A supervised-learning trial to predict hourly urban electricity load

Group 73
Jiachen Mao, Yihang Sui, Mazen S. Danaf
{maoj, suiyh, mdanaf}@mit.edu

December 12, 2017

Abstract

In this final project, we intend to predict the hourly urban electricity load in downtown Abu Dhabi (UAE) via machine learning methods. The methods include boosting tree, support vector regression (SVR), and Long Short-Term Memory (LSTM) neural network. Considering the availability of the power load data in reality, we test the developed models on two individual tasks. The results show that the power load is largely dependent on previous load and weather values, so the time-series analysis is highly recommended if corresponding data can be fully available. Besides, there seems to be a stronger relationship between the urban electricity load and weather condition during summer than during winter in downtown Abu Dhabi. The identified findings could be the subject of additional research to gain a better performance of power load forecasting in the future.

Keywords: Electric load forecasting, Boosting, SVR, Genetic algorithm, LSTM

1 Introduction

Modern smart grid is an advanced technology that takes advantage of sensing, communicating, and computing infrastructures to improve the reliability, efficiency, and security of the electric power grid. One recognized benefit is that *power load forecasting* could be carried out based on the measurements, which will be very helpful in constructing various strategies (e.g. fault detection and diagnosis, demand side management, etc.) to improve energy system, especially in urban areas.

However, although load forecasting has been widely studied for decades, there are still some challenges. One concern is that even if the power demand seems like a univariate time series, it is subject to various factors with discriminative influences. Another challenge lies in the fact that it is not easy to obtain optimal forecasting settings. The time granularity of metering is pretty flexible in modern smart grids. More often than not, we cannot fully obtain the sequential data at once. By investigating what kind of time granularities and dependences leads to an accuracy gain, we can not only provide empirical guidelines for forecasting but also evaluate whether a model can work well in existing grid settings.

Existing methods of developing the load forecasting models include physics-based methods, statistical methods, grey-box methods, etc. In particular, modern machine learning algorithms present satisfactory performances in classification and regression due to dramatic increases in the hardware and software computing power. Currently, there are three popular machine-learning models used in load forecasting tasks, namely decision tree [1], support vector regression [2], and neural network [3].

Based on the assumption that the future power demand may have similar trend and/or distribution as the observed history, some researchers start to take the recurrent neural network (RNN) with memory capability into consideration. RNN with memory capability has presented its success

in many research areas, such as speech recognition, language processing, etc. In particular, the Long Short-Term Memory (LSTM) neural network has been recently applied in electric load forecasting tasks [4].

It is interesting to explore the effectiveness of these ideas in power load forecasting, which has motivated this work. In this final project, we have tried some modern machine learning algorithms to predict hourly urban electricity load via the weather data during calendar year 2010 in downtown Abu Dhabi (UAE). In particular, we evaluate the performances of the boosting tree, support vector regression, and Long Short-Term Memory (LSTM) neural network using the same dataset on two individual tasks. We experiment different settings to evaluate the models. The results are expected to provide some new directions to power load forecasting.

2 Case study

In this study, the regression model is established using the underlying data of the electricity load and weather condition in the urban area of Abu Dhabi (UAE) during calendar year 2010 (see **Figure 1**). Abu Dhabi has been developed rapidly over the past 60 years. Its climate resembles that of the arid and semi-arid zones, and is characterized by a very hot summer from July to September and a mild winter from December to February. The city experiences a typical tropical climate with generally high temperatures and insufficient rainfall throughout the year. It has been reported that, during 2010, the industrial and agricultural load accounted less than 16% of the total load in the Emirate [5], while the electricity load in Abu Dhabi is mostly covered by residential/commercial buildings. Thus, the focus of this study is on building electricity use at the urban scale.

Substation-level hourly building electricity load data were measured by the Supervisory Control and Data Acquisition (SCADA) system in Abu Dhabi Emirate. A subset of the substations which are all within the Abu Dhabi Municipality was selected that presents an even better proxy for the building loads in aggregate. The annual electricity demand was about 6225 GWh, peaking at 1096 MW during summer. In the meantime, hourly weather data was monitored by the Masdar city's weather station throughout the calendar year 2010. Based on a correlation analysis, six meteorological parameters are considered and summarized in **Table 1**.

This final project is conducted based on the hourly data. As shown in **Figure 2**, the hourly values of the urban electricity load and the dry bulb air temperature present mostly consistent patterns throughout the year. This somewhat increases our confidence of using the weather condition to predict the electricity load via machine learning algorithms.

Table 1: Summary of the meteorological parameters.

Meteorological parameter	**Range**
Dry bulb air temperature [°C]	[4.1, 47.5]
Dew point air temperature [°C]	[-6, 30.2]
Relative humidity [%]	[8, 98]
Atmospheric pressure [kPa]	[98.9, 102.1]
Global horizontal illuminance [lux]	[0, 116300]
Wind speed [m/s]	[0, 12]

3 Method

Our method is generally to predict the hourly power load by modeling the relationship between the power load and relevant weather features. We do not expect each feature to be effective and equally contributing to the prediction. Based on the dataset, we develop three modern machine learning models to perform forecasting on two individual tasks.

Task 1 is to predict the hourly power load using the weather condition at the same time, while *Task 2* is to predict the hourly power load using the power load and weather condition at previous time points. In reality, we cannot easily obtain the sequential urban electricity load data. Thus, *Task 1* is designed to independently predict the hourly load merely using the weather data at the same time. However, once we have complete sequential power load data, we can perform time-series analysis on power load forecasting (i.e. *Task 2*).

The first two methods, boosting tree and support vector regression, assume that the hourly power load data is independent of each other. Thus, we totally rely on the weather condition to predict the power load at one time point (*Task*

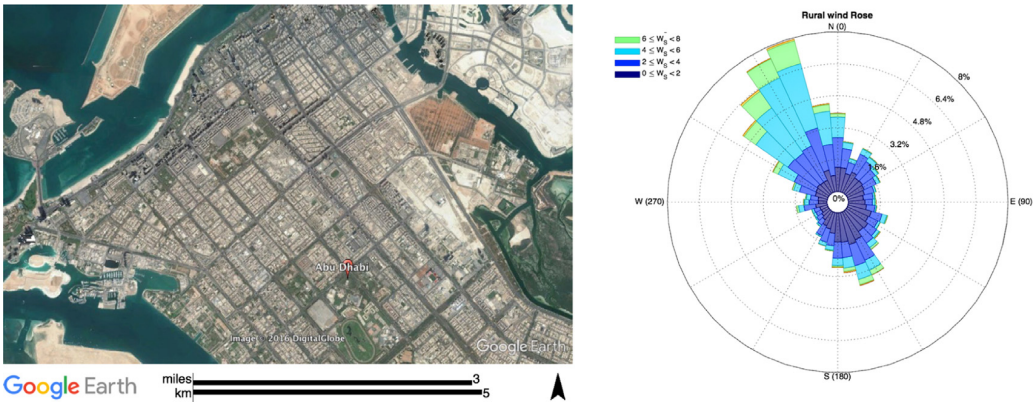

Figure 1: Downtown Abu Dhabi and the rural wind rose in 2010 (mainly from Ref. [6]).

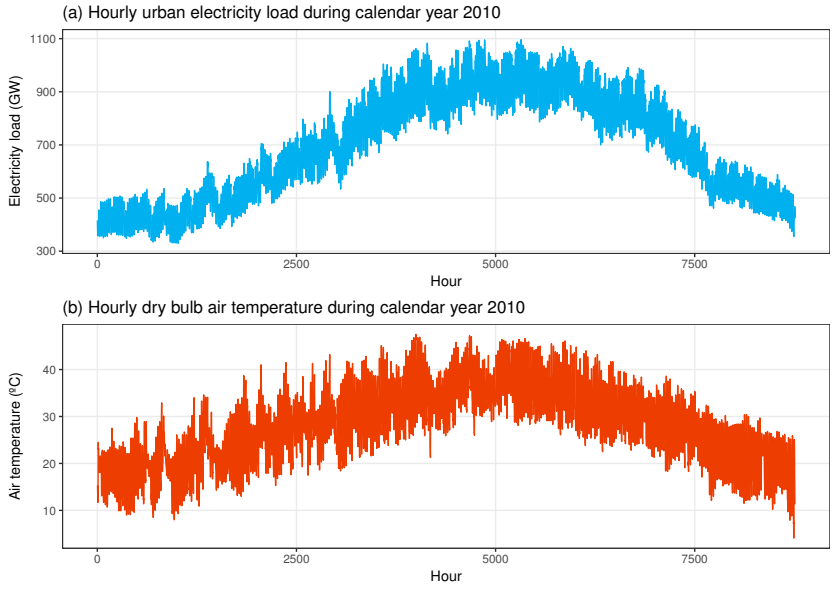

Figure 2: Hourly data during calendar year 2010: (a) urban electricity load; (b) dry bulb air temperature.

1). The third method, Long Short-Term Memory (LSTM) neural network, intends to capture the time-series characteristics of the measurements. Thus, we try to incorporate the historical observations into the development settings (*Task 2*). The details of each method are explained and specified in the following subsections.

3.1 Boosting tree

Boosting tree is a supervised learning predictive model applied for classification and regression. In general, boosting tree builds a series of decision trees in a step-wise manner. In each step, it incorporates one tree, while maintaining the existing tree(s) unchanged. The new tree is the optimal tree that minimizes the loss function or model residual. Thus, boosting tree can be seen as an ensemble of weaker learners, which somehow becomes a better learner.

In this study, we use the gbm package in R to perform the boosting tree method. Two hyperparameters, the number of nodes in a leaf and the number of trees, are tuned via cross validation. The minimum number of observations per leaf is varied between 5 and 30. The maximum number of trees is set as 5000 in order to avoid overfitting.

3.2 Support vector regression

Support vector machine (SVM) was developed by Vapnik [7] within the framework of statistical learning theory. Based on the structural risk minimization inductive principle, SVM can usually achieve a higher generalization performance than the traditional shallow neural networks in solving many machine-learning problems. Due to the power of kernel functions, training SVM is equivalent to solving a linearly-constrained quadratic programming problem, so we can usually achieve

a global optimum.

In this study, we use ϵ-SV regression from the LibSVM [8]. Suppose that the training data is given by $\{(x_1, y_1), ..., (x_l, y_l)\}$, where x_i is d-dimensional. The overall goal is to find a function $f(x_i)$ that can have at most ϵ deviations from the actual target y_i for all the training data and is simultaneously as flat as possible. That is to say, we do not care about the errors as long as they are less than ϵ, but will not accept any deviation larger than ϵ.

The SVR uses kernel functions to solve non-linear problems. In this study, we select the Gaussian radial basis function (RBF) as the kernel function, since it is very effective in handling non-linear problems. Two hyper-parameters, the regularization constant (c) and the inverse width of Gaussian RBF (g), need to be optimized in order to precisely assess the performance on regressing unknown data and to prevent overfitting. Usually the efficiency of grid search is low because it computes the performance with all the (c, g) to obtain the performance surface. So in this study, the optimization of hyper-parameters is performed using some heuristics, namely the *genetic algorithm*.

Genetic algorithm (GA), developed by Holland [9], is a stochastic optimization algorithm derived from an analogy with the evolution theory of Darwin. It is a powerful optimization method able to resolve every problem provided the convexity of the given function. The GA can give several final solutions to a complicated problem with a large number of inputs, so the difficulty in generating an optimal solution through GA lies in the adjustment of the algorithm. In this final project, to obtain good optimization results from GA, the population size, number of generations, crossover probability, mutation chance, and proportion of elitism are set to be 20, 100, 0.7, 0.01, and 0.1, respectively.

3.3 LSTM neural network

Long Short-Term Memory (LSTM) neural network, developed by Hochreiter and Schmidhuber [10], is a variant of recurrent neural network (RNN) that computes the new hidden state given the previous hidden state and new input. Since the traditional RNN suffers from gradient exploding or vanishing during training, the LSTM adopts a special form of memory block to avoid those challenges.

As shown in **Figure 3**, there is a memory cell with a self-recurrent connection and three gate functions to control the information flow. The input gate (i) governs the input flow into the memory cell, while the output gate (o) decides to what degree to output new activations. In addition, the special forget gate (f) controls a self-recurrent connection, enabling this cell to manipulate the previous state. In summary, the general principle of the memory block is given as follows:

$$i_t = a_i(W_i x_t + U_i h_{t-1}) \tag{1}$$

$$o_t = a_o(W_o x_t + U_o h_{t-1}) \tag{2}$$

$$f_t = a_f(W_f x_t + U_f h_{t-1}) \tag{3}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot a_c(W_c x_t + U_c h_{t-1}) \tag{4}$$

$$h_t = o_t \odot a(c_t) \tag{5}$$

where i_t, o_t, f_t, and c_t are the activation vectors at time t of the input gate, output gate, forget gate, and memory cell. W/U and a are the corresponding weight and activation function, and their subscripts indicate the connection. $\odot$ denotes an element-wise product operation with a gate value.

In this study, to perform time-series prediction with LSTM, we frame the supervised learning problem as predicting the hourly power load at the current hour (t) given the power load measurements and weather conditions at multiple prior time steps. Suppose that the training data is given by $\{(x_1, y_1), ..., (x_l, y_l)\}$, where x_i is d-dimensional. The overall goal is to find a prediction $g(x_t)$ based on $\{(x_{t-h}, y_{t-h}), ..., (x_{t-1}, y_{t-1})\}$, where h is the number of lag hours, e.g. 3 hours, 6 hours, and 12 hours.

We use the LSTM algorithms in the Keras sequential model API [11] to solve multivariate time-series forecasting problems. As default, we select mean squared error (MSE) for the loss function, sigmoid activation function for the hidden layer, and linear function for the output layer. Various hyper-parameters, including the batch size and the max epoch, need to be tuned in order to improve the forecasting performance and to prevent overfitting.

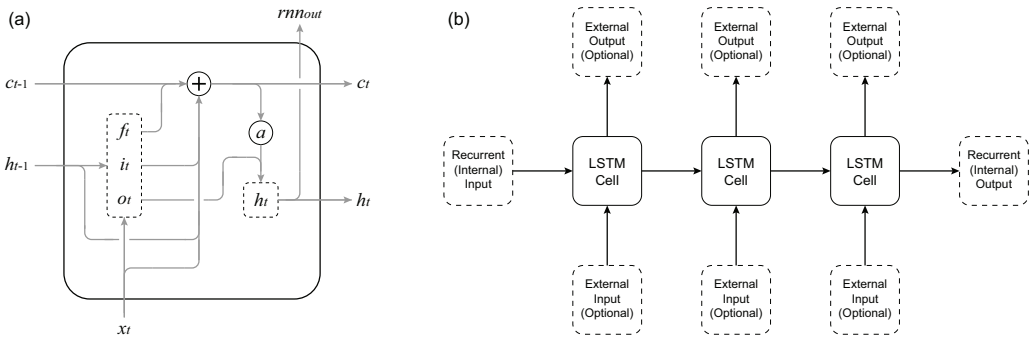

Figure 3: LSTM neural network: (a) An LSTM cell; (b) An LSTM neural network unfolded in time of the computation involved in its forward pass.

3.4 Evaluation metric

In order to evaluate the performances of the developed models, we use the root mean squared error (RMSE) and mean absolute percentage error (MAPE). The closer these values are to zero, the better the forecasting performance is. These two indices are given by:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i}^{n}(A_i - F_i)^2} \tag{6}$$

$$\text{MAPE} = \frac{100}{n}\sum_{i}^{n}\left|\frac{A_i - F_i}{A_i}\right| \ [\%] \tag{7}$$

where n is the number of testing values, A_i and F_i denote the actual and forecast value at time i, respectively.

4 Result and discussion

For each method with the corresponding task, we randomly select 60% of the data for training and cross validation, while the remaining 40% of the data is used as the testing set. In order to evaluate the uncertainty during forecasting, we run 10 trails for each method and calculate the corresponding statistics (i.e. mean and standard deviation). The following subsections report the prediction results from different methods and different tasks for evaluating the performance in the current case study.

4.1 Task 1

In *Task 1*, we try to predict the hourly power load merely using the weather condition at the same time. Both in boosting and SVR, the hyper-parameters are optimized via 5-fold cross validation during training.

In general, boosting is very sensitive to the number of trees we use. This is what we expected, because the subsequent trees should correct the bad predictions made by previous trees. In this study, we use up to 5000 trees. The running time would become excessive beyond that. The resulting RMSE and MAPE over the whole testing set are 93.20 $\pm$ 16.28 and 5.89 $\pm$ 0.10%, respectively.

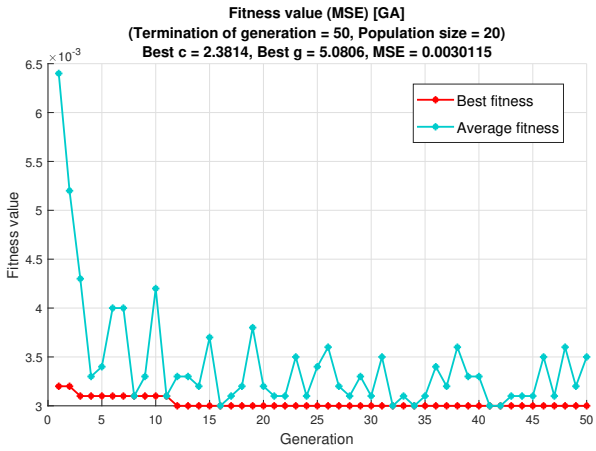

Figure 4: Fitness value during the optimization process of the hyper-parameters c and g in the SVR using GA.

On the other hand, one optimization process of the SVR hyper-parameters using GA is shown in **Figure 4**. The optimal c value is 2.3814, and the optimal g value is 5.0806. After the best (c, g) is found, the training set is trained again to generate the final SVR model. The resulting RMSE and MAPE over the whole testing set are 41.53 $\pm$ 12.63 and 4.96 $\pm$ 0.09%. In general, the SVR performs slightly better than the boosting tree in our current settings.

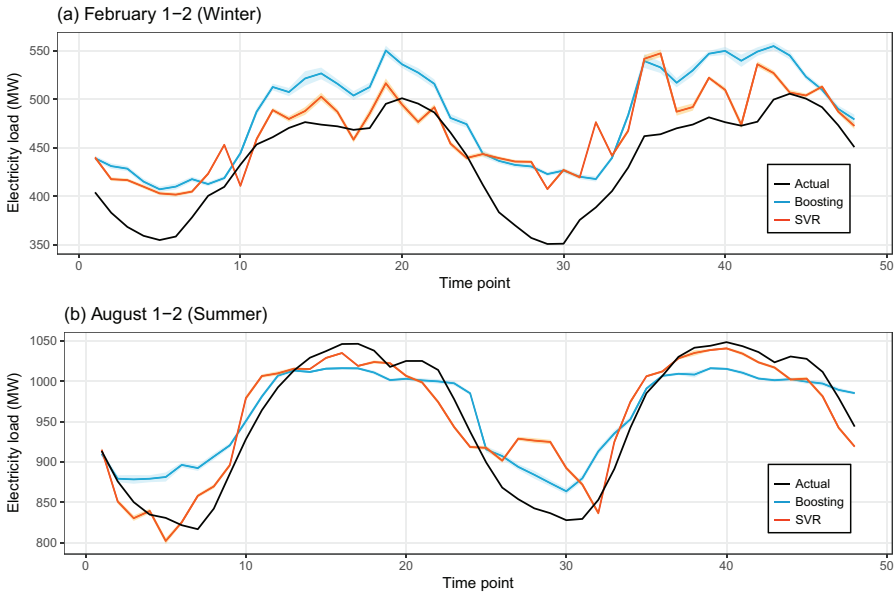

Figure 5: Forecasting results from the boosting tree and SVR on *Task 1*. The solid line represents the predicted values of the mean. The shaded area represents the predicted values ranging within one standard deviation.

Finally, we plot the forecasting results using the data (excluded from training) of February 1-2 (winter time) and August 1-2 (summer time). As shown in **Figure 5**, both boosting tree and SVR can generally capture the trend as well as peaks and valleys better in summer than in winter. So, there may be stronger relationships between the weather condition and electricity load during summer than during winter in Abu Dhabi.

In Abu Dhabi, there is nearly no heating mode and only the cooling system is operated throughout the year to maintain the predetermined indoor environment. Due to the extremely hot weather condition in Abu Dhabi during summer, the HVAC-related energy consumption dominates other energy uses. As a consequence, the weather condition can significantly indicate the urban electricity use in summer. During winter, when the difference between the outdoor and indoor air temperatures turns smaller and the HVAC-related energy consumption becomes comparable to the remaining energy uses, other parameters (such as the equipment/lighting load) start playing more important roles. Thus, in order to further improve the performance, we might need to incorporate more input variables into our current setting.

It can be observed that the predicted curves from both boosting tree and SVR are not very smooth over the time. In *Task 1*, we assume that the hourly power load forecasting is independent of each other. This largely limits the model's ability to capture the load pattern. It is hence natural to incorporate the time-series characteristics into the forecasting setting, which is *Task 2* in the following subsection.

4.2 Task 2

In *Task 2*, we try to incorporate the historical hourly power load observations into the model development. When training the LSTM model, we use the observations from previous 3 hours, 6 hours, and 12 hours as inputs and implement the algorithms from the Keras LSTM package.

Table 2: Optimal hyper-parameter set over 10 LSTM trails with multiple lag time steps.

Lag time step	**3 hr**	**6 hr**	**12 hr**
# of layers	1	1	1
# of neurons	50	50	50
Activation (hidden)	sigmoid	sigmoid	sigmoid
Activation (output)	linear	linear	linear
Optimizer	adam	adam	adam
Batch size	1	5	5
Epochs	50	50	75
Loss function	MSE	MSE	MSE
Shuffle	TRUE	TRUE	TRUE
Training RMSE	6.28	5.86	5.43
Testing RMSE	8.86	9.02	8.44
Training MAPE [%]	0.77	0.71	0.66
Testing MAPE [%]	0.87	0.84	0.79

Due to the limited size of our database, we use a two-layer LSTM neural network to avoid

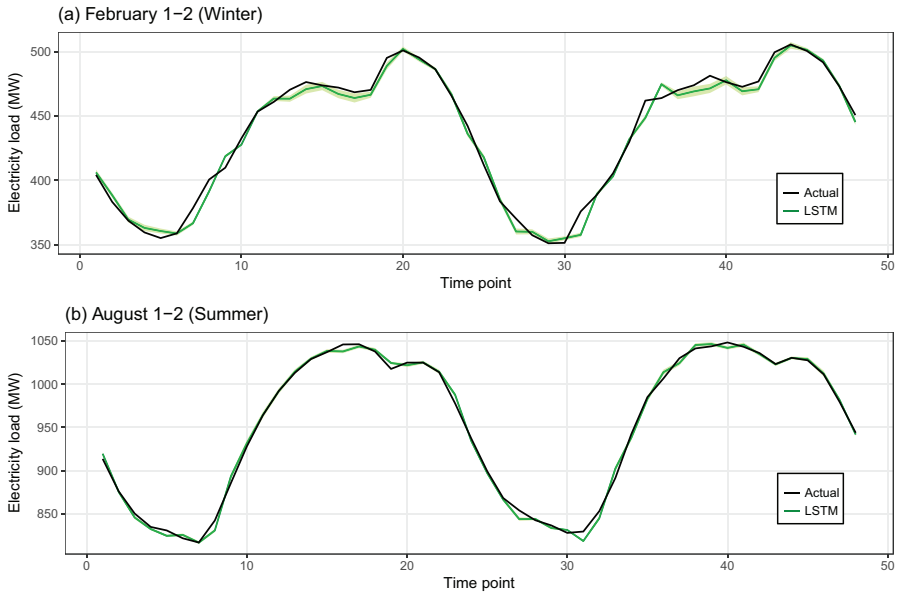

Figure 6: Forecasting results from the LSTM on *Task 2*. The solid line represents the predicted values of the mean. The shaded area represents the predicted values ranging within one standard deviation.

Table 3: Sample LSTM algorithm performance on multiple lag time steps over 10 trails.

Lag time step	**3 hr**		**6 hr**		**12 hr**	
Evaluation metric	Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev.
Training RMSE	6.75	0.52	5.92	0.06	5.91	0.75
Testing RMSE	9.53	0.56	**9.16**	**0.21**	9.13	1.07
Training MAPE [%]	0.85	0.09	0.71	0.01	0.73	0.13
Testing MAPE [%]	0.95	0.08	**0.86**	**0.03**	0.86	0.14

overfitting. In general, the LSTM is very sensitive to many hyper-parameters. From multiple runs, the optimal activation in the hidden layer is sigmoid, which is what we expected. Other hyper-parameters for the LSTM are also tuned according to the evaluation metrics. The corresponding optimal values and performances are shown in **Table 2**.

Next, we intend to select the optimal lag time step based on the algorithm performance and efficiency. When the lag time step is small, i.e. 3 hours, the model may not be able to easily capture the hourly load characteristics. On the other hand, when the lag time step is large, i.e. 12 hours, the model may demonstrate more of its fluctuation trend, which might be a hindrance to model accuracy and efficiency. There is hence a trade-off between the lag time step and the model capability as well as algorithm efficiency.

Based on **Table 3**, we select a 6-hour time step to perform the time-series forecasting via LSTMs. Here we use a batch size of 5 and a max epoch of 50. The running time would become excessive when the batch size decreases and the max epoch increases. The resulting RMSE and MAPE over the whole testing set are 9.16 ± 0.21 and $0.86 \pm 0.03\%$, respectively. In general, the LSTM is more accurate than the boosting tree and SVR, which makes sense because the current load prediction incorporates the hourly load and weather values at previous time points.

Finally, we plot the forecasting results using the data (excluded from training) of February 1-2 (winter time) and August 1-2 (summer time). As shown in **Figure 6**, the LSTM is able to capture the trend as well as peaks and valleys significantly. On the other hand, the predicted values are also closer to the actual values in summer than in winter, which has already been seen in *Task 1*. The predicted curves from LSTM are much smoother over the time, compared with those from boosting tree and SVR in *Task 1* (see **Figures 5 and 6**). This indicates that the power load has a strong time-series dependence.

In reality, if we can obtain complete sequential urban electricity load data (which is kind of not easy), the time-series analysis using LSTM neural network is highly recommended.

5 Conclusion

In this final project, we propose and apply three machine learning models to predict the hourly urban electricity load. The models are validated using the actual measurements of hourly system load for the city of Abu Dhabi, UAE.

Considering the availability of the power load data in reality, we test the models on two tasks. *Task 1* is designed based on the assumption that the urban electricity load data is not fully known over the whole horizon. We use the boosting tree and support vector regression (SVR) to predict the hourly power load merely using the weather data at the same time. We then switch to *Task 2*, where we assume that the sequential power load data can be fully obtained. In such case, the Long Short-Term Memory (LSTM) neural network is applied to perform time-series analysis.

In general, the performances of *Task 2* are largely better than those of *Task 1*. This indicates that the power load is strongly dependent on previous load and weather values. In reality, if we can obtain the complete sequential load data, the time-series analysis is highly recommended. For *Task 1* in this specific case study, the SVR (optimized via genetic algorithm) performs slightly better than the boosting tree. It is interesting to observe that the forecasting performances are generally better during summer than during winter. Hence, there may be a stronger relationship between the urban electricity load and weather condition during summer than during winter in downtown Abu Dhabi.

Division of labor

This work is funded under the Cooperative Agreement between the Masdar Institute of Science and Technology, Abu Dhabi, UAE and the Massachusetts Institute of Technology, Cambridge, MA, USA. We hereby acknowledge the support of the Executive Affairs Authority (EAA) of Abu Dhabi, UAE for providing the data. In addition, we would like to thank our teaching assistant, Zhi Xu, for the valuable advice throughout the semester.

For this final project during the semester, the project idea and corresponding data were initiated by Jiachen Mao. For the models specified in the report, the decision tree was performed by Mazen S. Danaf, the SVR optimized via genetic algorithm was performed by Jiachen Mao, and the LSTM neural network was performed by Yihang Sui. This report was mostly written by Jiachen Mao and Yihang Sui.

References

[1] Bansal A, Rompikuntla SK, Gopinadhan J, Kaur A, Kazi ZA. Energy Consumption Forecasting for Smart Meters. arXiv preprint arXiv:1512.05979 (2015).

[2] Hong W-C. Electric load forecasting by seasonal recurrent SVR (support vector regression) with chaotic artificial bee colony algorithm. *Energy* 36.9 (2011): 5568-5578.

[3] Gajowniczek K, Zbkowski T. Short term electricity forecasting using individual smart meter data. *Procedia Computer Science* 35 (2014): 589-597.

[4] Cheng Y, Xu C, Mashima D, Thing VL, Wu Y. PowerLSTM: Power Demand Forecasting Using Long Short-Term Memory Neural Network. In: *International Conference on Advanced Data Mining and Applications* (pp. 727-740), Springer, Cham, November 2017.

[5] SCAD. Statistical Yearbook of Abu Dhabi. Statistics Center Abu Dhabi, 2013.

[6] Afshari A, Friedrich LA. Inverse modeling of the urban energy system using hourly electricity demand and weather measurements, Part 1: Black-box model. *Energy and Buildings* (2017).

[7] Cortes C, Vapnik V. Support-vector networks. *Machine Learning* 20.3 (1995): 273-297.

[8] Chang C-C, Lin C-J. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3 (2011): 27.

[9] Holland JH. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, 1992.

[10] Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Computation* 9.8 (1997): 1735-1780.

[11] Chollet F. Keras. 2015.