMA), few details of the LSTM network was provided.

Nichiforov at al. [12] applied deep recurrent neural networks for load forecasting in large commercial buildings using long and short-term memory (LSTM) networks. Several LSTM networks, comprising a single LSTM layer of varying size and a single fully-connected layer, were tested on the one-year energy consumption data of two university campus buildings, collected at a 60-minute frequency.

Amber et al. [10] reported the use of multi-layer feedforward (MLFF) neural network and deep learning for predicting the daily electrical consumption of an administrative building in London. The building data used comprised daily electricity usage, daily mean surrounding temperature, daily mean global irradiance, daily mean humidity, daily mean wind velocity, and weekday index. The MLFF neural network used had 10 hidden layers each of 100 nodes which qualifies it as a deep network. No description of the other deep neural network used in the study was provided. The study found that the MLFF neural network performed best overall in comparison to the deep neural network and other shallow machine learning and regression methods used in the study.

Zheng et al. [13] compared the performance of single and two-layer 20-node LSTM and gated recurrent unit (GRU) RNNs for forecasting residential community load using a dataset of hourly loads. The study found that GRU and LSTM networks gave similar accuracy but the former incurred shorter training time.

Jiao et al. [14] investigated the energy consumption of non-residential consumers in business and industry. The dataset comprised electrical load data from 48 non-residential customers sampled at 15-minute intervals. This study is notable for using feature engineering together with deep networks. Three separate LSTM networks were trained using the 15-minute frequency load sequence and two other load sequences at the timescale of day and week respectively. A day-scale sequence comprised the loads at the same time point on consecutive days; a week-scale sequence has the loads at the same time point, day of the week on contiguous weeks. A range of 3–10 time steps were used for the 3 LSTM networks. Time and day of the week indices and binary holiday indications together with the load sequences were also fed to each LSTM network with hidden nodes ranging from 32 to 64. The study found that the multiple sequence LSTM solution performed best in general compared to random forest, SVR, and single layer feedforward network.

Rahman et al. [15] proposed a neural network based on an encoder–decoder architecture for building energy load forecasting. The encoder was a 2 or 3-layer LSTM network coupled to a single-layer feedforward network in two variants of the proposed neural network explored. All the layers both recurrent and feedforward had 24 hidden nodes. The input sequences comprised energy

load values for a single-day duration at 1-hour intervals. To overcome model overfitting, weights regularization was applied. The study found that the proposed neural networks performed better than a 3-layer feedforward model.

2.3. Household Energy Consumption Prediction

At the household level, unlike aggregated energy consumption, energy loads exhibit much more variability in patterns which hamper accurate forecasting. In households, daily routines, the lifestyle of its members, and the ownership of types of appliances all have substantial impact on the short-term energy load. Using the half-hourly energy load readings for a dataset of 96 households, Kong at al. [16] showed that daily household energy load profiles vary greatly from household to household. While some households have daily profiles which can be organized into major clusters with few outliers, others show no clusters at all. To handle this complexity, a deep LSTM neural network with lookback of 2–12 time steps, two hidden layers of 20 nodes each was used for a 1-step ahead household energy load forecast. The study found that in general the LSTM network performed best in comparison with a MLFF neural network and other shallow machine learning methods.

In another study, Kong et al. [17] used a larger 2-layer 512-node LSTM network with a lookback of 2–12 time intervals to predict household energy consumption using the energy consumption data from individual household appliances. The study again found that the LSTM network outperforms other forecasting methods.

In addition to using LSTM networks, Hossen et al. [18] tested 2-layer recurrent networks of 50 to 100 nodes using GRU nodes for forecasting on a dataset of individual sub-meter readings using varying lookback of 30 to 50 time intervals. The LSTM network performed better than the GRU network while both recurrent networks were superior to other conventional time series methods.

Shi et al. [19] proposed a pooling method to overcome a lack of training data which led to overfitting in deep LSTM networks. By pooling energy consumption data from multiple households in the same locality during model training, the study found that deep LSTM networks of up to 5 layers of 30 hidden units each can be successfully trained to outperform smaller LSTM networks as well as shallow machine learning models in forecasting individual household energy consumption. The authors attributed the success of the pooling approach to the fact that energy consumption was driven by factors common to all households such as ambient temperature, day of the week, etc. Pooling the energy consumption data hence allowed the model to learn these patterns across households in addition to providing more training data to the deep network which rendered it less susceptible to overfitting.

Due to the limited training data available, Belyaev et al. [20] used a simple RNN instead of more complex LSTM or GRU networks. The dataset comprised of data from 47 households sampled only once or twice a day. The best model was a single-layer RNN network of 10 hidden nodes in comparison with classical polynomial regression.

In their study on building energy load forecasting, Rahman et al. [15] also applied their proposed encoder–decoder RNN to aggregate energy consumption of households. While the performance of the RNN was better than a regular feedforward neural network at building level, the feedforward network outperformed the RNN for aggregated household energy consumption.

2.4. Deep Learning in Energy Consumption Prediction

Several studies have already explored the use of deep learning in the form of deep feedforward as well as recurrent networks as shown in Table 1. In most of these studies, while there is some exploration of the parameter space to find the set of network parameters which gives the best forecast accuracy, there is little mention of how network parameters are selected in relation to the problem on hand as has already been noted by Ding at al. [8].

Table 1. Neural networks in energy load forecasting (??? = information not provided in paper).

S/No.	Authors	Year	System Level	Network Type	Size (#Layers × #Nodes per Layer)	Sequence Length
1.	Macedo et al.	2015	grid	Self-organizing	1 × 10	-
2.	Ding et al.	2016	grid	Feedforward	2 × (1–10)	-
3.	Liu et al.	2017	grid	LSTM	1 × 256	28
4.	Zheng et al.	2017	building	LSTM	???	960
5.	Nichiforov et al.	2018	building	LSTM	1 × ???	???
6.	Amber et al.	2018	building	Feedforward	10 × 100	-
7.	Zheng et al.	2018	building	LSTM/GRU	2 × 20	???
8.	Jiao et al.	2018	building	3 x LSTM	1 × (32–64)	3–10
9.	Kong et al.	2017	household	LSTM	2 × 20	2–12
10.	Kong et al.	2018	household	LSTM	2 × 512	2–12
11.	Hossen et al.	2018	household	LSTM/GRU	2 × (50–100)	30–50
12.	Shi et al.	2018	household	LSTM	(1–5) × (5–100)	48–336
13.	Rahman et al.	2018	Building/household	LSTM	(3–4) × 24	24

The LSTM and GRU networks explored are all single or two-layer networks with up to a maximum of 512 nodes per layer. Only Zheng et al. [13] and Shi et al. [19] explored the use of long sequence lengths.

A key concept in deep learning is the learning of deep distributed representations in which high level features in the data are learned through the progressive composition of many non-linearities learned in each layer of the network. While it may not be impossible for a shallow network to learn the same, given that a single layer network is shown to be a universal approximator, it will require likely a large number of hidden nodes. It has also been proven that a deeper network of more layers can represent a function using a smaller number of hidden nodes compared to a shallow model, achieving a higher statistical economy [21]. Deeper networks hold the potential to achieve better performance over large shallower networks.

The sequence length processed by the network determines the high-level features which can be learned in sequential data. A suitable sequence length which covers the period of the cycles expected in the data (e.g., weekly) enhances the learning ability of the network. It should also obviate the need for manual feature engineering of the type seen in Jiao et al. [14] where time and day indices are provided as input to the recurrent network. A suitably long sequence length amplifies the ability of the network to independently learn deep distributed representations and avoid brittle manual feature engineering which is a key reason for employing a deep learning approach over shallow models.

In a survey of the state of the art in load forecasting, Fallah et al. [7] had noted that deep learning approaches are known to suffer more from overfitting issues than shallow machine learning models. Powerful deep models with a large learning capacity easily learn incidental patterns in the training data which do not appear in other data, causing it to underperform more simple models. However, apart from Shi et al., all cited studies using deep models did not mention overfitting as an issue and thereby also any measures taken to overcome it. The likely reason is that the models studied were not deep enough that their performance was degraded by overfitting. While Shi et al. [19] did explore overcoming overfitting in their 5-layer deep LSTM model, it was from the angle of increasing the size of the training data without the use of any regularization methods. Deep models require regularization measures to achieve good performance over shallower models. Among other developments, Lecun et al. [4] singled out the dropout regularization method as one key enabler in the rise of deep learning.

3. Case Building and Case Dataset

The data comprises the electricity consumption and cooling thermal energy of a mixed function institutional building in a university campus in Singapore. Each data is measured at a half-hourly frequency from June 2015 to March 2017. As seen in the plots of the data (Figure 1), there is missing data in both datasets.

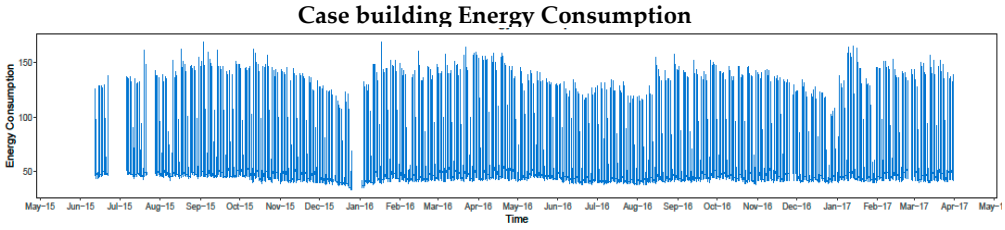

Figure 1. Electricity consumption and cooling thermal energy consumption data from June 2015 to March 2017.

The energy consumption data exhibits weekly cycles (Figure 2). The plot shows the energy consumption for a typical week. Energy consumption peaks during office hours and reduces to a minimum in the hours of the night and early morning. The energy consumption is also lower on Saturdays and lowest on Sundays. The plot also shows a clear correlation between the energy consumption and cooling capacity in use.

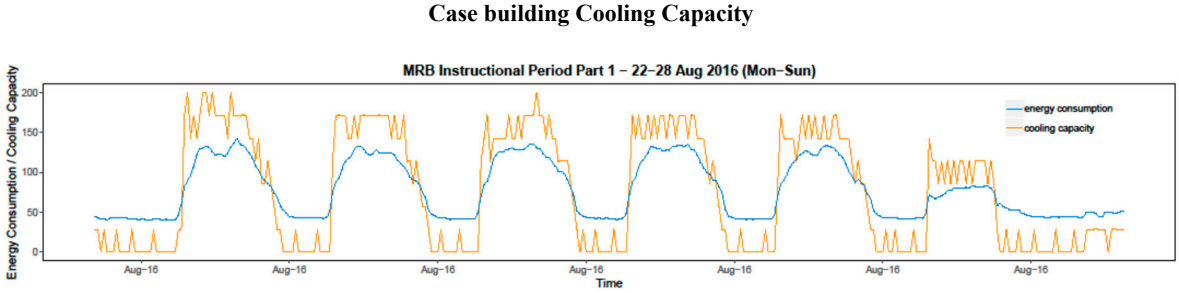

Figure 2. Weekly cycles in the data.

The energy consumption data also exhibits a longer-term trend over the period of an academic calendar year. The weekly cycles differ in amplitude during different phases in the year. The plot (Figure 3) shows the lower peaks in a typical week during the vacation compared to the instructional period.

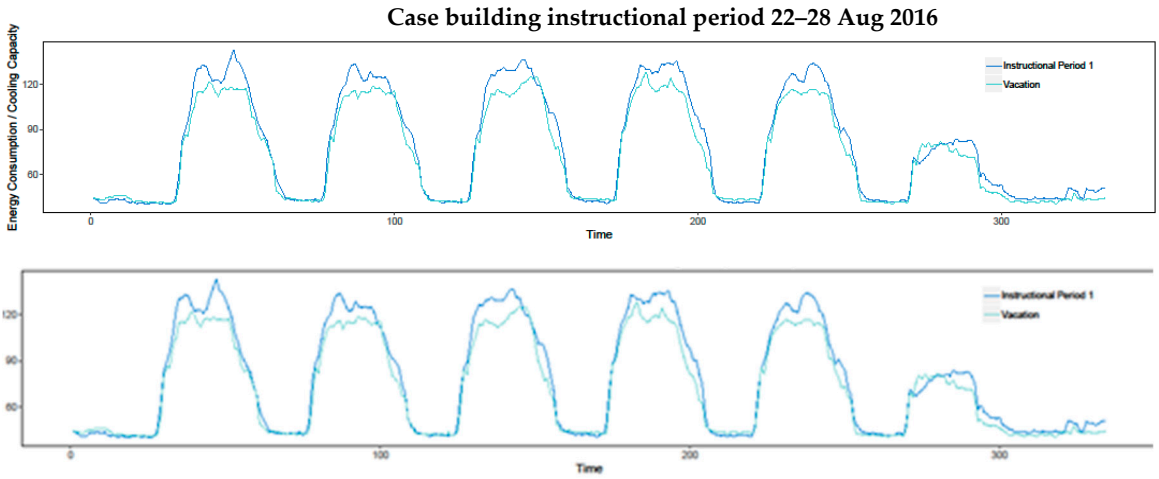

Figure 3. Trend in the data.

3.1. Data Transformation

The plot in Figure 4 (left) shows missing data statistics for energy consumption data. Figure 4 (right) shows the distribution of missing data in the energy consumption data.

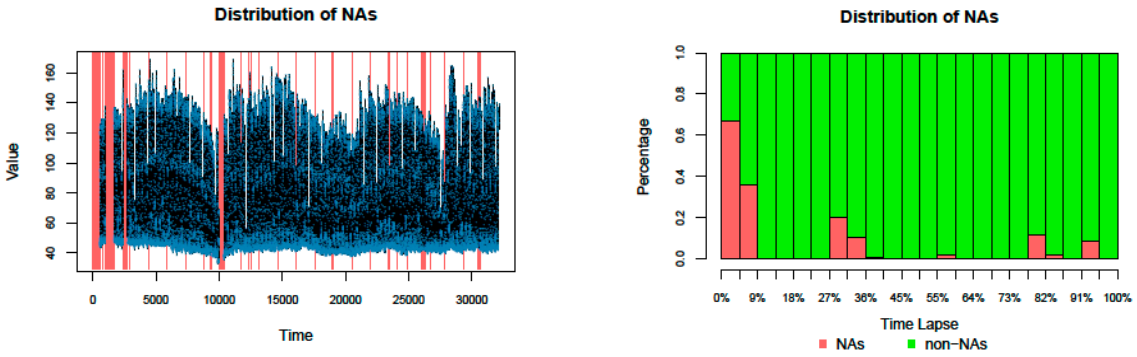

Figure 4. Missing data statistics for energy consumption data.

There are 21 instances of a 2-time step gap, 3 instances of a gap of 6-time steps and single instances of large gaps in the data of sizes ranging from 9 to 660-time steps (Figure 5).

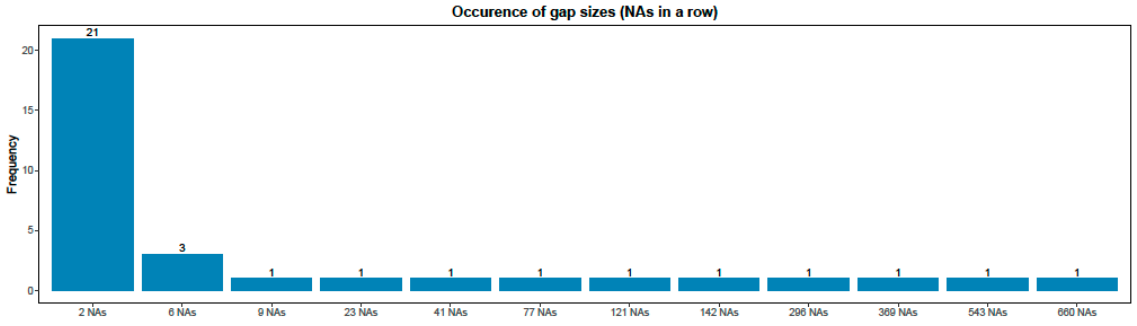

Figure 5. Occurrence of gap sizes.

In order to preprocess the raw data, 4 data imputation methods were tested on the data, structural model with Kalman smoothing, ARIMA model with Kalman smoothing, spline interpolation and exponential moving average.

The exponential moving average and spline interpolation are relatively simple general mathematical methods using in data imputation. The ARIMA method is commonly use in the domain of financial and econometric analyses while the use of the structural model is typically seen in moving object trajectory tracking applications [22].

Ten periods of time in which there were no missing data were extracted from the data. Missing data is next simulated on these 10 datasets using an exponential distribution with a range of values for the rate parameter λ (Figure 6). The rate parameter defines the average number of missing events per time step. As in most missing data methods, the simulation assumed that the missing data is missing completely at random (MCAR) [23]. Missing data which is generated by a systemic process requires an analysis of the generation process.

The missing data is imputed using the 4 imputation methods and the imputation error calculated.

Imputation using AutoRegressive Integrated Moving Average (ARIMA) produced large errors in 3 of the data sets for $\lambda = 0.25$ (bottom right in Figure 7). The outliers produced are shown in together with correctly imputed data for comparison (Figure 8).

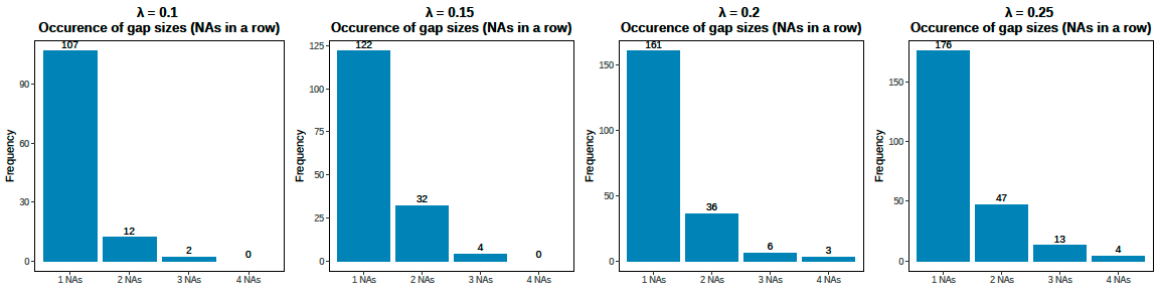

Figure 6. Distribution of simulated missing data.

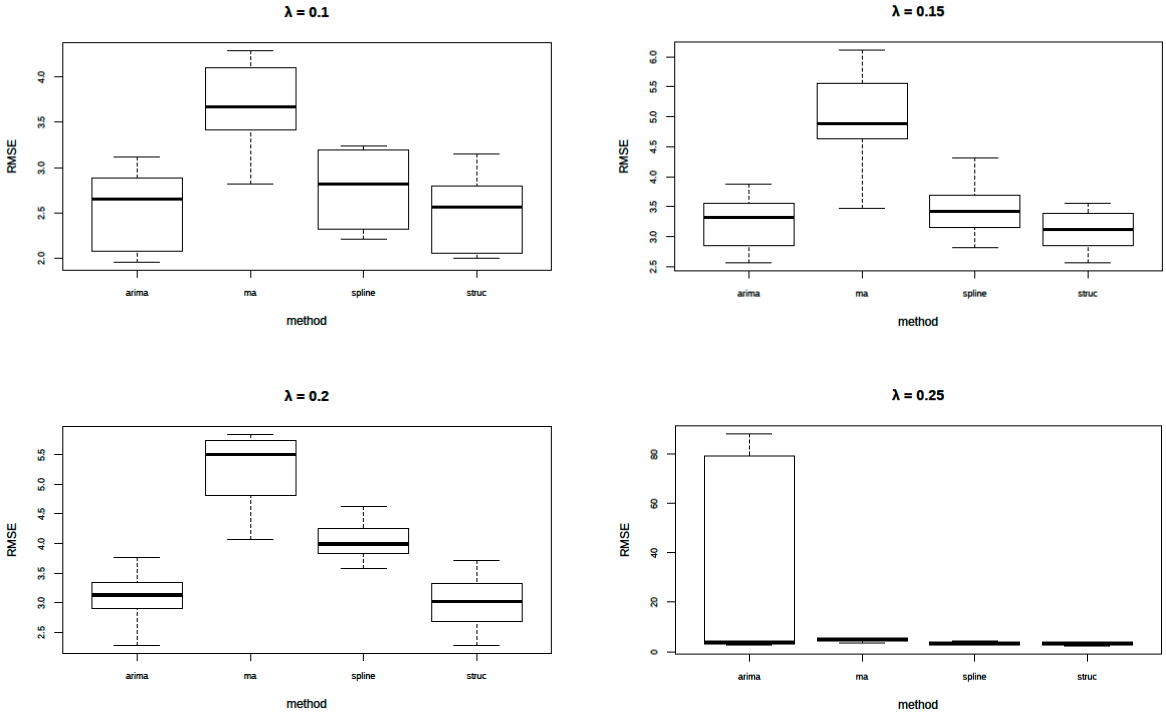

Figure 7. Distribution of data imputation errors.

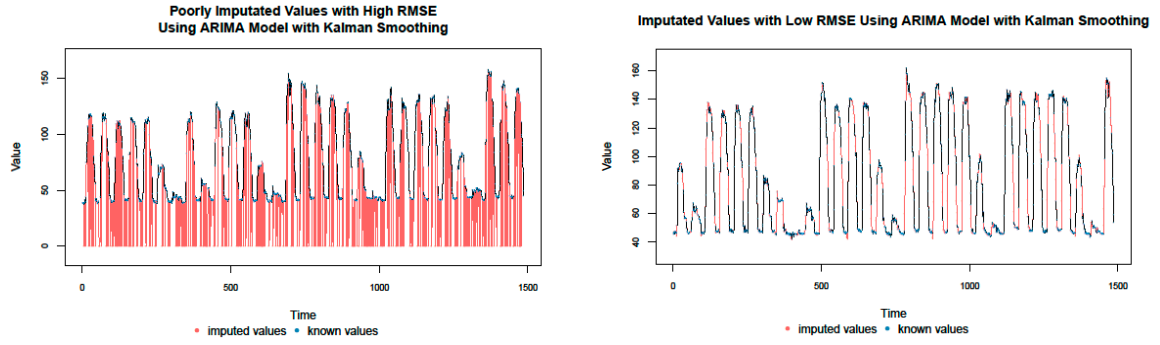

Figure 8. Outliers and correctly imputed data using ARIMA method.

A one-way ANOVA and pairwise T-test showed significant differences in the errors with the simple exponential averaging method performing worst for all the values of λ. For λ = 0.1, 0.15, and 0.25, the other three methods did not perform differently (ARIMA was excluded from the comparison for lambda = 0.25 because of the large errors). For λ = 0.2, ARIMA and structural model performed significantly better than spline interpolation.

The sensitivity of gap size and available data percentage on the imputation accuracy has been tested in Figure 9. It shows the accuracy reduces with the increase of gap size for most of the time. There is no clear elbow point to determine when to stop the imputation. Since the data imputation will use the existing data to impute the missing data, Figure 9 shows the sensitivity of exsiting data percentage on the data imputation accurancy. The results shows the accuracy does not change with the decrease of available data percentage.

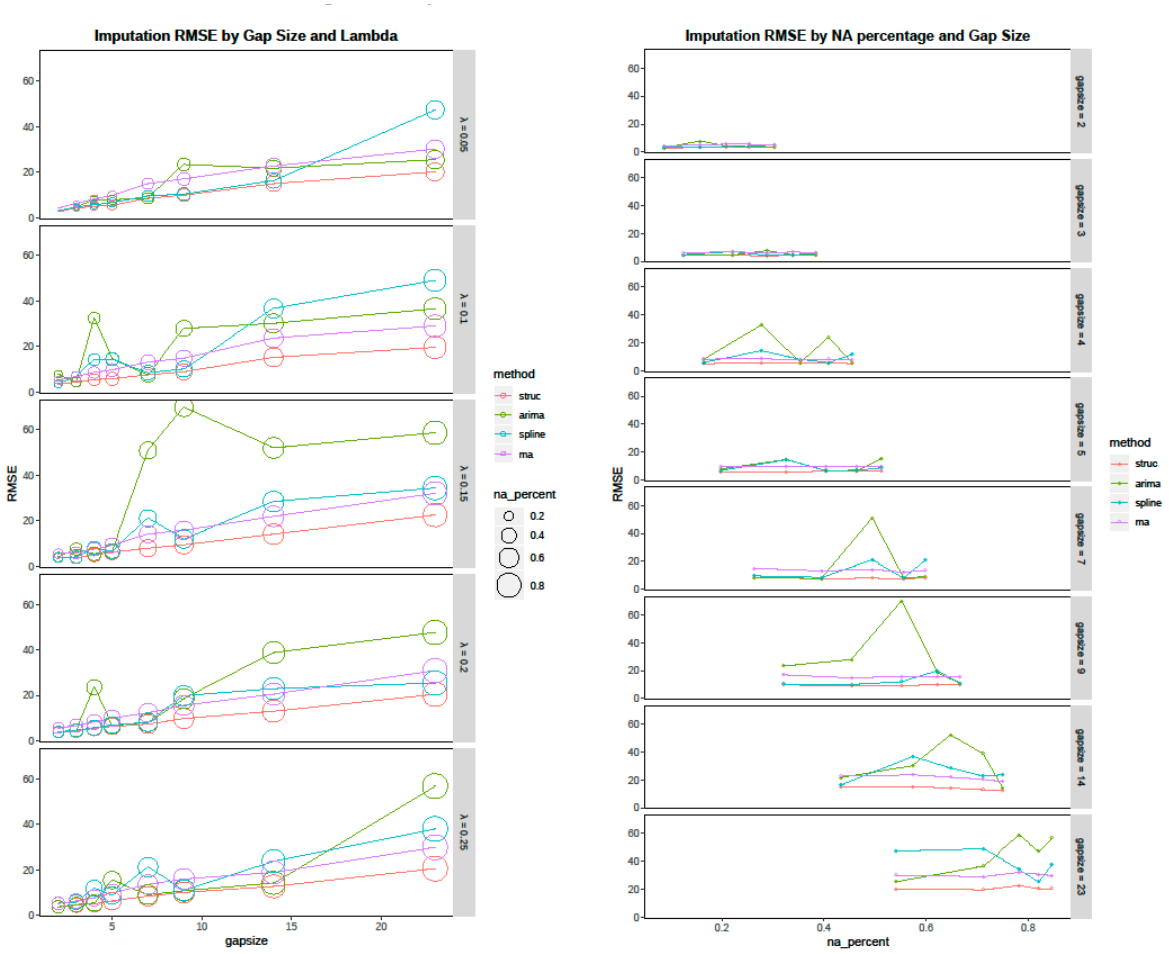

Figure 9. Sensitivity of gap size/existing data percentage on the data imputation accurancy.

3.2. Missing Values in the Training Data Set

SVR is used as the benchmarking base model for comparison of the gated neural network, where SVR is unable to handle missing values in training data. Discarding data has two disadvantages. A resulting smaller training dataset may adversely affect the performance of the trained model. In addition, if the missing data mechanism is not MCAR [24], its removal will introduce biases into the training data. Given the results in Table 2 root mean square error (RMSE) for data imputation (lowest RMSE highlighted), the structural model method was selected for missing data imputation. A conservation value for the maximum missing data gap size of 3-time steps was also selected to minimize imputation error. Only missing data of gap size less than this value was subject to data imputation. Missing data of larger gap size were not imputed.

Neural network models handle missing input data in another way. In this phase of modeling, the remaining missing data were replaced by a value not observed in the data. The selected value is –1 as the input data was normalized to [0, 1] using min–max scaling during pre-processing. Only data samples in which the label or all the inputs are missing, were discarded [25]. During training of

the neural network, it will both learn the meaning of the value –1 and learn to ignore the value [26], as shown in Table 3.

Table 2. Root mean square error (RMSE) for data imputation (lowest RMSE highlighted).

λ	Structural Model	ARIMA	Spline Interpolation	Exponential Moving Average
0.10	2.497596	2.524483	2.801944	3.687430
0.15	3.096260	3.259848	3.446748	4.962951
0.20	3.020435	3.141991	4.027506	5.230314
0.25	3.296376	27.560808	3.453490	5.008605

Table 3. Handling of missing data in neural networks.

Input Data	Label	Accept/Discard
Some values are −1	Valid value	Valid data sample
Any sequence of values	−1	Discarded
All values are −1	Valid value	Discarded

4. Forecasting Methodology

4.1. Machine Learning Workflow

The typical machine learning workflow comprises the following steps:

1. Define problem and measure of success;
2. Define an evaluation protocol;
3. Prepare data;
4. Develop benchmark and base models;
5. Scale up and regularize base model.

This basic workflow applies to both shallow models as well as deep. Steps 1 and 2 are largely self-explanatory. In this study, the problem is energy load forecasting and a suitable metric is specified in a later section, 4.4 Forecasting Task. The standard evaluation protocol used in machine learning is cross-validation [27]. In this study, due to limitations in the availability of computation resources, an out-of-sample or last-block evaluation approach is used instead. The time series dataset is partitioned using a simple 2:1 split with the larger split forming the training split and the other the validation split.

The main difference in workflow between shallow and deep models lies in the data preparation step. In this step, carefully crafted manual features which help shallow models perform well are introduced into the modeling process. Deep learning model obviate the need for such intensive and manual feature engineering work by learning the requisite higher-level representations for successfully performing the machine learning task automatically and independently.

The benchmark models serve as a baseline to demonstrate that the base machine learning model which uses default parameters and hyper-parameters is capable of making reasonably accurate predictions. The base model is gradually scaled-up to increase its predictive power until overfitting sets in and adversely affects its performance. To reduce the effects of overfitting and improve the performance of the base model, network regularization methods are employed together with fine-tuning of the network hyper-parameters.

4.2. Gated Recurrent Units

RNNs are neural networks specialized for processing sequential data [21]. However, simple RNNs perform poorly in the learning of long-term dependencies between an input and the predicted

output. In 1997, Hochreiter et al. [28] introduced a gated RNN based on the LSTM unit to greatly improve the performance of RNNs in learning such dependencies. The gates in the LSTM unit are themselves fully-connected feedforward neural networks. Proposed more recently by Cho et al. [29], the GRU is an RNN hidden unit selected in this study for its simplicity to implement and to compute in comparison with the LSTM. The GRU consolidates the three input, output, and forget gates in the complex LSTM architecture [30], whose design rationale is not totally clear as noted, into two reset and update gates while retaining an ability to learn dependencies over different time scales.

4.3. Regularization Methods

The network regularization methods used are dropouts [31] and weights regularization [32]. Dropouts can be applied to the outputs of any layer in the network. A fraction of the outputs of a layer are randomly zeroed out by the applied dropouts according to the specified dropout rate. This effectively adds noise into the network during training so that overfitting does not set in easily.

Weights regularization accomplishes the same effect by adding a penalty term to the loss function proportional to the value of the weights in the network. This prevents the weights from growing too quickly and also decays weights which are not updated to zero, both of which aids in delaying the onset of overfitting. There are two types of weight regularization functions. The L1 function is the sum of all the absolute value of the weights while the L2 function is the sum of all the square of the weights.

4.4. Forecasting Task

The forecasting task for this analysis is a one-step or 30-minute ahead energy consumption forecast using a 5-day lookback which is a sequence of 240-time steps (i.e., 48-time steps per day × 5 days).

The sequence length is selected given the weekly cycle seen in the data. Intuitively, looking back at least 5 days will allow the current day of the week to be determined given the weekly cycle of H-H-H-H-H-L-LL where H is high consumption, L is low, and LL lowest. For example, for a sequence H-H-H-H-H, the next value in the sequence is L.

The metric used in the comparison analysis is mean absolute error (MAE).

5. Results

5.1. Benchmarking Base Model

To obtain a benchmark for comparison against the performance of the recurrent network models, two models were developed. The first is the last observation carry forward (LOCF). It is a naïve model in which the one-step ahead prediction is the last observed value in the sequence. The MAE calculated for the LOCF model on the validation split using a simple 2:1 split is 3.983. Despite being extremely simple, the LOCF model otherwise also known as simple persistence forecasting, is nonetheless a credible benchmark as research in forecasting has shown. Outperforming the LOCF model when the variation in the data is high is in reality difficult. The second benchmarking base model is support vector regression. Since the method of support vector machines(SVR) was proposed in 1995 [33], SVR is a popular machine learning model proposed for forecasting tasks even recently [34–36]. It has also been used as a benchmark for comparison in various studies in the survey of related works [14,18].

The SVR benchmark model was implemented using the sci-kit learn machine learning library [37]. The model parameters were selected using grid search on a 5-fold time series split using 240 lag values as input data. Samples with missing data after imputation were discarded as the SVR training algorithm did not handle missing values. The parameters were a gamma value of 1e-3 for the radial basis function kernel and a value of 100 for the C parameter. When trained using a simple 2:1 split, the validation MAE for the best SVR model was 4.388. The LOCF model outperformed the SVR model in MAE which showed that it set a high bar despite being a naïve model, as shown in Figure 10.

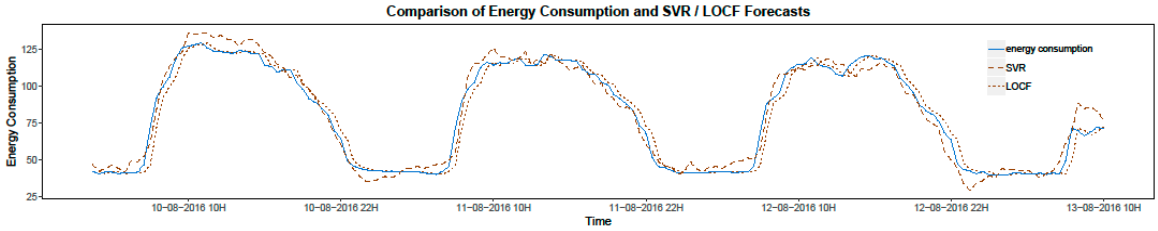

Figure 10. Comparison of LOCF forecasting, SVR forecasting, and measured data.

5.2. Base Recurrent Neural Network Models

All neural network models were implemented on the Keras [38] and Tensorflow [39] machine learning libraries. The following 4 base models (Table 4) were trained on the data:

Table 4. Base models (FF = feedforward).

S/No.	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
1	128 GRU	128 GLRU	128 GRU	128 GRU	32 FF	32 FF
2	64 GRU	64 GRU	64 GRU	64 GRU	32 FF	-
3	32 GRU	32 GRU	32 GRU	32 GRU	-	-
4	32 GRU	32 GRU	-	-	-	-

The training curves (Figure 11) show that the two deeper networks overfit quicker than other networks. Without overfitting badly, the smaller networks achieve a better performance in MAE, the best of which is 2.122 (in actual units) from the smallest two-layer model (model s/n 4 in Table 4). All the base models outperformed the benchmark LOCF model MAE of 3.983.

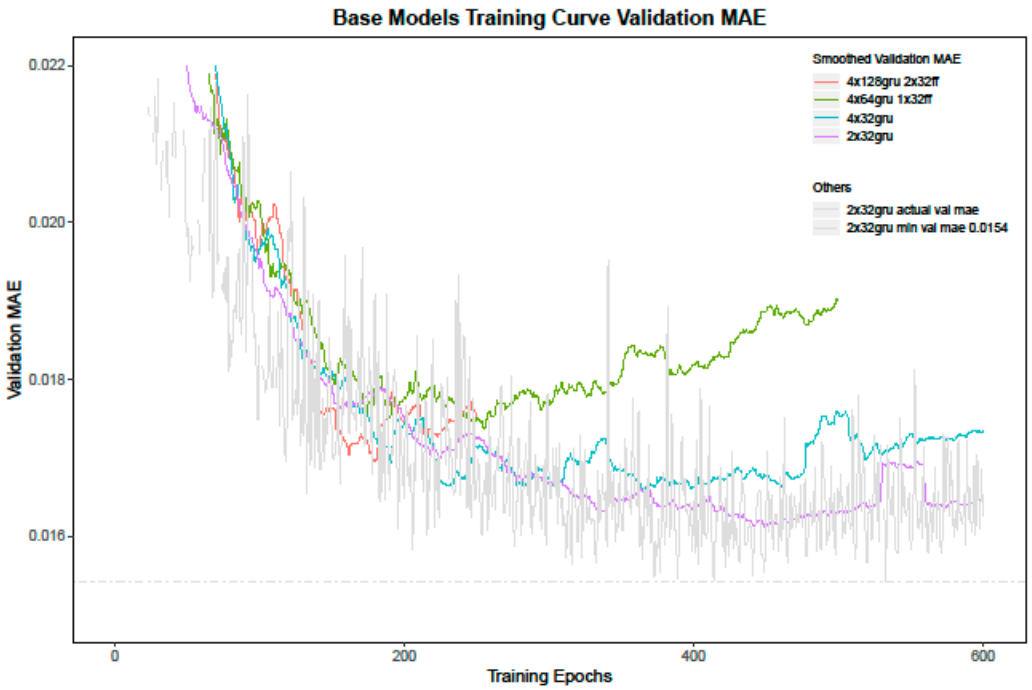

Figure 11. Base model training curves (training curves are smoothed using 20-epoch moving average; the background plot is the actual training curve of the best model).

5.2.1. Dropout Regularization

Dropouts were used without any success on the deepest model. Dropouts of different rates from 0.05 to 0.2 were used on the GRU layers for both input and recurrent state. The validation MAE achieved was worse than that of the base model.

5.2.2. Weights Regularization

4 × 128 GRU, 2 × 32 FF Model

L1 and L2 weights regularization were used on the model. Various values of the regularization parameters were explored to determine the best value as shown in Figure 12. The best set of parameters found while training using the default learning algorithm RMSPROP [33] was also re-run using ADAM [40,41]. The best MAE of 2.146 is achieved using both L1 and L2 regularization with a parameter value of 3e-7 on all the weights (bias, kernel, and recurrent) of all the recurrent layers using ADAM. However, this best model is still outperformed by the base model with a MAE of 2.122. Regularization prevented overfitting from setting in early but failed ultimately to achieve a better performance over the base model.

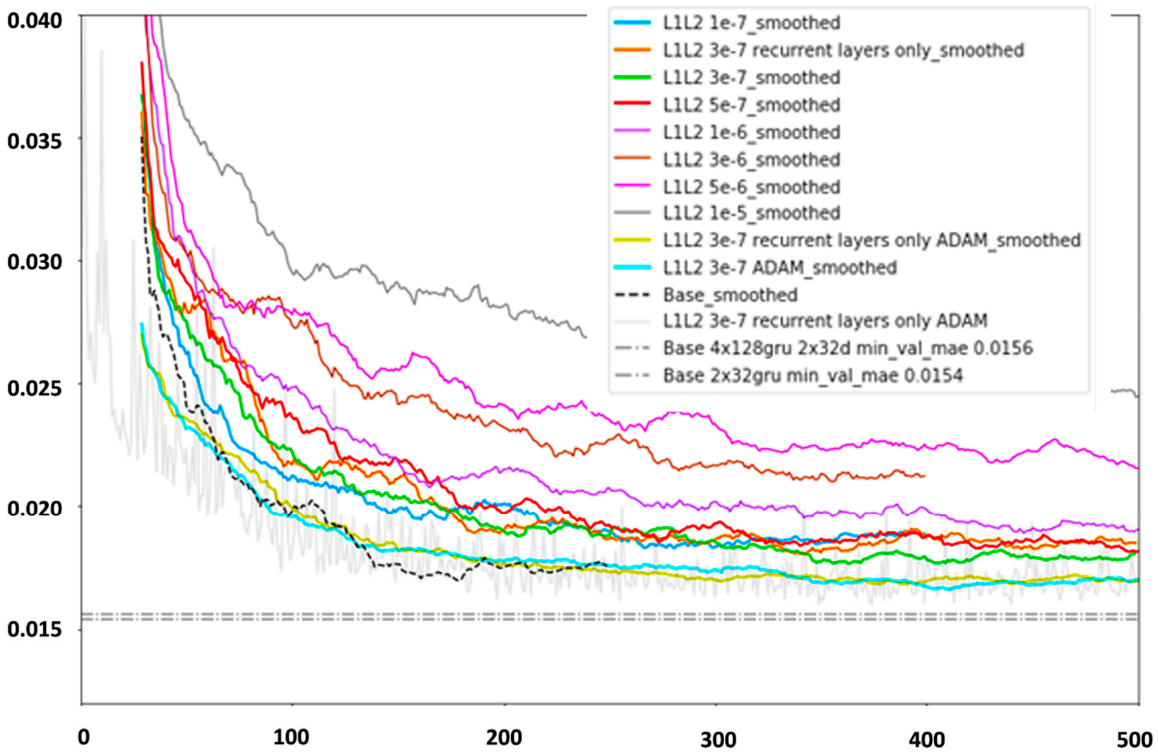

Figure 12. Deepest model (4 × 128 GRU, 2 × 32 FF) with weights regularization.

4 × 64 GRU, 1 × 32 FF Model

The next smaller base model after the largest was selected for regularization as shown in Figure 13. A smaller model has less learning capacity which also means it overfits less, increasing the chance that regularization can produce higher performance than the base model. The same L1 and L2 weights regularization was applied, using the training algorithm ADAM which showed produced better results for the larger model.

The training curve above clearly showed that regularization delayed the onset of overfitting in the base model. The model was trained with a L1 and L2 regularization parameter of 3e-7 on all the weights (bias, kernel, and recurrent) of all the recurrent layers and also those on the single fully-connected layer (bias, kernel) achieved a MAE of 2.141 which beats the base model MAE of 2.149.

However, this still does not outperform the smallest model with a MAE of 2.100. The summary of the results are shown in Table 5.

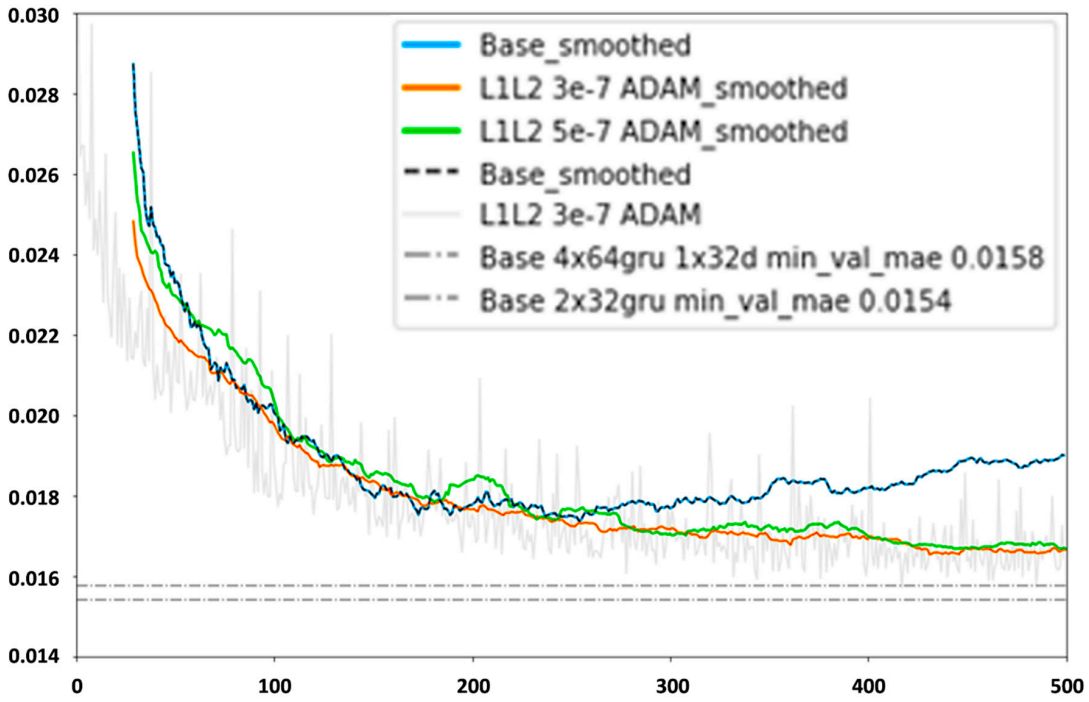

Figure 13. Second deepest model (4 × 64 GRU, 1 × 32 FF) with weights regularization.

Table 5. Summary of results (energy consumption data range is ~40–170; * performance of best recurrent neural networks model).

Network	Base Model Run #1	Base Model Run #2	Base Model Run #3	Base Model Best MAE	Regularized Model MAE
4 × 128 GRU, 2 × 32 D	2.159	2.169	2.122	2.122	2.146
4 × 64 GRU, 1 × 32 D	2.205	2.155	2.149	2.149	2.141
4 × 32 GRU	2.124	2.174	2.110	2.110	-
2 × 32 GRU	2.100	2.176	2.124	2.100*	-

6. Discussion

All the tested base recurrent network models performed better than the benchmark model (Figure 14).

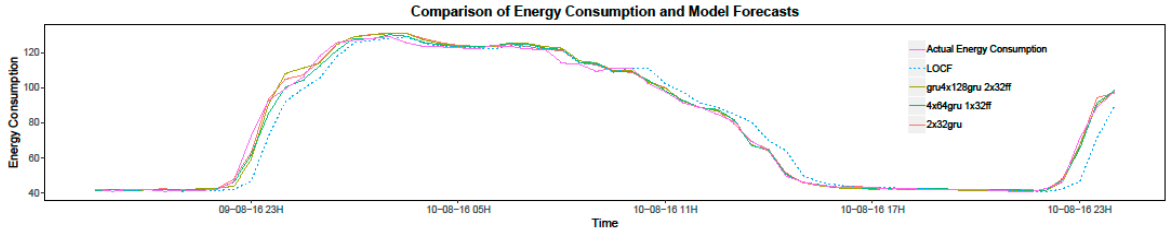

Figure 14. Performance of base models vs. LOCF benchmark; all the base model forecasts fit closer to the actual data than the LOCF forecast.

In applying regularization methods to overcome model overfitting, the study found that weights regularization did indeed delay the onset of overfitting. However, although overfitting set in later

in the model training process, this did not always bring produce higher performance relative to the base model. Weights regularization improved on the performance of the base model in only one of the two deeper models in which it was applied. In both models, regularization did not produce a higher performance than the best base model which was the smallest model.

There are two possible reasons for this finding. First, the search space for a set of parameters and hyper-parameters for a model is large and further fine-tuning may yet produce better results. New directions of exploration include tuning the learning rate such as using cyclical rates and warm restarts [42] sharing state across mini-batches during training. A new network architecture can also be explored by adding convolution 1D and max-pooling layers to pre-process the data. Simply training with more data if available will also alleviate model overfitting and may produce better results. Secondly, the other reason is that the best model may already have sufficient learning capacity for the problem and therefore any scaling up of the model will not further improve on its performance. For any specified problem, there is no known method to determine the right size of model in terms of layers and hidden nodes which is why models are developed according to the machine learning workflow.

7. Conclusions and Future Work Recommendation

This study presents a deep recurrent forecasting model while dealing with raw metered energy data with missing values of institutional buildings. The sensitivity study on data imputation and the comparison of different imputation method has provided a fundamental basis for detailed big data analysis. After data pre-processing, deep recurrent models are discussed in the domain of improving forecasting accuracy and how to address the issue of overfitting in deep learning recurrent model. Compared with the benchmarking base model where MAE of 3.983 is achieved by LOCF, and a MAE of 4.388 can be achieved by SVR, the deep recurrent model is able to achieve a MAE of 2.10. Overfitting in deep models occurs in the course of model training and will adversely impact performance if a model is over trained. In the domain of neural networks, regularization methods exist which aid deep networks in overcoming overfitting to achieve higher performance. In addition to a set of optimal network parameters and hyper-parameters, the search for a best model includes regularization parameters. While this work has raised questions regarding building energy data forecasting, it is computationally intensive to test on each model, especially when more data will be used to train the model. One future work is to look at the application of transfer learning in the building energy data forecasting to reduce the computational time when applying the same method to more buildings or meters. Another direction is to de-trend, or de-seasonalize data according to building function and occupancy patterns, which could increase the accuracy and reduce additional computational time.

Author Contributions: J.Y. contributes to the methodology, data analysis, data acquisition, supervision; K.K.T. contributes to programing and running different algorithm; S.E.L. contributes to funding acquisition, M.S. contributes to supervision, manuscript structure and state-of-art advice.

Funding: This research was funded by National University of Singapore grant number [C-296-000-029-001].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Administration U.S. Energy Information. *Monthly Energy Review*; Administration U.S. Energy Information: Washington, DC, USA, 2019.
2. Amasyali, K.; El-Gohary, N.M. A review of data-driven building energy consumption prediction studies. *Renew. Sustain. Energy Rev.* **2018**, *81*, 1192–1205. [CrossRef]
3. Deb, C.; Lee, S.E.; Yang, J.; Santamouris, M. Forecasting diurnal cooling energy load for institutional buildings using Artificial Neural Networks. *Accept. Energy Build.* **2016**, *121*, 284–297. [CrossRef]
4. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436. [CrossRef] [PubMed]
5. Siano, P. Demand response and smart grids—A survey. *Renew. Sustain. Energy Rev.* **2014**, *30*, 461–478. [CrossRef]

6. Yang, J.; Ning, C.; Deb, C.; Zhang, F.; Cheong, D.; Lee, S.E.; Sekhar, C.; Tham, K.W. K-Shape clustering algorithm for building energy usage patterns analysis and forecasting model accuracy improvement. *Energy Build.* **2017**, *146*, 27–37. [CrossRef]
7. Fallah, S.N.; Deo, R.C.; Shojafar, M.; Conti, M.; Shamshirband, S. Computational intelligence approaches for energy load forecasting in smart energy management grids: State of the art, future challenges, and research directions. *Energies* **2018**, *11*, 596. [CrossRef]
8. Ding, N.; Benoit, C.; Foggia, G.; Bésanger, Y.; Wurtz, F. Neural network-based model design for short-term load forecast in distribution systems. *IEEE Trans. Power Syst.* **2016**, *31*, 72–81. [CrossRef]
9. Liu, C.; Jin, Z.; Gu, J.; Qiu, C. Short-term load forecasting using a long short-term memory network. In Proceedings of the 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), Torino, Italy, 26–29 September 2017.
10. Amber, K.; Ahmad, R.; Aslam, M.W.; Kousar, A.; Usman, M.; Khan, M.S. Intelligent techniques for forecasting electricity consumption of buildings. *Energy* **2018**, *157*, 886–889. [CrossRef]
11. Zheng, J.; Xu, C.; Zhang, Z.; Li, X. Electric load forecasting in smart grids using long-short-term-memory based recurrent neural network. In Proceedings of the 2017 51st Annual Conference on Information Sciences and Systems (CISS), Baltimore, MD, USA, 22–24 March 2017.
12. Nichiforov, C.; Grigore, S.; Iulia, S.; Vasile, C.; Ioana, F.; Sergiu, S. Deep Learning Techniques for Load Forecasting in Large Commercial Buildings. In Proceedings of the 2018 22nd International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 10–12 October 2018.
13. Zheng, J.; Chen, X.; Yu, K.; Gan, L.; Wang, Y.; Wang, K. Short-term Power Load Forecasting of Residential Community Based on GRU Neural Network. In Proceedings of the 2018 International Conference on Power System Technology (POWERCON), Guangzhou, China, 6–8 November 2018.
14. Jiao, R.; Zhang, T.; Jiang, Y.; He, H. Short-Term Non-Residential Load Forecasting Based on Multiple Sequences LSTM Recurrent Neural Network. *IEEE Access* **2018**, *6*, 59438–59448. [CrossRef]
15. Rahman, A.; Srikumar, V.; Smith, A.D. Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. *Appl. Energy* **2018**, *212*, 372–385. [CrossRef]
16. Kong, W.; Dong, Z.Y.; Jia, Y.; Hill, D.J.; Xu, Y.; Zhang, Y. Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Trans. Smart Grid* **2017**, *10*, 841–851. [CrossRef]
17. Kong, W.; Dong, Z.Y.; Jia, Y.; Hill, D.J.; Luo, F.; Zhang, Y. Short-term residential load forecasting based on resident behaviour learning. *IEEE Trans. Power Syst.* **2018**, *33*, 1087–1088. [CrossRef]
18. Hossen, T.; Nair, A.S.; Chinnathambi, R.A.; Ranganathan, P. Residential Load Forecasting Using Deep Neural Networks (DNN). In Proceedings of the 2018 North American Power Symposium (NAPS), Fargo, ND, USA, 9–11 September 2018.
19. Shi, H.; Xu, M.; Li, R. Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Trans. Smart Grid* **2018**, *9*, 5271–5280. [CrossRef]
20. Belyaev, A.; Boldyrev, A.S.; Gorbunova, E.B.; Sinutin, E.S.; Sinutin, S.A. Energy consumption prediction in urban areas for the smart city ecosystem. In Proceedings of the Remote Sensing Technologies and Applications in Urban Environments III, Berlin, Germany, 25 October 2018.
21. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cabridge, MA, USA, 2016.
22. Steffen, M.; Alexis, S.; Thomas, B.; Martin, Z.; Jörg, S. Comparison of different methods for univariate time series imputation in R. *arXiv* **2015**, arXiv:1510.03924.
23. Schafer, J.L.; Graham, J.W. Missing data: Our view of the state of the art. *Psychol. Methods* **2002**, *7*, 14. [CrossRef]
24. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
25. Chollet, F. *Deep Learning with Python*; Manning Publications Company: Shelter Island, NY, USA, 2017.
26. Bergmeir, C.; Benítez, J.M. On the use of cross-validation for time series predictor evaluation. *Inf. Sci.* **2012**, *191*, 192–213. [CrossRef]
27. Hochreiter, S.; Schmidhuber, J. LSTM can solve hard long time lag problems. In Proceedings of the Advances in Neural Information Processing Systems, Denver, CO, USA, 6 December 1997.
28. Cho, K. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.

29. Gers, F.A.; Schmidhuber, J.; Cummins, F. Learning to forget: Continual prediction with LSTM. In Proceedings of the ICANN 99 International Conference on ANN, Edinburgh, Scotland, 7–10 September 1999; IET: London, UK, 1999; Volume 2, pp. 850–855.
30. Jozefowicz, R.; Zaremba, W.; Sutskever, I. An empirical exploration of recurrent network architectures. In Proceedings of the International Conference on Machine Learning, Lille, France, 6 July 2015.
31. Srivastava, N.; Georey, H.; Alex, K.; Ilya, S.; Ruslan, S. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
32. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Pearson Education Limited: Kuala Lumpur, Malaysia, 2016.
33. Macedo, M.; Galo, J.J.M.; de Almeida, L.A.L.; de C. Lima, A.C. Demand side management using artificial neural networks in a smart grid environment. *Renew. Sustain. Energy Rev.* **2015**, *41*, 128–133. [CrossRef]
34. Jiang, H. A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization. *IEEE Trans. Smart Grid* **2018**, *9*, 3341–3350. [CrossRef]
35. Vrablecová, P. Smart grid load forecasting using online support vector regression. *Comput. Electr. Eng.* **2018**, *65*, 102–117. [CrossRef]
36. Zhang, F.; Deb, C.; Lee, S.E.; Yang, J.; Shah, K.W. Time series forecasting for building energy consumption using weighted Support Vector Regression with Differential Evolution optimization technique. *Energy Build.* **2016**, *126*, 94–103. [CrossRef]
37. Pedregosa, F.; Gael, V.; Alexandre, G.; Vincent, M.; Bertrand, T.; Olivier, G.; Mathieu, B.; Peter, P.; Ron, W.; Vincent, D.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
38. Chollet, F. Keras. GitHub. Available online: https://github.com/fchollet/keras (accessed on 10 March 2015).
39. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. Tensorflow: A system for large-scale machine learning. In Proceedings of the 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), Savannah, GA, USA, 2–4 November 2016.
40. Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera: Neural Netw. Mach. Learn.* **2012**, *4*, 26–31.
41. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
42. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.