UNIVERSITY OF CALIFORNIA,
IRVINE

Prediction of Building Power Loads Using Statistical and Machine Learning Methods: A Case Study of a LEED-Certified Institutional Building

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Mechanical Engineering

by

Dongyang Li

Dissertation Committee:
Professor Yun Wang, Chair
Professor Manuel Gamero-Castaño
Assistant Professor Penghui Cao

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

The author would like to thank the UCI central plant, Altura Associates, and Dongheng Jing for their support in accessing data and building the neural network model.

ABSTRACT OF THE THESIS

Prediction of Building Power Loads Using Statistical and Machine Learning Methods: A Case Study of a LEED-Certified Institutional Building

by

Dongyang Li

Master of Science in Mechanical Engineering

University of California, Irvine, 2020

Professor Yun Wang, Chair

In response to the growing challenge of energy and power management caused by increasing implementation of volatile sustainable energy sources, a case study of forecasting building electric loads using statistical and machine learning methods is conducted for a multi-purpose LEED-certified institutional building on the UC Irvine campus. Four data-driven methods, which require no detailed building information and strong building energy knowledge, are employed and compared, including the polynomial regression, Autoregressive Integrated Moving Average (ARIMA), TBATS (Trigonometric seasonal formulation, Box-Cox transformation, ARMA errors, Trend, and Seasonal components), and backpropagation Artificial Neural Networks (ANN). These models are investigated to satisfy the ASHRAE standards and optimize the prediction performance. A full year of hourly electric load and meteorological data of the building in 2019 was obtained using the existing meters for this data-driven study. Root Mean Square Error (RMSE), Coefficient of Variance (CV), Mean Absolute Percentage of Error (MAPE) and R^2 are calculated as evaluation criteria to compare the performances of these data-driven methods in terms of prediction accuracy. Akaike's Information Criteria (AIC) is introduced as a guideline to determine the optimal model for several prediction models. The

polynomial regression is performed using MATLAB and is shown capable of only data fitting instead of forecasting when the total hour is used as the independent variable. When using the daily and weekly data, the polynomial regression method fails for forecasting. For the whole-month data, ARIMA, TBATS, and ANN methods are used to predict hourly power load in the next month with Python. The ARIMA model shows relatively low accuracy, indicating that it is unable to handle multiple seasonlities in the data. TBATS shows a substantially improved accuracy and satisfactory prediction. The backpropagation ANN is also conducted with its configuration, including inputs, number of hidden layers and neurons, optimizer, and activation functions, optimized after extensive testing. Different sets of training data are examined for both TBATS and ANN. The ANN's forecasting accuracy is found to be about 5~20% better than TBATS' when only using one month's data for training. The residuals of these forecasting methods show there could be information uncaptured in forecasting. It is speculated that operation and activity schedules can serve as additional inputs for the ANN to achieve better forecasting accuracy.

CHAPTER 1. INTRODUCTION

The building sector consumes a large portion of energy worldwide. Approximately 20% of global delivered energy consumption was used in the building sector in 2018, which includes residential and commercial structures [1]. The residential and commercial sectors accounted for up to 40% of total U.S. energy consumption in 2018. The U.S. Energy Information Administration (EIA) predicts that global energy consumption by buildings will possibly continue to grow by 1.3% or even a few times more per year on average from 2018 to 2050, depending on different countries.

At the meantime, energy consumption and emissions reduction has become the relatively universal goals for policy makers worldwide in response to emerging environmental issues such as global warming and pollution. State of California ambitiously aims for the development of zero net energy buildings. All new residential and commercial construction will be zero net energy by 2020 and 2030 respectively [2]. Sustainable energy sources have become widely implemented in recent years to achieve on-site renewable energy generation goals. However, these energy sources are volatile in nature, rendering energy management increasingly challenging as their proportion in the energy generation profile grows.

In particular, clean and renewable power sources, such as photovoltaics (PV), wind power, polymer electrolyte membrane (PEM) fuel cells and flow batteries, are emerging technologies for building power applications [3]. PEM fuel cells produce electric power, water, and waste heat, which can benefit the multiple needs of buildings [4]. However, the water and thermal management needs to be carefully controlled in practice. Distributed PEM fuel cell systems can serve as heat-power co-generation and uninterruptable power supply for the residential and commercial sectors [5]. Ham et al. proposed a simplified PEM fuel cell simulation model for

commercial cogeneration systems and validated the model by comparing observed data from experiments with model predictions [6]. Ellamla et al. reviewed the state of fuel cell cogeneration systems for the residential sector and believed they are the most beneficial and promising technology for cogeneration [7]. In addition, flow battery has been implemented as an advanced energy storage solution in the commercial and residual sector with significant efforts in modeling and analysis that aim to widen the adoption of these battery technologies [8] [9]. Kear et al. reviewed the state of development of the all-vanadium redox flow battery for energy storage and found a significant level of battery commercialization in terms of grid load leveling, utility-scale renewable electricity generation and distributed-energy/remote area power supply [10]. A vanadium redox flow battery model was developed and validated through experiments for distributed storage implementation in residential building energy systems by D'Agostino et al [11].

All of these renewable and clean energy sources require proper design and control in order to meet the building energy demand, which accentuates the significance of energy load forecasting.

Building energy consumption prediction is essential for energy planning, management, and sustainability. For example, the wind and solar PV energies generate power varying with time and weather, which need to be stored properly in batteries or hybrid with a power generator such as fuel cells for building energy demand that is relatively constant. PEM fuel cells and batteries are dynamic systems with dynamics determined by their internal physics and component dimensions. In practice, they require proper power management to optimize their reaction, operation, and efficiency [12] [13].

There are two main approaches for building energy consumption prediction: physical modelling approach and data-driven approach. Physical modelling method constructs a building

model based on fundamental thermodynamic rules. A wide range of building energy simulation software employs physical modelling approach and is popular among the industries, including EnergyPlus, OpenStudio, eQuest, etc. Detailed inputs related to building characteristics and meteorological information such as lighting density, operation schedules and dry-bulb temperature are imported to this type of software for building energy simulation. Such inputs are often not available or not accessible in time to the involved engineers or technicians [14]. In addition, on-site commissioning is required to obtain detailed and accurate building information. The building models are validated through comparing simulation results with utilities bills or real-time data capture with installed metering devices. This process is referred to model calibration and usually consumes considerable time and manpower, strongly depending on the experiences and expertise of the tasked engineers.

On the other hand, data-driven method does not require detailed parameters of simulated buildings or the time-consuming calibration process for load prediction. Only historical energy consumption data is necessary. Data-driven energy consumption forecasting method has been extensively researched in recent years. Fang et al. used a simple regression model added with weekly rhythm of heat consumption and the seasonal autoregressive integrated moving average model (SARIMA) with exogenous variables that accounted weather factors to forecast hourly heat demand for the city of Espoo in Finland [15]. Bagnasco et al. proposed a backpropagation multi-layer artificial neural network (ANN) to predict the electrical energy consumption of a hospital facility in Italy [16]. Wang et al. employed several data-driven methods including multiple linear regression, adaptive linear filter algorithms and Gaussian mixture model regression to predict the hourly energy usage of a U.S. Department of Energy (DOE) reference building and an existing office building in Des Moines, Iowa [17]. Solomon et al. developed a

predictive computer model using Support Vector Machine Regression (SVMR) to improve the energy efficiency of a Manhattan skyscraper [18]. Brozyna et al. showcased the capabilities of the TBATS models to forecast the electric energy demand of a Polish internet service in hourly, daily, and monthly time resolution with multiple present seasonalities [19]. Edwards et al. compared forecasting accuracy of various methods including linear regression, feed forward neural network, support vector regression, and least square vector machine for hourly electrical consumption data from an ASHRAE data set and real residential buildings [20]. Yokoyama et al. adopted a global optimized backpropagation ANN to predict the cooling demand of a commercial building with the two-phase method, where in the first phase the temperature and relative humidity are initially predicted to enable energy demand prediction in the second phase [21]. Fan et al. proposed a semi-parametric addictive model that estimated the relationships between energy demand and exogenous inputs for short-term half-hourly electric demand forecast for power systems in Australia [22]. Zhao et al. statistically analyzed variable refrigerant volume system in office buildings located in several cities in East China and employed ARIMA, SVM, and ANN to develop a case study for the office building in Shanghai by predicting hourly energy consumption based on three months' data of the building's variable refrigerant volume system [23]. Catalina et al. proposed a polynomial regression model to predict the monthly heating energy demand of a Romanian building based on the relationship between weather data and heating energy consumption found by analyzing TRNSYS building model simulations of buildings located in multiple cities including Moscow, Nice, and Burcharest [24]. An ANN model based on the Levenberg-Marquardt algorithm was adopted by Leung et al. to predict hourly electrical power demand of the cooling system in a building located in Hong Kong and high prediction accuracy was achieved with the help of occupancy space electrical power

demand as one of the input parameters [25]. Learning window reinitialization was applied to AR and SVM forecast models of multiple buildings as a post-processing method by Borges et al. [26]. Neto et al. compared feed-forward ANN forecast models with different inputs including temperature, humidity, and solar radiation with an EnergyPlus building energy simulation model using a building on University of São Paulo campus as a cast study [27]. Two Long Short Term Memory (LSTM) based neural networks with standard and sequence to sequence architecture respectively were investigated by Marino et al. in terms of electric energy consumption forecasting of hourly and minutely resolution based on a data set from a residential building [28]. Ekici et al. examined the feasibility and applicability of Adaptive Network Based Inference System (ANFIS) model to forecast heating and cooling energy consumption of buildings located in a cold region located in Turkey [29]. Zhang et al. developed an electric energy consumption forecasting model of half-hourly resolution for an institutional building in a university campus in Singapore with weighted SVM and introduced differential evolution algorithm to determine the weights [30]. The effects of temporal and spatial monitoring granularity on forecasting accuracy of the forecasting model for a multi-family residential building in New York were investigated by Jain et al. using SVM [31]. In addition, there are many works in the field of renewable energy employing machine learning methods that can be incorporated into building energy prediction [32–34].

This thesis aims to conduct a case study of a commercial building on the campus of University of California, Irvine (UCI), by using four time series forecasting models, including polynomial regression, ARIMA, TBATS and ANN. The main objective is to a suitable model for building electric load forecasting that complies the model calibration guidelines of ASHRAE. On top of satisfying the ASHRAE requirement, the performances of the forecasting models will be

optimized in terms of forecasting accuracy. The models are also expected to be computed within a reasonable time in order to be practically usable.

1.1 BUILDING INTRODUCTION

Biological Science III (BioSci3) building on UCI campus as shown in Figure 1 was selected for the case study. BioSci3 is a multi-purpose laboratory building with a large lecture hall and a

Figure 1. BioSci3 building on UCI campus

classroom capable of accommodating 401 and 32 people, respectively. The building has a total area of roughly 44,000 square feet, with about 70% of the area available for Photovoltaics (PV) panels installation on rooftop. An estimated capacity of 50 kW PV panels has already been installed.

BioSci3 is one of the most energy efficient laboratory buildings on campus as part of the campus Smart Labs Program [35]. In 2013, it received LEED Platinum certification [36]. Centralized demand-controlled ventilation is utilized in the building. Indoor air quality is monitored while supply and exhaust air delivery are adjusted based on indoor contaminants levels by the system. Such system automatically samples packets of air and analyze with a battery of sensors to determine required air change rates for each thermal zone. To ensure accurate measurements, the sensors are calibrated every six months. The system is monitored through a web base system. Occupancy serves as the guideline of determining indoor ventilation

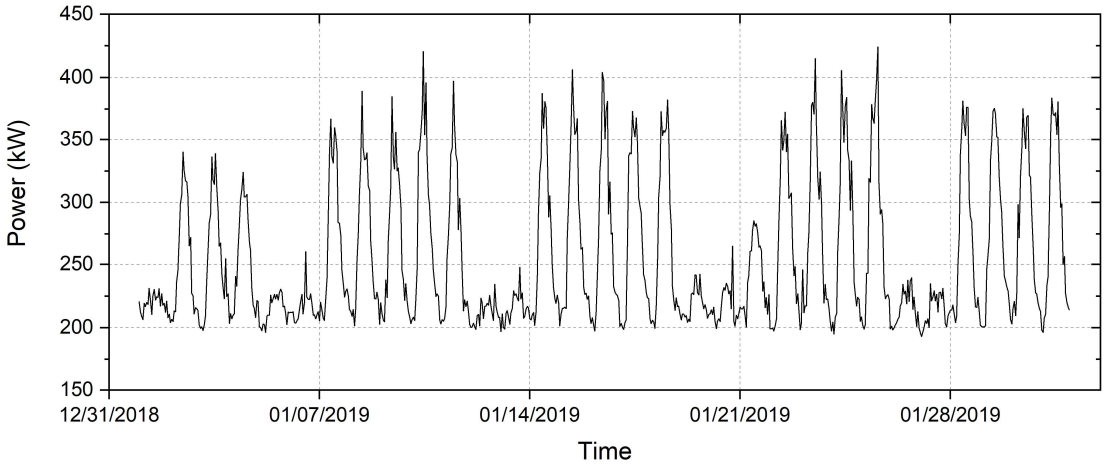

Figure 2. Hourly electric power in January 2019

by the system. Sensors detect occupancy and allow air change rate reductions when labs are unoccupied. Air change rates normally remain at roughly 4 air changes per hour (ACH) when the lab is occupied and 2 ACH when unoccupied. Heating and cooling are provided through hot and chilled water loop connected to the UCI central plant. Interior lighting of BioSci3 is designed to maximize interior illumination through natural lighting. Photosensors are also employed to automatically control lighting intensity based on outdoor light availability. In addition,

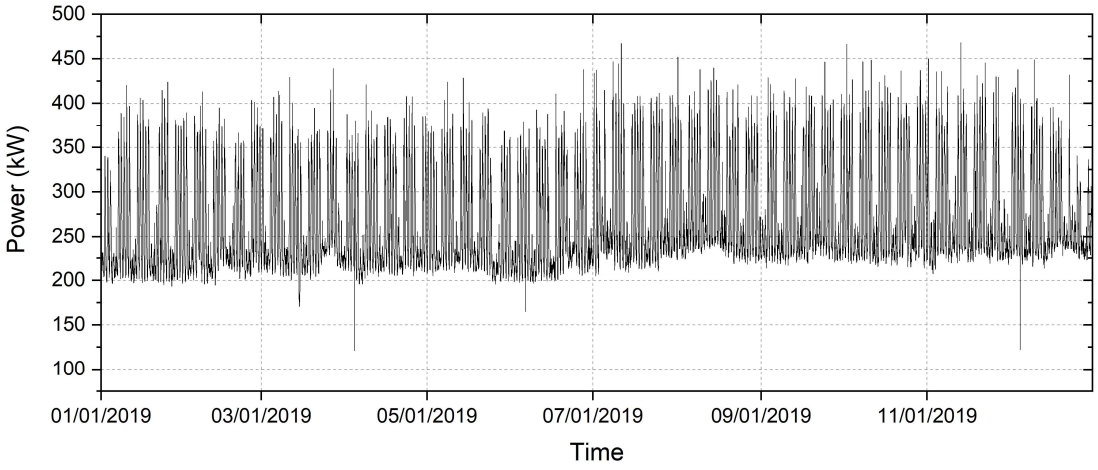

Figure 3. Hourly electric power in 2019

occupancy sensors automatically turn off overhead lighting when labs are vacant. The ventilation system is equipped with high efficiency particulate filters to improve indoor air quality and lower energy costs.

The hourly electric consumption of the building is metered on site and monitored through a web-based system, as well as the meteorological data. A full year of data in 2019 is available. *Figure 2* presents the hourly electric power of BioSci3 in January 2019. It can be seen from the figure that daily and weekly patterns exist. The electric load appears to rise in the morning, peak during the noon, and drop during afternoon until the morning of the next day. During the weekends, the electric consumption plummets as school activities subside while the loads are much higher in normal weekdays with school activities. A whole year of hourly electric power data of BioSci3 in 2019 is shown in *Figure 3*. As shown in the figure, the electric consumption seems relatively consistent throughout the year with little seasonal variation, although the consumption in the second half of the year appears slightly higher on average. The reason is potentially the moderate weather of Irvine, California throughout the year, combined with the fact that heating and cooling are provided through the water loop connected to the UCI central plant. Consequently, the January and February data were selected as the test sets before implementing the models throughout the year.

CHAPTER 2. PERFORMANCE EVALUATION

Several evaluation measures of electric load prediction are utilized in this study to compare the forecasting performances between different methods. Root mean square error (RMSE), coefficient of variance (CV), mean absolute percentage error (MAPE) and R-squared (R^2) are employed in this study. These measures are calculated using the following equations:

$$\text{Root Mean Square Error (RMSE)} = \sqrt{\frac{\sum_{i=1}^{n}\left(y_{predict,i} - y_{data,i}\right)^2}{n}}, \quad (1)$$

$$\text{Coefficient of Variation (CV)(\%)} = \frac{\sqrt{\frac{\sum_{i=1}^{n}\left(y_{predict,i} - y_{data,i}\right)^2}{n}}}{\bar{y}_{data}} \times 100, \quad (2)$$

$$\text{Mean Absolute Percentage Error (MAPE)(\%)} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_{predict,i} - y_{data,i}}{y_{data,i}}\right| \times 100\,, \quad (3)$$

$$\text{R} - \text{Squared (R2)} = 1 - \frac{\sum_{i=1}^{n}\left(y_{predict,i} - y_{data,i}\right)^2}{\sum_{i=1}^{n}\left(y_{data,i} - \bar{y}_{data,i}\right)^2}, \quad (4)$$

where $y_{predict,i}$ is the predicted electric load at time point i, $y_{data,i}$ is the actual electric load at time point i, $\bar{y}_{data}$ is the average actual electric load within the available dataset, and n is the number of data points in the dataset.

These are the most commonly used evaluation measures of energy prediction models [14]. RMSE is dependent on the scale of the data. A smaller RMSE indicates better forecasting accuracy. CV is most commonly used for to prediction performance in the studies. It is one of the measures endorsed by the American Society of Heating Refrigerating and Air-conditioning Engineers (ASHRAE) guideline of whole-building prediction model performance [37]. CV is required to be below 30% for hourly power calibration. Unlike RMSE, the prediction error is normalized by the average electric load and therefore CV is not scale-dependent. MAPE is also

not scale-dependent like CV. In this case study where all data are positive and much greater than zero, MAPE provides a simple unitless measure that is convenient for comparison purposes. R^2 is known as the coefficient of determination and serves as a statistical measure of how close the data are to the predicted curves. It ranges between 0 and 1 and the closer it is to 1 indicates better prediction accuracy.

Akaike's Information Criterion (AIC) is widely used as a measure to compare goodness-of-fit for statistical models such as ARIMA and TBATS. AIC is calculated as [38]:

$$\mathrm{AIC} = \log\left(\frac{SSE}{T}\right) + 2(k+2), \tag{5}$$

$$\mathrm{SSE} = \sum_{t=1}^{T} e_t^2\,, \tag{6}$$

where T is the number of estimated observations, k is the number of predictors in the model. SSE stands for sum of squared errors with e_t representing the errors. The model with the smallest AIC usually achieves the best forecasting performance. AIC penalizes the goodness-of-fit with complexity of the model.

CHAPTER 3. STATISTICAL METHODS

3.1 POLYNOMIAL REGRESSION

Polynomial regression returns a polynomial of specified degrees that provides the best fit of the given data points. The method of least square estimation is used to determine the coefficients of the polynomial [39]. Suppose a straight line:

$$y = \theta_0 + \theta_1 x, \tag{7}$$

needs to be determined to approximate observations $y_1, \ldots, y_n$ of dependent variable y taken at independent values $x_1, \ldots, x_n$ of independent variable x. The least squares estimates $\hat{\theta}_0$ and $\hat{\theta}_1$ interpret the values of θ_0 and θ_1 that minimize the sum:

$$S(\theta_0, \theta_1) = \sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i)^2. \tag{8}$$

The sum represents the summation of squared deviations of the observed values y_i from the approximated values $\theta_0 - \theta_1 x_i$. In Euclidean squared distance form, S can be written as:

$$S(\theta_0, \theta_1) = \|\boldsymbol{y} - \theta_0 \mathbf{1} - \theta_1 \boldsymbol{x}\|^2, \tag{9}$$

where $\boldsymbol{x} = (x_1, \ldots, x_n)^T$, $\mathbf{1} = (1, \ldots, 1)^T$, and $\boldsymbol{y} = (y_1, \ldots, y_n)^T$. By letting the partial derivatives of S with respect to θ_0 and θ_1 equal to zero respectively, the vector $\widehat{\boldsymbol{\theta}} = (\hat{\theta}_0, \hat{\theta}_1)^T$ can be proven to satisfy the following "normal equations":

$$X^T X \widehat{\boldsymbol{\theta}} = X^T \boldsymbol{y}, \tag{10}$$

where $X = [\mathbf{1}, \boldsymbol{x}]$. The normal equations have at least one solution because $S(\boldsymbol{\theta}) \geq 0$ and $S(\boldsymbol{\theta}) \to \infty$ as $\boldsymbol{\theta} \to \infty$. Let $\widehat{\boldsymbol{\theta}}^{(1)}$ and $\widehat{\boldsymbol{\theta}}^{(2)}$ be the solutions of the normal equations, then they satisfy the equation given as:

$$\left(\widehat{\boldsymbol{\theta}}^{(1)} - \widehat{\boldsymbol{\theta}}^{(2)}\right)^T X^T X \left(\widehat{\boldsymbol{\theta}}^{(1)} - \widehat{\boldsymbol{\theta}}^{(2)}\right) = 0, \tag{11}$$

which is equivalent to $X\hat{\boldsymbol{\theta}}^{(1)} = X\hat{\boldsymbol{\theta}}^{(2)}$. This argument can be universally applied to least squares estimation for the general linear model. A set of data points is given as:

$$(x_{i1}, x_{i2}, \dots, x_{im}, y_i), \quad i = 1, \dots, n \text{ with } m \leq n.$$

The least squares estimate $\hat{\boldsymbol{\theta}} = (\hat{\theta}_1, \dots, \hat{\theta}_m)^T$ of $\boldsymbol{\theta} = (\theta_1, \dots, \theta_m)^T$ minimizes the sum:

$$S(\boldsymbol{\theta}) = \sum_{i=1}^{n} (y_i - \theta_1 x_{i1} - \cdots - \theta_m x_{im})^2 = \left\| \boldsymbol{y} - \theta_1 \boldsymbol{x}^{(1)} - \cdots - \theta_m \boldsymbol{x}^{(m)} \right\|^2, \tag{12}$$

where $\boldsymbol{x}^{(j)} = (x_{1j}, \dots, x_{nj})^T$, $\boldsymbol{y} = (y_1, \dots, y_n)^T$, $j = 1, \dots, m$. Vector $\hat{\boldsymbol{\theta}}$ satisfies the following equation as proven previously:

$$X^T X\hat{\boldsymbol{\theta}} = X^T \boldsymbol{y}, \tag{13}$$

where X is a $n \times m$ matrix $X = [\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}]$. Therefore, $\hat{\boldsymbol{\theta}}$ can be solved by:

$$\hat{\boldsymbol{\theta}} = (X^T X)^{-1} X^T \boldsymbol{y}. \tag{14}$$

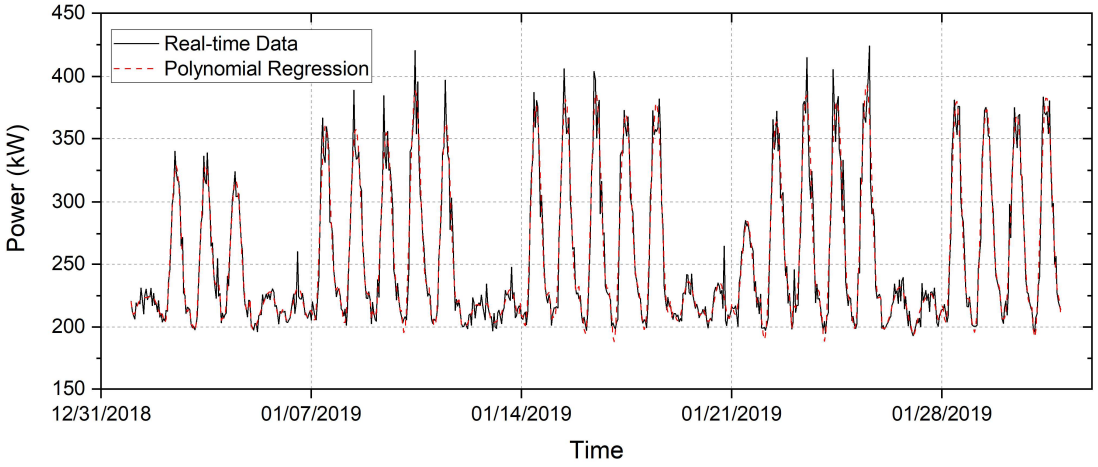

Figure 4. Comparison between real and polynomial regression fitted hourly electric power in January

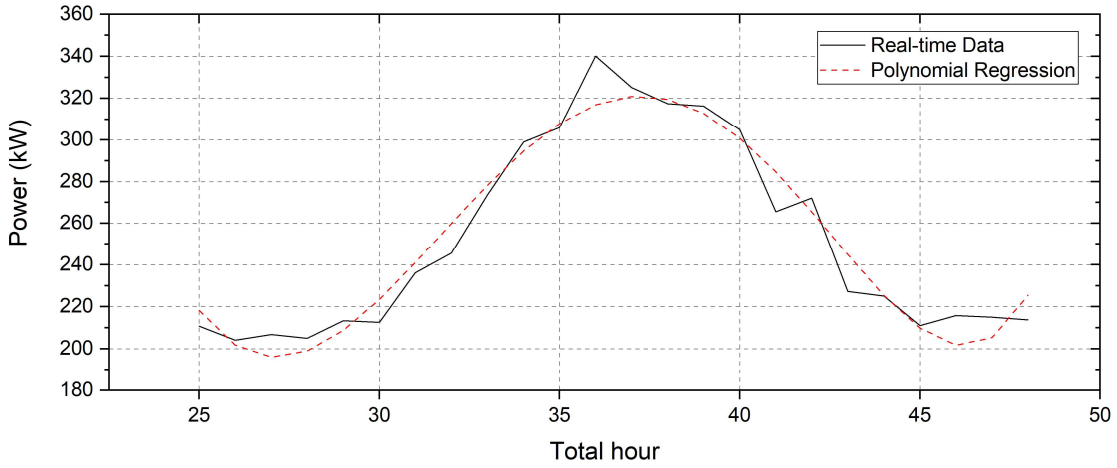

Figure 5. Comparison between real and polynomial regression fitted hourly electric power of one day in January

Table 1. Performance evaluation measures of daily polynomial regression fitting in January

RMSE	CV (%)	MAPE (%)	R-Squared
11.793	4.638	2.993	0.957

The polynomial coefficients vector $\widehat{\boldsymbol{\theta}}$ will be unique if and only if $X^T X$ is nonsingular.

Otherwise, there will be infinite solutions of $\widehat{\boldsymbol{\theta}}$, although the vector of approximated values $X\widehat{\boldsymbol{\theta}}$

will be the same for all solutions. The "polyfit" function of MATLAB was used to apply

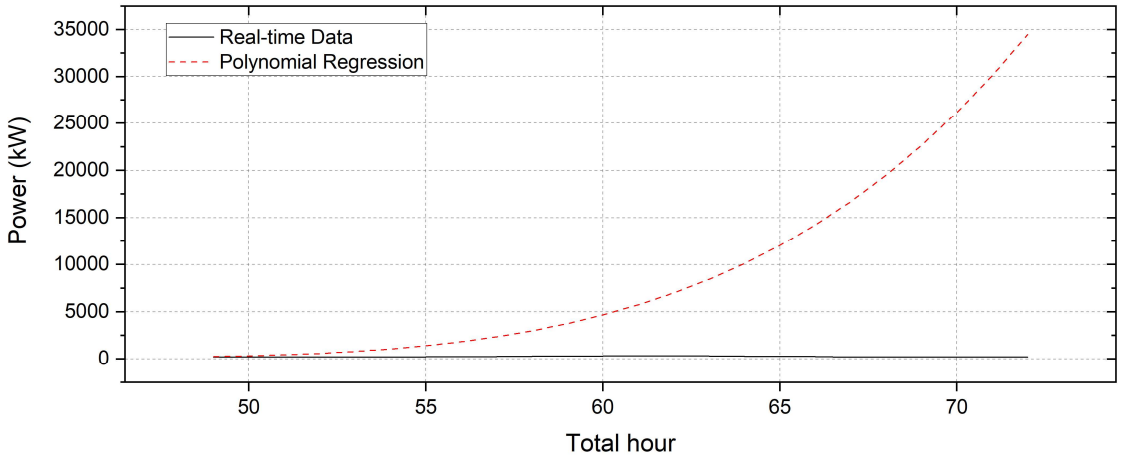

Figure 6. Comparison between real and polynomial regression forecasted hourly electric power of one day in January

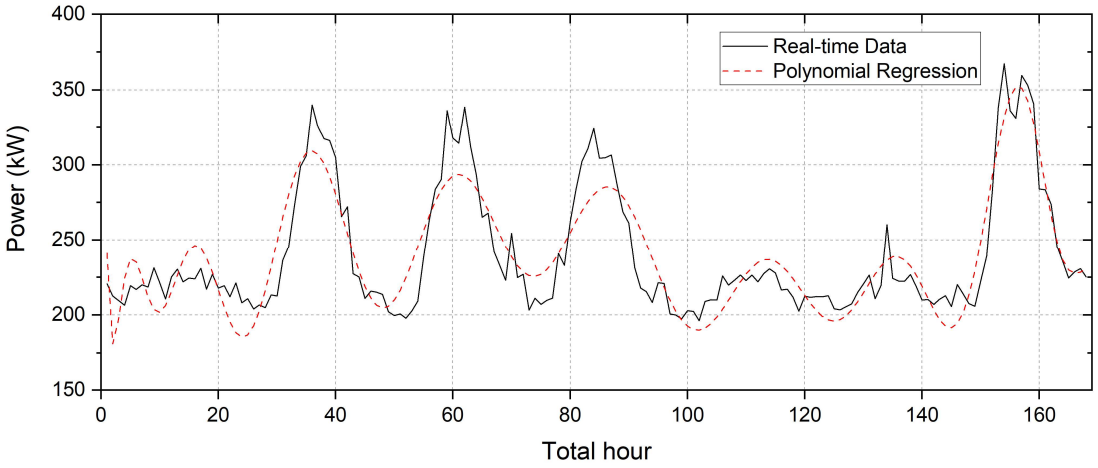

Figure 7. Comparison between real and polynomial regression fitted hourly electric power of one week in January

polynomial regression to the hourly electric power data in January 2019. Initially, fitting a full month of data, i.e. 744 data points in total, was attempted. Polynomial regression was unable to handle the dynamics of the data in such size. Each day, i.e. 24 data points, was fitted separately and 31 days of fitted values were then stitched together to compare with actual hourly power as a result. The order of the polynomial was determined to be 6 to ensure fitting accuracy. Figure 4 and Table 1 show that polynomial regression fitting achieves relatively satisfying results, as the fitted and real power curves match properly, and the metrics indicate high accuracy. However, the problem with polynomial regression is that the values given by the polynomial will inflate rapidly as the independent variable for the polynomial, which is the total count of hour, continuously increases. Figure 5 presents one of the fitted polynomials in January. This particular polynomial was then used for the attempt to forecast the power of the next day as shown in Figure 6. The total hour count of the next day, as the independent variable, was plugged into the polynomial to calculate the power of the next day. It can be seen that even though the total hour count increases by just 24, the polynomial blows up out of portion.

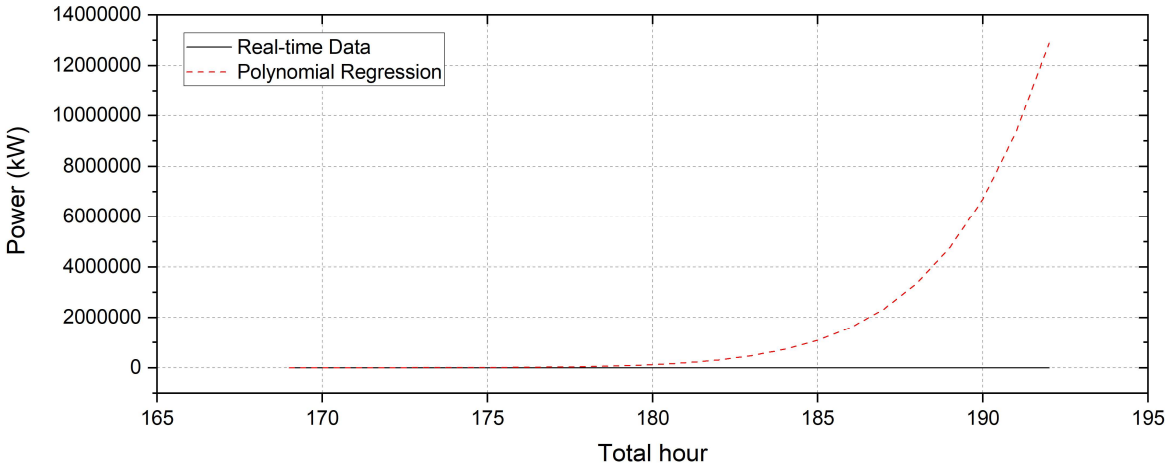

Figure 8. Comparison between real and polynomial regression forecasted hourly electric power of one week in January

Table 2. Performance evaluation measures of weekly polynomial regression fitting in January

RMSE	CV (%)	MAPE (%)	R-Squared
17.894	7.460	6.113	0.815

Polynomial regression was also examined with weekly period. A week in January was fitted as presented in Figure 7. The order of the polynomial was set to 20 in order to fit the real power curve properly. The fitting appears to be substantially less satisfactory, which is expected because the dynamics of weekly data are considerably more complicated than those of daily data. The performance metrics also indicate a worse accuracy in fitting as shown in Table 2. The total hour count of the next day was plugged in as the input, the same as the case of daily power polynomial regression before. As indicated by Figure 8, the polynomial blows up again. Therefore, polynomial regression with total hour count as the independent variable is proven not suitable for the case study.

3.2 ARIMA

ARIMA models provide an approach to time series forecasting. ARIMA models, along with exponential smoothing, are among the most widely used approaches to time series forecasting.

The autocorrelations that measure the linear relationship between lagged values in the time series data can be interpreted through an ARIMA model [38].

Predicting the variable of interest in an autoregressive model is achieved by using a linear combination of past values of the variable. The term autoregression illustrates that the variable does regression against itself. An autoregressive model of order p can be written as

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t, \tag{15}$$

where y_t is the variable of interest for forecasting, ε_t is the white noise $WN(0, \sigma^2)$, and ϕ_i is the ith autoregressive coefficient. This is referred as an AR(p) model.

Unlike the autoregressive model, a moving average model utilizes forecast errors in the past to make predictions in a regression-like model. A moving average model of order q is given as:

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} \ldots + \theta_q \varepsilon_{t-q}, \tag{16}$$

where θ_j represents the jth moving average coefficient. y_t is regarded as a weighted moving average of several forecast errors in the past for this model. This is denoted as a MA(q)model.

A non-seasonal ARIMA model can be obtained by combining differencing with an autoregressive model and a moving average model. Differencing is an approach to make non-stationary time series stationary by computing the differences between consecutive data points in the series. A time series is stationary if its statistical properties are not dependent on the time at which the series is observed. Stationarity is essential for the ARIMA method to achieve satisfactory forecasting performances. The first order differencing, often referred as the random walk model, is given by the following equation:

$$y'_t = y_t - y_{t-1}, \tag{17}$$

where y'_t is the calculated variable in the differenced series and y_{t-1} is the observation one time interval ahead. The differenced series describes change between consecutive observations in the

original series. The process of differencing is usually described using backshift operator B for convenience. The first order differencing mentioned above is denoted as:

$$y'_t = (1 - B)y_t, \tag{18}$$

$$By_t = y_{t-1}. \tag{19}$$

To generalize, dth order differencing is written as:

$$y'_t = (1 - B)^d y_t. \tag{20}$$

Therefore, a non-seasonal ARIMA model is denoted as ARIMA(p, d, q) with the following expression:

$$\left(1 - \phi_1 B - \cdots - \phi_p B^P\right)(1 - B)^d y_t = c + \left(1 + \theta_1 B + \cdots + \theta_q B^q\right)\varepsilon_t. \tag{21}$$

A time series data is considered to have seasonality when there is a seasonal pattern occurs. The pattern is caused by seasonal factors such as the time of the year of the day of the week. The frequency of seasonality is always fixe and known. A seasonal ARIMA model consists of additional seasonal terms to account for seasonal effects, denoted as SARIMA$(p, d, q)(P, D, Q)_m$. P, D and Q terms represent the seasonal orders of autoregression, differencing, and moving average. m is the number of the seasonal period. A SARIMA$(p, d, q)(P, D, Q)_m$ model can be expressed as:

$$\left(1 - \phi_1 B - \cdots - \phi_p B^P\right)(1 - \Phi_1 B - \cdots - \Phi_P B^P)(1 - B)^d (1 - B)^D y_t$$
$$= c + \left(1 + \theta_1 B + \cdots + \theta_q B^q\right)\left(1 + \Theta_1 B + \cdots + \Theta_Q B^Q\right)\varepsilon_t. \tag{22}$$

The SARIMA model used in this study was found by using a Python statistical library package called "pmarima." It is the equivalent to the "auto.arima()" function in R language that employs a variation of the Hyndman-Khandakar algorithm to obtain an ARIMA model, developed by Hyndman et al [40]. The Hyndman-Khandakar algorithm has the following general workflow:

1. The number of differencing d is found using repeated KPSS (Kwiatkowski-Phillips-Schmidt-Shin) tests.
2. The value of p and q are determined by minimizing the AIC after differencing the original series with the order determined in the last step. The algorithm uses the stepwise search method to find optimal combination of p and d.
    a. Four initial models are fitted
        i. ARIMA$(0, d, 0)$
        ii. ARIMA$(2, d, 2)$
        iii. ARIMA$(1, d, 0)$
        iv. ARIMA$(0, d, 1)$
    b. The model with the smallest AIC is chosen as the "current model".
    c. The current model is be altered and investigated:
        i. change p and/or q from the current model by ± 1;
        ii. include/exclude c from the current model.

        A new current model with a smaller AIC is found in this way.
    d. Repeat Step 2(c) until no model with a smaller AIC can be found.

The SARIMA model determined by the Python function "pmdarima" is denoted as SARIMA$(3,0,0)(4,1,3)_{24}$. The seasonal period was set to be 24 to account for the daily seasonality. In Table 3, the coefficients of the model are presented: three AR coefficients, four seasonal AR coefficients, and three seasonal MA coefficients. A seasonal differencing is also necessary. The P-values of the coefficients are all smaller than 0.05, indicating that all the AR

Table 3. Coefficients of the SARIMA model

	Coefficients	Standard error	z	P>|z|
Intercept	0.1099	0.116	0.947	0.344
AR.L1	0.6021	0.031	19.551	0
AR.L2	0.182	0.035	5.148	0
AR.L3	0.0792	0.034	2.328	0.02
AR.S.L24	1.1082	0.077	14.373	0
AR.S.L48	-1.1763	0.091	-12.934	0
AR.S.L72	0.2924	0.058	5.021	0
AR.S.L96	-0.2654	0.04	-6.645	0
MA.S.L24	-1.858	0.079	-23.637	0
MA.S.L48	1.6993	0.139	12.246	0
MA.S.L72	-0.7224	0.078	-9.214	0

and MA terms are statistically significant. The SARIMA model was then used to forecast the hourly electric power in February with January's data. Figure 9 exhibits the comparison between real and SARIMA predicted power in February. The large discrepancies between real and forecasted values reveal the fundamental problem with the SARIMA model, which is the lack of capability to account for multiple seasonalities. The model can effectively capture the daily pattern of electric load, i.e. rise in the morning, peak at noon, and drop in the afternoon, as the SARIMA forecasted power curve demonstrates. During weekdays, the forecasted and real power curves conform with each other substantially. On the other hand, the large drops in load during weekends are not properly recognized by the model. The discrepancies between real and forecasted values are the largest during weekends because the model treats them as ordinary weekdays. Consequently, the forecasting accuracy of the SARIMA model is not satisfactory as Table 4 implies.

Table 4. Performance evaluation measures of SARIMA forecast

RMSE	CV (%)	MAPE (%)	R-Squared
34.491	13.381	9.839	0.587

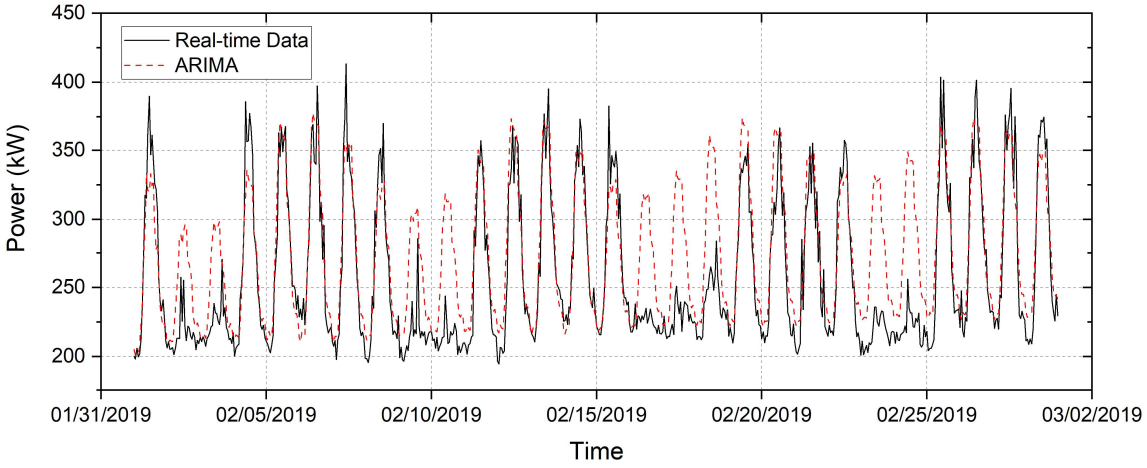

Figure 9. Comparison between real and SARIMA predicted hourly electric power in February

3.3 TBATS

TBATS was developed by De Livera et al. as an automated approach to tackle multiple and complex seasonality utilizing Fourier terms with an exponential smoothing state space model and a Box-Cox transformation [41]. TBATS is an acronym for key features of the model: Trigonometric seasonal formulation, Box-Cox transformation, ARMA errors, Trend, and Seasonal components. It was built upon the extension by Taylor to the Holt-Winters method as follows:

$$y_t = l_{t-1} + b_{t-1} + s_t^{(1)} + s_t^{(2)} + d_t, \tag{23}$$

$$l_t = l_{t-1} + b_{t-1} + \alpha d_t, \tag{24}$$

$$b_t = b_{t-1} + \beta d_t, \tag{25}$$

$$s_t^{(1)} = s_{t-m_1}^{(1)} + \gamma_1 d_t, \tag{26}$$

$$s_t^{(2)} = s_{t-m_2}^{(1)} + \gamma_2 d_t, \tag{27}$$

where m_1 and m_2 represent the periods of seasonality; d_t is the white-noise random variable representing the prediction error or disturbance; l_t and b_t are the level and trend component at time t, respectively; $s_t^{(i)}$ represents the ith seasonal component at time t; Coefficient α, β, γ_1 and γ_2 are the smoothing parameters. The average value in the time series is referred as the level while the increasing or decreasing value is referred as the trend. De Livera et al. extended this model to include Box-Cox transformation, ARMA errors and T seasonal periods as the following:

$$y_t^{(\omega)} = \begin{cases} \frac{y_t^{(\omega)}-1}{\omega}, & \omega \neq 0, \\ \log y_t, & \omega = 0, \end{cases} \qquad (28)$$

$$y_t^{(\omega)} = l_{t-1} + \phi b_{t-1} + \alpha d_t, \qquad (29)$$

$$l_t = l_{t-1} + \phi b_{t-1} + \alpha d_t, \qquad (30)$$

$$b_t = (1-\phi)b + \phi b_{t-1} + \beta d_t, \qquad (31)$$

$$s_t^{(i)} = s_{t-m_i}^{(i)} + \gamma_i d_t, \qquad (32)$$

$$d_t = \sum_{i=1}^{p} \varphi_i d_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t, \qquad (33)$$

where m_i is the seasonal period with $i = 1, \ldots, T$; l_t and b_t are the local level and trend in period t; b is the long-run trend in period t; d_t in this model represents an ARMA(p, q) process with φ_i and θ_i as the autoregressive and moving average coefficients; ϕ is the damping parameter for the Gardner and McKenzie damped trend; ε_t is the Gaussian white-noise process with zero mean and constant variance, designated as $WN(0, \sigma^2)$; α, β and γ_i are the coefficients of smoothing parameters. $y_t^{(\omega)}$ denotes the Box-Cox transformed observations at time t with the transformation parameter ω. With the extension, the model becomes the BATS model, which can accommodate multiple seasonality as generalization of the traditional seasonal innovations

models. However, the model cannot handle non-integer seasonality. In addition, it has a large amount of states that results in an enormous number of values for seasonality with large periods. To address these issues in a parsimonious manner, the trigonometric representation of seasonal components was introduced:

$$s_t^{(i)} = \sum_{j=1}^{k_i} s_{j,t}^{(i)}, \tag{34}$$

$$s_{j,t}^{(i)} = s_{j,t-1}^{(i)} \cos \lambda_j^{(i)} + s_{j,t-1}^{(i)} \sin \lambda_j^{(i)} + \gamma_1^{(i)} d_t, \tag{35}$$

$$s_{j,t}^{*(i)} = -s_{j,t-1} \sin \lambda_j^{(i)} + s_{j,t-1}^{*(i)} \cos \lambda_j^{(i)} + \gamma_2^{(i)} d_t, \tag{36}$$

$$\lambda_j^{(i)} = \frac{2\pi j}{m_i}, \tag{37}$$

where $\gamma_1^{(i)}$ and $\gamma_2^{(i)}$ are the smoothing parameters; k_i denotes the required seasonal harmonics for the ith seasonal component. The stochastic level of the ith seasonal component is designated as $s_{j,t}^{(i)}$ while the $s_{j,t}^{*(i)}$ describes the growth of $s_{j,t}^{(i)}$. With the addition of trigonometric formulation, a TBATS model is formed and designated as TBATS$(\omega, \phi, p, q, \{m_1, k_1\}, \{m_2, k_2\}, \ldots, \{m_T, k_T\})$.

The TBATS model utilized in this study was automatically determined by the 'tbats' Python function package, which is an equivalent to the same function in R language based on the study by De Livera et al [41]. The function follows the general workflow as below:

1. Estimate unknown parameters: $\omega, \phi, \gamma_1^{(i)}, \gamma_2^{(i)}$ based on maximum likelihood estimates
2. Select the number of seasonal harmonics k_j
    a. De-trend the data with an appropriate de-trending method
    b. Approximate the de-trended data with linear regression method
    c. Start with a single harmonic
    d. Gradually increase the number of harmonics while testing significance

e. Fit the required model to the data and computer AIC

f. Repeatedly fit the model to the estimation sample while gradually increasing one harmonic, keeping others constant, until lowest AIC is achieved

3. Select the ARMA orders by using the automatic ARIMA algorithm by Hyndman et al. introduced before [40].

Using "tbats" function in Python, TBATS$(0, NA, 2{,}4, \{24{,}8\}, \{168{,}6\})$ was decided as the best model for predicting loads in February with January's data. The Box-Cox transformation parameter ω is 0. No damping is required. The smoothing parameters are exhibited in Table 5. The ARMA parameters are 2 and 4, meaning there are two AR terms and four MA terms. Table 6 presents the ARMA coefficients. Seasonal periods are 24 and 168, representing daily and weekly patterns correspondingly in the number of data points. Forecasting hour electric load was attempted with this TBATS model. As Figure 10 implies, utilizing TBATS method addresses the problem of ARIMA method, which is the lack of capability to handle multiple seasonalities. Compared with the predicted power curve of SARIMA model, the TBATS predicted power curve captures the dips in electric load during weekends. The forecasting accuracy improves considerably, as the performance evaluation measures in Table 7 indicate. It is worth mentioning that the TBATS model performs worse during weekends in terms of forecast accuracy. The prediction is off by a wide margin during the weekends of the last two weeks in February. The

Table 5. Smoothing parameters of the TBATS model

	α	β	γ1	γ2	γ3	γ4
TBATS	0.0295	NA	-3.877E-05	-2.938E-05	-6.530E-05	1.154E-05

Table 6. ARMA coefficients of the TBATS model

	AR1	AR2	MA1	MA2	MA3	MA4
TBATS	-6.530E-05	1.154E-05	0.528	0.408	0.361	0.245

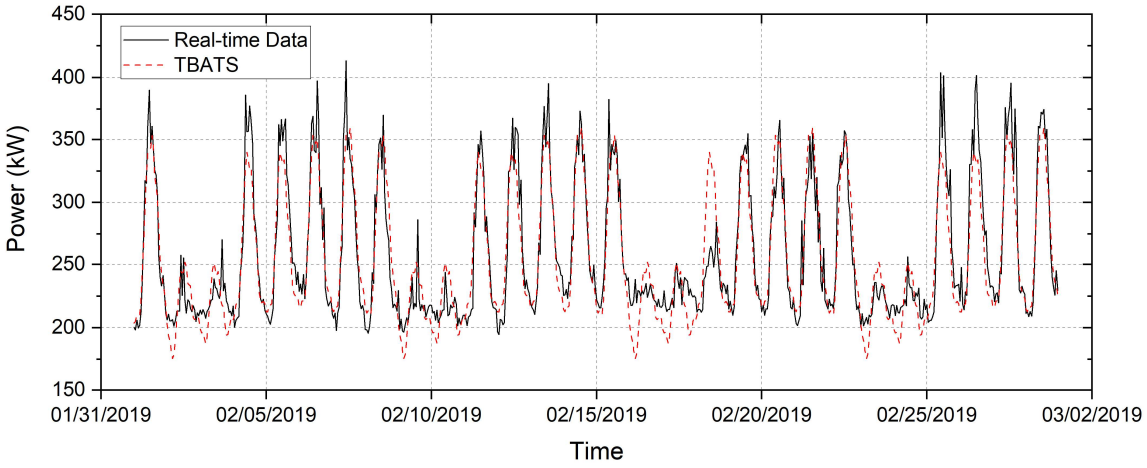

Figure 10. Comparison between real and TBATS predicted hourly electric power in February

Table 7. Performance evaluation measures of TBATS forecast

RMSE	CV (%)	MAPE (%)	R-Squared
20.608	7.996	5.951	0.853

electric power dynamics of weekends are different from those of weekdays. More spikes in electric load occur during weekends, which render forecasting more challenging.

TBATS was applied to the rest of the data throughout the year of 2019 in the same way, i.e. predicting a month of hourly electric loads with previous month's data. The results are shown in Table 8. The prediction accuracy of March, November and December is substantially worse than that of the other months. While the cause of reduction in accuracy remains unclear for March, holidays and outliers in the data lead to the unsatisfactory performance for November and December. Figure 11 shows the TBATS predicted load in December. It can be observed that after December 23rd, the prediction error becomes significantly larger than before. The Christmas holiday is unexpected for the TBATS method as it solely relies on past values for prediction, without any indicator to acknowledge the existence of holidays. In addition, there is an outlier in the first week. Prediction of April and November loads faces similar issues of outliers or

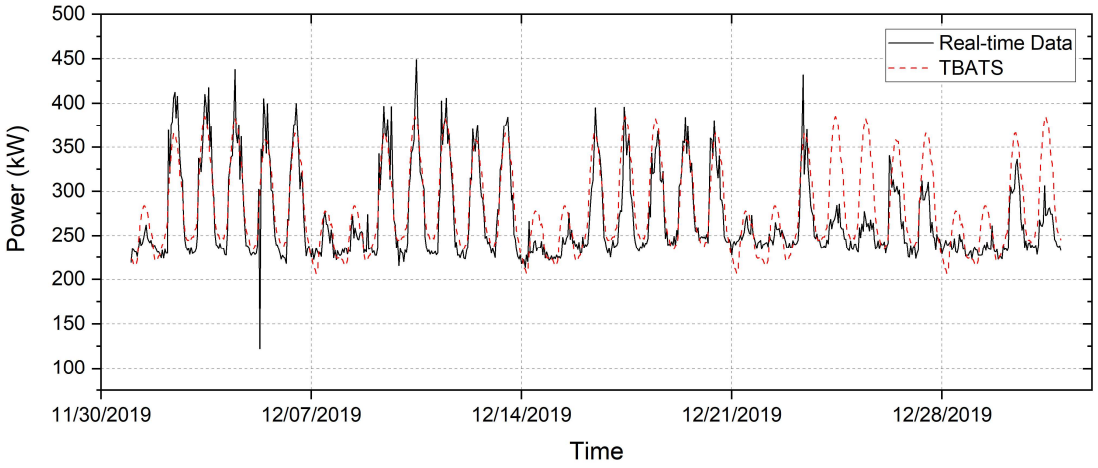

Figure 11. Comparison between real and TBATS predicted hourly electric power in December

Table 8. Performance evaluation measures of TBATS forecast of consecutive months

Trained/Predicted Month	RMSE	CV (%)	MAPE (%)	R-Squared
Jan/Feb	20.608	7.996	5.951	0.853
Feb/Mar	33.100	12.774	9.830	0.597
Mar/Apr	22.615	8.668	7.146	0.824
Apr/May	23.779	9.133	6.853	0.803
May/Jun	24.712	9.731	6.946	0.776
Jun/Jul	27.979	10.083	8.068	0.784
Jul/Aug	25.674	9.065	7.073	0.796
Aug/Sep	24.319	8.760	7.033	0.816
Sep/Oct	23.500	8.337	7.152	0.854
Oct/Nov	35.054	12.693	10.497	0.631
Nov/Dec	31.181	11.591	8.397	0.600
Mean	26.593	9.894	7.722	0.758

holidays where the loads are lower than those of the normal weekdays. The outliers and holidays were eliminated, and the performance statistics were recalculated with the results shown in *Table 9*, which show substantial improvement in accuracy. A different prediction strategy was also examined. Instead of fitting with one month of data and predicting the data of succeeding month, multiple months of data are used for prediction. July, August, September, and October were chosen as the example months for examination. Up to nine preceding months of data were used

Table 9. Performance evaluation measures of TBATS forecast of consecutive months (without outliers and holidays)

Trained/Predicted Month	RMSE	CV (%)	MAPE (%)	R-Squared
Jan/Feb	20.608	7.996	5.951	0.853
Feb/Mar	33.100	12.774	9.830	0.597
Mar/Apr	21.782	8.342	6.967	0.836
Apr/May	23.779	9.133	6.853	0.803
May/Jun	24.712	9.731	6.946	0.776
Jun/Jul	27.979	10.083	8.068	0.784
Jul/Aug	25.674	9.065	7.073	0.796
Aug/Sep	24.319	8.760	7.033	0.816
Sep/Oct	23.500	8.337	7.152	0.854
Oct/Nov	30.967	11.062	9.393	0.729
Nov/Dec	21.822	7.940	6.321	0.837
Mean	25.295	9.384	7.417	0.789

to predict the loads of these months as shown in Table 10. The results indicate that the prediction accuracy substantially increases when using more than one month of data for the model. On average, 5~20% of improvement in accuracy is achieved depending on different evaluation metrics and the amount of used data. Due to limited accessibility of load data in 2018, an

Table 10. Performance evaluation measures of TBATS forecast of multiple months

Trained/Predicted Month	RMSE	CV (%)	MAPE (%)	R-Squared
Feb~Jun/Jul	28.283	10.193	7.084	0.779
Apr~Jun/Jul	26.271	9.468	6.666	0.809
Jun/Jul	27.979	10.083	8.068	0.784
Jan~Jul/Aug	23.690	8.365	6.306	0.826
Mar~Jul/Aug	22.311	7.878	5.949	0.846
May~Jul/Aug	22.185	7.833	5.951	0.847
Jul/Aug	25.674	9.065	7.073	0.796
Feb~Aug/Sep	22.297	8.032	5.828	0.845
Apr~Aug/Sep	22.284	8.027	5.814	0.845
Jun~Aug/Sep	23.098	8.321	6.077	0.834
Aug/Sep	24.319	8.760	7.033	0.816
Jan~Sep/Oct	22.365	7.934	6.010	0.868
Mar~Sep/Oct	21.042	7.465	5.651	0.883
May~Sep/Oct	20.684	7.338	5.664	0.887
Jul~Sep/Oct	21.976	7.796	5.856	0.872
Sep/Oct	23.500	8.337	7.152	0.854

extensive examination, where every month of 2019 is predicted using multiple preceding months' data, cannot be achieved. Nonetheless, making prediction with multiple months of data is proven better in terms of accuracy.

CHAPTER 4. MACHINE LEARNING METHOD

4.1 BACKPROPAGATION ANN

A neural network can be considered as a network of "neurons" that are organized in layers [42]. Figure 12 depicts how an individual neuron works. A neuron takes inputs, x_1, x_2, …, and to compute the output, weights, w_1, w_2, …, are introduced. Weights are real numbers that denote the importance of the respective input to each output. The output of neuron is determined by whether the linear combination of the sum of the products of weights and inputs exceed an arbitrary threshold value in an inequality equation. The threshold value can be moved to the other side of the inequality and becomes the bias of the neuron, described as $b \equiv -\text{threshold}$. Therefore, the weighted input of a neuron is defined as:

$$z = w \cdot x + b. \tag{38}$$

The output is calculated by applying a so-called activation function $f(z)$ to the weighted input of a neuron. Generally, the activation function is utilized to ensure that changes in weights and biases will result changes in the output from the neuron according to the degree of the changes. Learning of the neural network is therefore achievable by repeatedly changing the weights and biases to produce better outputs and thus optimal weights and biases can be found.

The learning process starts with using a training data set. The training input is denoted as x and the desired corresponding output is described as $y = y(x)$. An algorithm, known as the

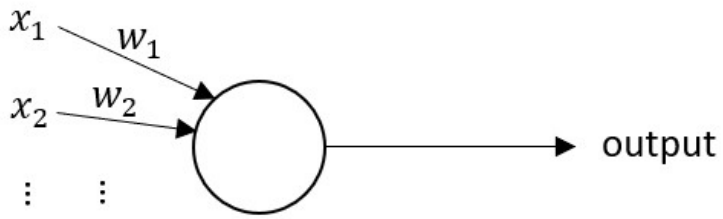

Figure 12. Structure of a neuron

optimizer, is required to let the network find weights and biases so the output of the network approaches $y(x)$ for all training inputs x. Before introducing the algorithm, the cost function is defined to quantify how well the outputs from the network agree with $y(x)$. A commonly used cost function, known as the quadratic cost function, is given below:

$$C(w,b) = \frac{1}{2n}\sum_{x}\|y(x) - a\|^2, \tag{39}$$

where w represents the collection of all weights in the network; b is all the biases; n is the number of total training inputs; a is the vector of outputs from the network, depending on x, w, and b; the sum is over all training inputs. The cost is always positive and when it becomes small, it indicates that $y(x)$ is relatively close to a. So, the goal of the algorithm is to find a particular set of weights and biases that minimizes the cost function.

Such algorithm is known as gradient descent. Suppose there is some function $C(v)$ of two real valued variables, $v = v_1, v_2$. These are general variables that do not necessarily need to be set within the neural network context. Minimizing $C(v)$ means to find where it reaches its global minimum. Changing v_1 and v_2 by a small amount of Δv_1 and Δv_2 leads to the change of the cost as follows:

$$\Delta C \approx \nabla C \cdot \Delta v, \tag{40}$$

where ∇C is the gradient vector consisted of $\partial C/\partial v_1$ and $\partial C/\partial v_2$ in this case, and Δv is the vector formed by Δv_1 and Δv_2. To reduce the cost, Δv_1 and Δv_2 need to be chosen in a certain way to make ΔC negative. This can be pictured as nudging a ball in v_1 and v_2 directions down a valley. Let Δv be chosen as:

$$\Delta v = -\eta \nabla C, \tag{41}$$

where η is a small, positive parameter that represents the learning rate. By defining Δv in such way, $\Delta C \approx \nabla C \cdot \Delta v = -\eta \nabla C \cdot \nabla C = -\eta \|\nabla C\|^2$. Since $\|\nabla C\|^2 \geq 0$, it is guaranteed that $\Delta C \leq 0$. In other words, ΔC always decreases as desired because the goal is to reduce the overall cost. The value of v can then be updated repeatedly through computing Δv:

$$v \to v' = v - \eta \nabla C. \tag{42}$$

The cost can be continuously decreased by this update rule and eventually reach a global minimum, which is the essence of the gradient descent algorithm. The gradient descent update rule in the context of neural network has the following form:

$$w_k \to w_k' = w_k - \eta \frac{\partial C}{\partial w_k}, \tag{43}$$

$$b_l \to b_l' = b_l - \eta \frac{\partial C}{\partial b_l}. \tag{44}$$

In this way, the network learns from training inputs and gradually decreases the overall cost.

In practice, the gradient descent algorithm is often replaced with stochastic gradient descent in order to speed up the learning process. The overall cost is calculated by averaging the cost over all individual training examples. Similarly, computing the gradient ∇C requires the network to compute the gradients ∇C_x and average them over all training examples. The computation becomes challenging and the learning rate decreases as the number of training inputs becomes large. To address this issue, the stochastic gradient descent algorithm randomly selects a small number m of training inputs chosen randomly. The randomly chosen inputs are referred as $X_1, X_2, \ldots, X_m$, which form a mini-batch. If the sample size m is large enough, the average value of ∇C_x can be approximated by calculating the average ∇C_{X_j} as the following:

$$\frac{\sum_{j=1}^{m} \nabla C_{X_j}}{m} \approx \frac{\sum_x \nabla C_x}{n} = \nabla C. \tag{45}$$

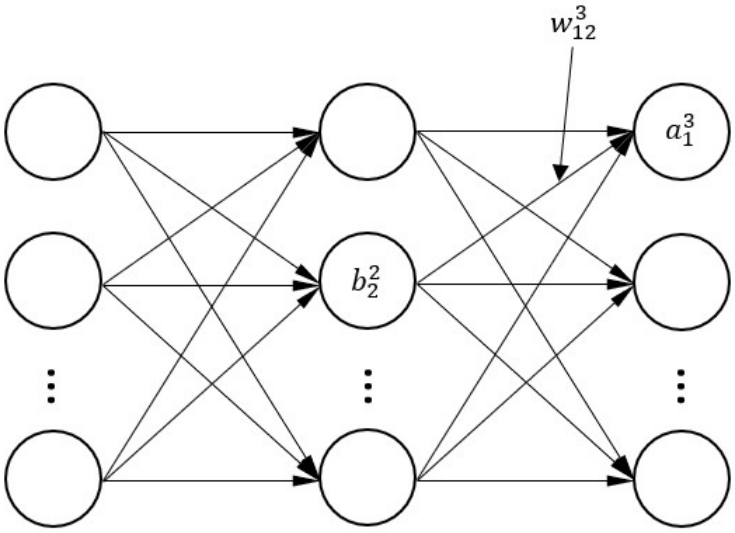

Figure 13. Structure of a backpropagation neural network

Therefore, the overall gradient can be approximated by computing the gradients of randomly selected mini-batch. The update rule of stochastic gradient descent algorithm is given as:

$$w_k \to w_k' = w_k - \frac{\eta}{m}\sum_j \frac{\partial C_{X_j}}{\partial w_k}, \tag{46}$$

$$b_l \to b_l' = b_l - \frac{\eta}{m}\frac{\partial C_{X_j}}{\partial b_l}, \tag{47}$$

where all training examples X_j in the current mini-batch are summed. Once the network finishes training with one mini-batch, another mini-batch will be randomly selected for a new round of training. The process will be repeated until all training examples have been trained with.

A simple backpropagation neural network has a structure presented in Figure 13. From the figure, there are three layers consisted of circles that denote the neurons. The leftmost in the network is called the input layer, which consists of input neurons. The output layer is the rightmost layer, containing the output neurons in this case. The intermediate layer is called the hidden layer and the neurons in the layer are neither input nor output neuron. In the diagram, w_{jk}^l refers to the weight from the k-th neuron in the $(l-1)$-th layer to the j-th neuron in the l-th

layer and b_j^l represents the bias of the j-th neuron in the l-th layer. The activation a_j^l of the j-th neuron in the l-th layer is calculated by:

$$a_j^l = f\left(\sum_k w_{jk}^l \, a_k^{l-1} + b_j^l\right), \tag{48}$$

$$a^l = f(w^l a^{l-1} + b^l), \text{in vectorized form.} \tag{49}$$

This expression reveals how activations between adjacent layers relate to each other: the activations in the previous layer is applied with the weight matrix, added with the bias vector, and finally applied with the activation function.

Two assumptions are required to apply backpropagation algorithm in a neural network. The first assumption is given by the following equation:

$$C = \frac{1}{n}\sum_x C_x\,, \tag{50}$$

where C_x is the cost function for individual training example x. This assumption denotes that the cost function can be written as an average over the cost function of individual training examples. The assumption is needed because what the backpropagation actually does is computing the partial derivatives $\partial C_x/\partial w$ and $\partial C_x/\partial b$ of individual training examples. $\partial C/\partial w$ and $\partial C/\partial b$ are then calculated by averaging over the partial derivatives of examples. The second assumption is that the cost can be expressed as a function of the outputs from the neural network as follows:

$$\text{cost } C = C(a^L). \tag{51}$$

Take the common quadratic cost function as an example. It can be written as:

$$C = \frac{1}{2}\|y - a_L\|^2 = \frac{1}{2}\sum_j \left(y_j - a_j^L\right)^2, \tag{52}$$

and hence it agrees with the assumption as a function of the output activations.

There are four fundamental equations behind the backpropagation algorithm. Before introducing these equations, it is necessary to first define an intermediate quantity δ_j^l, which serves as the error in jth neuron in the lth layer. The essence of backpropagation is to understand how the variations in weights and bias affect the cost function. This would lead to computing partial derivatives $\partial C/\partial w_{jk}^l$ and $\partial C/\partial b_j^l$. The error δ_j^l is used to relate $\partial C/\partial w_{jk}^l$ and $\partial C/\partial b_j^l$. Suppose there is a small perturbation Δz_j^l added to the weighted input of a neuron. The output of neuron changes from $\sigma(z_j^l)$ to $\sigma(z_j^l + \Delta z_j^l)$ and the change propagates through the network. The overall cost is eventually influenced and changed by an amount $\frac{\partial C}{\partial z_j^l}\Delta z_j^l$. If $\frac{\partial C}{\partial z_j^l}$ has a large value, then the perturbation Δz_j^l can be chosen to have the opposite sign of $\frac{\partial C}{\partial z_j^l}$ to reduce the overall cost. On the contrary, if $\frac{\partial C}{\partial z_j^l}$ is close to zero, then adjusting the perturbation Δz_j^l does not cause significant changes in cost. Therefore, the error is defined as

$$\delta_j^l = \frac{\partial C}{\partial z_j^l}. \tag{53}$$

The four fundamental equations together provide an approach of computing the error and the gradient of the cost function. The first equation is for the error in the output layer given as

$$\delta_j^L = \frac{\partial C}{\partial a_j^L}\sigma'\left(z_j^L\right), \tag{54}$$

$$\delta^L = \nabla_a C \odot \sigma'(z^L), \text{in matrix} - \text{based form.} \tag{55}$$

The $\frac{\partial C}{\partial a_j^L}$ term measures the rate of change of the cost as a function of the jth output activation while the $\sigma'\left(z_j^L\right)$ term measures the rate of change of the activation function at z_j^L. These terms can be easily obtained. The computational burden of $\frac{\partial C}{\partial a_j^L}$ depends on the particular form of the

cost function, but it should remain small as the function is already known. The weighted inputs z_j^L are already computed in the natural workflow of the neural network and only an extra step of differentiating is needed for $\sigma'(z_j^L)$.

The second equation is for the error as a function of the error in the next layer:

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l), \tag{56}$$

where $(w^{l+1})^T$ is the transposed matrix composed of weights in the $(l+1)$-th layer. The transpose weight matrix is applied to move the error backward through the network and provide a measure of the error at the output in layer l. This equation effectively propagates the error backward by the activation function in the l-th layer to obtain the error δ^l in the weighted input z^l.

The error δ^l of any layer in the network can be computed by combining the first and second fundamental equations. The error in the output layer δ^L is first calculated using the first fundamental equation; then, apply the second fundamental equation to obtain δ^{L-1}; the second equation is repeatedly applied and eventually errors of all layers are computed.

The third fundamental equation measures the rate of change of the cost with respect to any bias in the network as follows:

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l. \tag{57}$$

The rate of change $\frac{\partial C}{\partial b_j^l}$ is equal to the error δ_j^l, which is convenient since the previous two fundamental equations have already showed how to compute the errors.

The fourth fundamental equation describes the rate of change of the cost with respect to any weight in the network in such way:

$$\frac{\partial C}{\partial w_{jk}^{l}} = a_k^{l-1}\delta_j^l. \tag{58}$$

This equation illustrates how to compute the partial derivative $\partial C/\partial w_{jk}^{l}$ with terms a_k^{l-1} and δ_j^l. Both the activation term a_k^{l-1} and error term δ_j^l can be computed easily with methods shown before. The equation indicates that when the activation is small, the gradient term also tends to be small. This leads to slow learning rate of weights during the gradient descent process.

4.2 ANN CONFIGURATION SELECTION

It is necessary to investigate appropriate inputs variables for the ANN. The following inputs variables are considered in this case study:

- temperature
- humidity
- windspeed
- total hour
- hour
- weekday/weekend
- day of week
- previous week same hour power (PW)
- previous day same hour power (PD)
- previous 24-hour average power (P24)

Temperature, humidity and windspeed are metered hourly by the UCI central plant. These weather condition parameters can potentially have significant impact on building energy [15] [43]. Cooling and heating energy consumption might be related to outdoor temperature and

humidity. Strong wind enhances the effects of cold weather. In addition, air infiltration increases with high wind speed, which causes a temperature drop inside the building. Building wall surface heat loss is also dependent on windspeed as higher windspeed generates more surface heat loss. However, the effects on building energy consumption due to weather condition parameters are heavily building-dependent. Types of insulation materials, location of the building, and types of existing energy system can all lead to drastically different reactions to changing weather conditions in terms of building energy consumption.

Total hour is a count for the hourly electric power data depending on the size of dataset. A full year of hourly electric power data set consists of 8,760 data points and total hour ranges from 1 to 8,760 in this case. Hour is a recurring variable that marks the number of the hour every day for each hourly power data point. It ranges from 1 to 24 and repeats itself over the data set. Weekday/weekend is a Boolean value input for distinguishing data points of weekdays from those of weekends. Weekday data points are represented with 1 and 0 represents weekend data points. Day of week is a recurring input like hour, ranging from 1 to 7. Each number corresponds

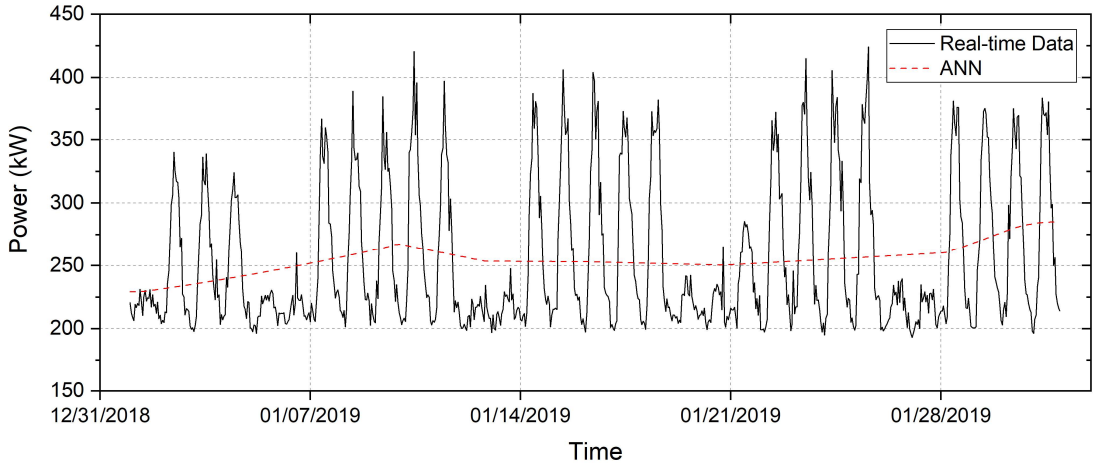

Figure 14. Comparison between real and ANN predicted hourly electric power in January with input: total hour

to a day in the week starting from Monday. Total hour is considered the fundamental input because it keeps tracks of time series data. Weekday/weekend is crucial for the ANN to capture the conspicuous dip in electric load during the weekends. Hour and day of week are examined to observe possible effects of patterns recognition. Electric power load of BioSci3 follows a similar trend each day; rises in the morning; peaks at noon; and drops in the afternoon. A similar trend may also be present for each day of the week. Hence, it is worth investigating these two inputs.

PW, PD and P24 are input variables referring to the actual electric hourly power data. Specifically, PW and PD are the actual hourly electric powers 168 hours and 24 hours ago respectively. P24 is the mean value of previous 24 data points of actual power. Electric powers at the same hour adjacent weeks may be similar or somehow related. The same idea can also apply for powers between two adjacent days. In addition, patterns of electric power with a period of 24 hours may exist. PW, PD and P24 can potentially aid in addressing the multiple seasonality present in the hourly power.

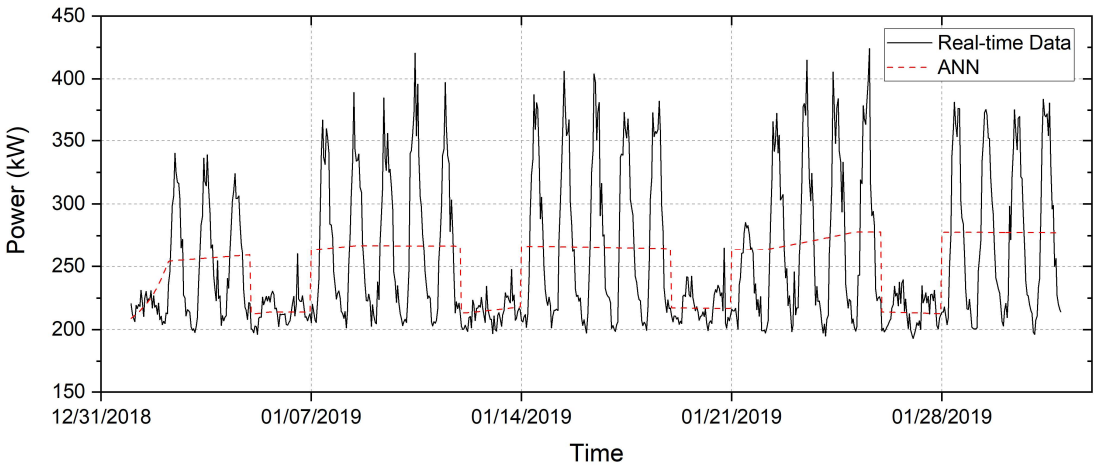

Figure 15. Comparison between real and ANN predicted hourly electric power in January with input: total hour, weekday/weekend

A temporary configuration of ANN was used to find the best configuration: ReLU/sigmoid activation function, Adam optimizer, and three hidden layers with 25 neurons in each layer. The ANN was built with Keras, an open-source neural network library written in Python.

In order to select appropriate inputs from the aforementioned candidates, the ANN was trained with hourly electric power data in January 2019 of BioSci3 building and prediction of the hourly electric power for a whole month was made using the inputs of the same month. The selection process is guided by the forecasting performances of the ANN forecasts that utilize different inputs. The performances are evaluated with the comparison of power curves between ANN forecasted and actual power, as well as the previously introduced evaluation measures including RMSE, CV, MAPE and R^2.

The total hour and weekday/weekend inputs are regarded as fundamental for the ANN. Their forecasting performances were initially observed. In Figure 14, the ANN does not appear to be working properly while the ANN seems to recognize the significant power drop during weekends in Figure 15 with the help of weekday/weekend input. However, only total hour and

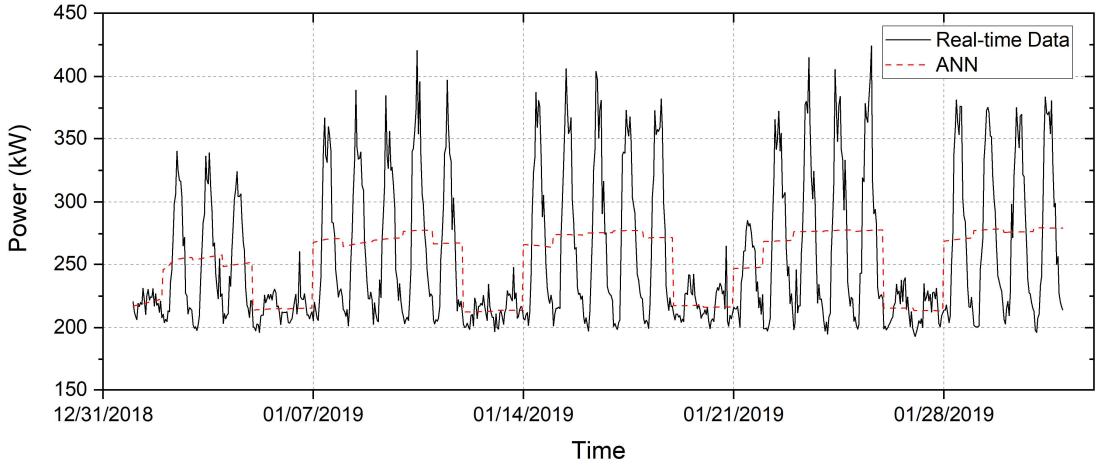

Figure 16. Comparison between real and ANN predicted hourly electric power in January with input: total hour, day of week, weekday/weekend

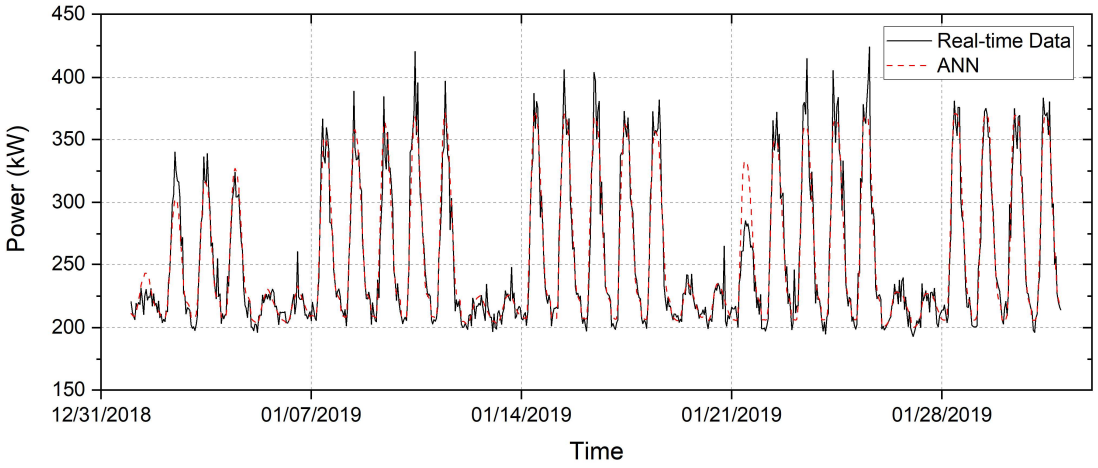

Figure 17. Comparison between real and ANN predicted hourly electric power in January with input: total hour, hour, weekday/weekend

weekday/weekend inputs are not sufficient for the ANN as two curves in Figure 15 barely match each other. The ANN with total hour and weekday/weekend inputs was chosen as the base model for further optimization.

The hour and day of week inputs were then introduced to the base model respectively for improvement in forecasting performances. The day of week input added to the base model does not provide conspicuous improvement in terms of reducing the error between forecasted and

Table 11. Performance evaluation measures of ANN model with input: total hour, day of week, hour, weekday/weekend

Run/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
1	14.177	5.576	3.718	0.938
2	15.045	5.917	3.740	0.930
3	13.092	5.149	3.283	0.947
4	12.406	4.879	3.173	0.952
5	12.353	4.858	3.245	0.953
6	15.167	5.965	3.795	0.929
7	17.762	6.986	4.130	0.902
8	16.590	6.525	4.376	0.915
9	14.479	5.695	3.467	0.935
10	14.919	5.868	3.636	0.931
Mean	14.599	5.742	3.656	0.933

Table 12. Performance evaluation measures of ANN model with input: total hour, hour, weekday/weekend

Run/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
1	11.702	4.602	3.101	0.958
2	13.992	5.503	3.516	0.939
3	11.872	4.669	3.123	0.956
4	11.613	4.567	3.055	0.958
5	12.186	4.793	3.118	0.954
6	11.556	4.545	3.038	0.959
7	12.766	5.021	3.276	0.950
8	12.362	4.862	3.370	0.953
9	12.937	5.088	3.297	0.948
10	13.820	5.435	3.486	0.941
Mean	12.480	4.909	3.238	0.952

actual power as shown in Figure 16. The forecasted power curve seems to become more dynamic than the base model but whether such characteristic is beneficial in terms of forecasting is uncertain since the overall error between two curves is too large in this case.

On the contrary, the hour input significantly improves the forecasting performances as indicated in Figure 17. This can be attributed to the fact that electric power load is similar during certain hour of each day or there may exist a particular pattern of electric power load during certain hour in a day. Hence, a new base model is determined with three inputs: total hour, hour, and weekday/weekend.

The effect of the day of week input remained unclear. Therefore, it was later added to the base model to further investigate.

Outputs of ANN vary within a certain range when the model is run each time. To avoid fluctuations in performances that potentially lead to incorrect results of input selection, each model is run ten times and the final evaluation is based on the average evaluation measures of ten runs. Table 11 and Table 12 present the performance evaluation measures of the base model

and adjusted base model with added day of week input. All performance evaluation measures of the adjusted base model are better compared to those of the base model. RMSE, CV and MAPE are smaller while R^2 is close to one as shown in the tables. Although day of week did not appear to be a valuable input in the first place, it was proven to be significant in the further investigation. As a result, the updated base model employs four inputs: total hour, hour, day of week, and weekday/weekend.

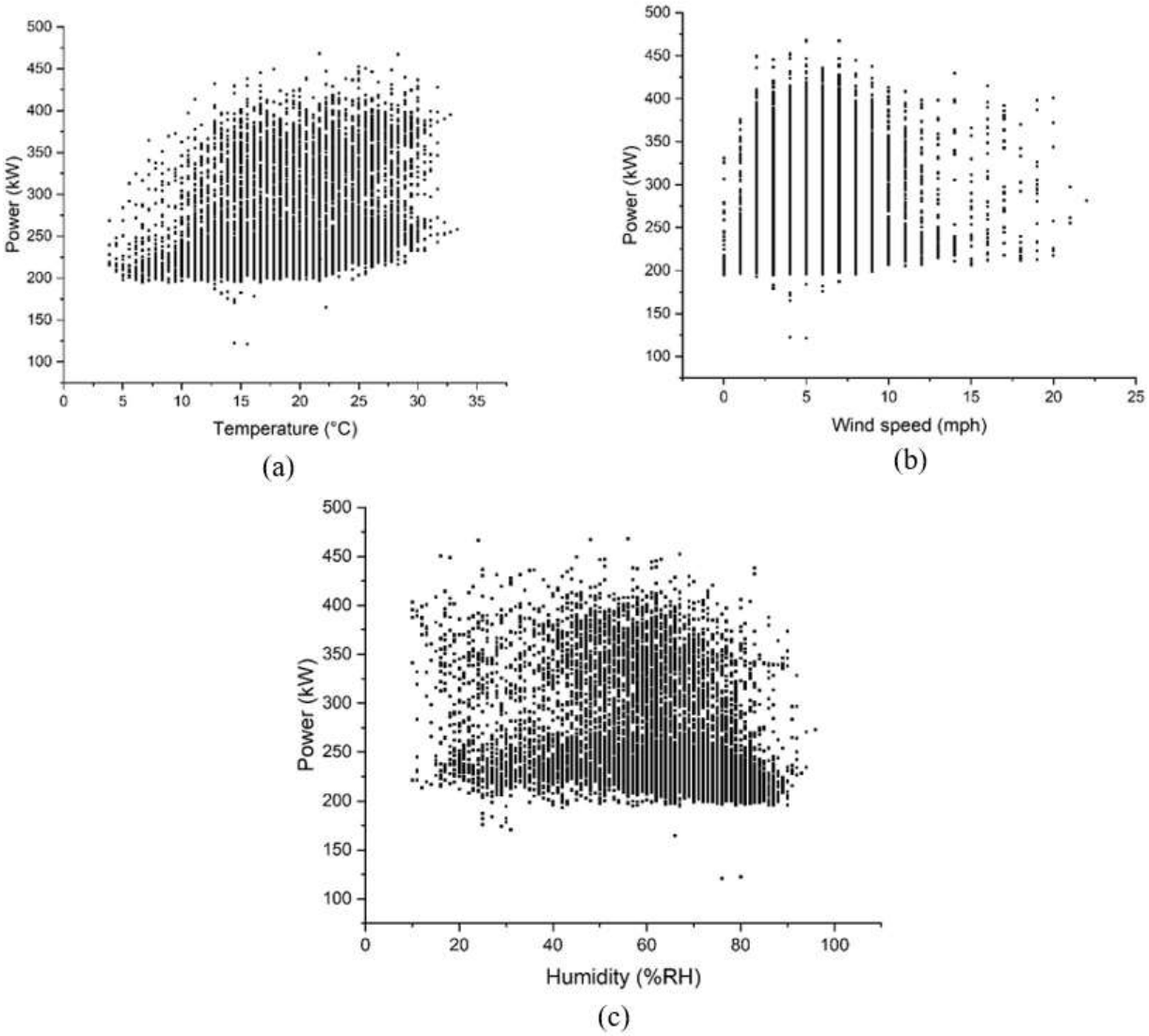

Figure 18. Hourly weather condition parameters versus corresponding electric power in 2019: (a) dry-bulb temperature versus power, (b) wind speed versus power, (c) relative humidity versus power

Before adopting weather condition parameters including hourly dry-bulb temperature, windspeed and relative humidity as ANN inputs, it is necessary to examine their relationship with electric power in the case of the BioSci3 building. As explained before, the dependence of energy consumption on weather condition parameters can differ drastically between buildings. Figure 18 presents the scatter plots of hourly temperature, wind speed and humidity with their corresponding electric power in 2019. It can be seen from Figure 18 (a) that most of the data points approximately fall within 10~30 ℃ temperature range. In the temperature range of 5~10 ℃, the data points are scarce except for the power range of 200~200 kW. The data points are scarce in the temperature range of 5~10 ℃ as the result of the moderate weather in Irvine, California. No conspicuous relationship between temperature and electric power can be observed in the 10~30 ℃ temperature range, as the data points are almost evenly distributed, forming a rectangle. For wind speed, most data point s fall within 0~12.5 mph range while data points become sparse in 12.5~22.5 mph range as presented in Figure 18 (b). Similar to the case of temperature, the relationship between wind speed and electric power does not appear significant. Data points of humidity are distributed more sparsely comparing with those of temperature and wind speed, as indicated in Figure 18 (c). The points are more concentrated across the span of 10~90% relative humidity, within the power range of 200~200 kW. In addition, more points are spotted within the power range of 250~400 kW and humidity range of roughly 45~80% relative

Table 13. Comparison of different ANN model input configurations: (a) base, (b) base with temperature input, (c) base with wind speed input, (d) base with humidity input

Configurations/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
a	12.480	4.909	3.238	0.952
b	12.629	4.967	3.165	0.951
c	12.035	4.733	3.070	0.955
d	12.372	4.866	3.144	0.953

humidity. The reason that leads to the concentration of data points in such way is unclear, as well as the relationship between humidity and electric power. Overall, employing weather condition parameters as inputs for the ANN model is precarious judging from the scatter plots. Nonetheless, weather condition parameters were utilized as inputs in the ANN to verify what was leant from the scatter plots. Temperature, wind speed and humidity were introduced to the base model as additional inputs. Table 13 presents the evaluation measures of different ANN configurations as each configuration was run ten times. Configuration (a), (b), (c) and (d) represent the base model and base model with temperature, wind speed and humidity as inputs, respectively. Compared to the base model, configuration (b) and (c) are relatively close in terms of performance statistics. The differences in RMSE, CV, MAPE and R^2 are roughly within 2%. This is possibly caused by the inherent fluctuations in performances of ANN since the differences are minor. Configurations of the network with temperature and humidity as inputs do not exhibit significant advantages in performances over the base configuration, as the scatter plots indicate that there is no clear relationship between these weather condition parameters and electric power. In contrast, configuration (c) is approximately 3~5% better than the base model in terms of forecasting performances according to the statistics. However, the scatter plot of wind speed in Figure 18 does not indicate that relationship between wind speed and electric power exists. It is unclear why configuration (c) outperforms the other two configurations with weather parameter inputs. Inconsistency with ANN outputs can potentially be one of the causes. Wind

Table 14. Comparison of different ANN model input configurations: (a) base with PW input, (b) base with PD input, (c) base with P24 input, (d) base PW, PD and P24 inputs

Configurations/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
A	12.218	4.805	3.094	0.954
B	12.189	4.794	3.126	0.954
C	11.987	4.715	3.066	0.955
D	11.922	4.689	3.012	0.956

speed is later further tested as an input when performing out-of-sample forecasting, i.e., predicting electric power given with inputs outside of the training data.

PW, PD and P24 serve as extra inputs for the base model in the ANN to validate their effectiveness. Table 14 present the statistics of ANN configurations utilizing PW, PD and P24 to compare their performances. Configuration (A), (B), (C) and (D) describe the base model with inputs: PW, PD, P24 and all three combined. As usual, each configuration was run ten times. All configurations indicate considerable enhancement in performances compared to the base configuration evaluating from the statistics. Configuration (A) and (B) exhibit similar level of forecasting accuracy, whereas configuration (C) and (D) perform almost equally well. It is desirable to select configuration (D) as the determined set of ANN inputs based on its exceptional forecasting accuracy demonstrated by the evaluation measures. However, it should be noted that more inputs for the ANN result in more computational power and longer computational time required. If, for example, the difference in out-of-sample forecasting accuracy of configuration (A) and configuration (D) is negligibly small, then a model with less input is preferred considering the parsimony of input. Hence all configurations were chosen to perform out-of-sample forecasting.

Previously, the ANN was trained with January 2019 data and it predicted the hourly power of the same month with the same inputs from the training samples. To do out-of-sample forecasting,

Table 15. Comparison of different ANN model input configurations: (1) base, (2) base with PW input, (3) base with PD input, (4) base with P24 input, (5) base with PW, PD and P24 inputs, (6) base with wind speed input

Configurations/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
1	22.556	8.751	5.582	0.823
2	20.748	8.050	5.463	0.850
3	23.002	8.924	5.705	0.816
4	21.950	8.516	5.569	0.832
5	22.078	8.566	5.830	0.830
6	24.138	9.365	5.926	0.798

the ANN was trained with January 2019 data and predicted February hourly power using inputs from February. Configurations that had been proven to be promising in terms of forecasting accuracy were examined. Table 15 shows the tested configurations with inputs: (1) base inputs, (2) base inputs with PW, (3) base inputs with PD, (4) base inputs with P24, (5) base inputs with PW, PD and P24, (6) base inputs with wind speed. The forecasting performances noticeably deteriorate for all configurations comparing with in-sample forecasting, which is expected to a certain degree as the network handles unseen inputs in this case. In terms of the statistics on average of out-of-sample forecasting, RMSE, CV and MAPE are approximately twice as much; R^2 decreases about 15%, i.e., 15% less accurate. It can be seen from Table 15 that configuration (2) demonstrate best forecasting performances by comparing each performance evaluation measure. Notice configuration (6) performs worst among all configurations, even worse than the base model. This contradicts with the results from in-sample forecasting, where the adjusted base model with wind speed input, i.e., configuration (c), exhibits better prediction accuracy than the base model. As mentioned earlier, the scatter plot of wind speed does not indicate a relationship between wind speed and electric power in Figure 18. Therefore, the forecasting performance of configuration (6) agrees with the verdict from the scatter plot, although the exact cause of the contradiction remains uncertain. It is also worth mentioning that during in-sample prediction, the adjusted base model with PW, PD and P24 inputs combined, i.e. configuration (D), shows best forecasting accuracy. On the contrary, the adjusted base model with PW input, which is configuration (2), makes the most accurate prediction of out-of-sample forecasting. The reason can potentially be the difference between the daily electric power patterns in January and February. What is learnt from inputs and outputs in January does not translate smoothly into February, to the point where it even impairs forecasting accuracy, although this is merely a

speculation. The black-box nature of an ANN prediction model renders detailed understanding of how different inputs affect forecasting accuracy challenging. Configuration (2) was chosen as the input configuration for the backpropagation neural network and used for further testing.

Activation function, optimizer, and number of hidden layers and neurons were left undetermined for the ANN after inputs were defined. Different activation functions can be used between the different layers of ANN. Some preliminary tests were done to winnow activation functions and the candidates of activation functions were given as the following:

1. Sigmoid function: $sigmoid(x) = \frac{1}{1+e^{-x}}$
2. Rectified Linear Unit (ReLU) function: $ReLU(x) = \max(0, x)$
3. Hyperbolic tangent function: $\tanh(x)$
4. Scaled Exponential Linear Unit function (SELU):
    a. If $x > 0$: return $scale * x$
    b. If $x < 0$: return $scale * alpha * (e^x - 1)$
    c. $scale = 1.05070098$, $alpha = 1.67326324$.

ReLU was found to be most promising in terms of forecasting accuracy as the activation function between the input layer and hidden layers. The aforementioned candidates were further tested through February hourly power forecasting as the activation function between the hidden layer and output layer. Each activation configuration was tested ten times to reduce the fluctuation in performance.

Table 16. Comparison of different ANN activation function configurations

Configuration/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
ReLU/sigmoid	20.748	8.050	5.463	0.850
ReLU/ReLU	22.563	8.754	5.835	0.820
ReLU/tanh	21.762	8.443	5.556	0.833
ReLU/SELU	21.971	8.524	5.674	0.831

The results in Table 16 reveal that ReLU/sigmoid configuration has the best prediction performance and hence it is selected as the definitive activation function combination for the ANN. Among the spectrum of available optimizers, Adam was chosen due to its capability of fast convergence [44]. The numbers of hidden layers and neurons in each layer were examined extensively as indicated in Table 17. It can be observed that in the structure of three hidden layers, changing the number of neurons does not drastically affect the performance of forecasting. Instead, the number of neurons decreases from 25 to 10 while the performance remains roughly on the same level, which might be caused by the fluctuations rather than actual differences in performances. Similar argument can be made for changing the number of hidden layers while keeping the number of neurons constant. The number of hidden layers changes from three to one but the forecasting accuracy does not change accordingly. On the other hand, prediction accuracy drastically decreases as the number of neurons drops below a certain threshold. The performance of one hidden layer with 5 neurons configuration is significantly worse than the configuration with one hidden layer consisting 10 neurons. ANN with less hidden layers and neurons are preferred because of less computational burden.

To further investigate the optimal configurations of the ANN, the electric power loads of the rest of the months throughout the year was predicted, with results shown in Table 18. The statistics are the average values of 10 runs, which were conducted to avoid fluctuation in the

Table 17. Comparison of different ANN hidden layers and neurons configurations

Configuration/Statistics	RMSE	CV (%)	MAPE (%)	R-Squared
3 layers, 25 neurons	20.748	8.050	5.463	0.850
3 layers, 20 neurons	20.881	8.101	5.461	0.848
3 layers, 15 neurons	20.586	7.987	5.413	0.852
3 layers, 10 neurons	21.144	8.203	5.621	0.844
2 layers, 10 neurons	21.018	8.155	5.508	0.846
1 layer, 10 neurons	20.582	7.986	5.672	0.853
1 layer, 5 neurons	25.357	9.838	7.007	0.775

Table 18. Performance evaluation measures of ANN forecast of consecutive months

Trained/Predicted Month	RMSE	CV (%)	MAPE (%)	R-Squared
Jan/Feb	21.144	8.203	5.621	0.844
Feb/Mar	23.248	8.972	6.181	0.801
Mar/Apr	21.797	8.355	6.161	0.836
Apr/May	23.198	8.910	6.358	0.812
May/Jun	22.869	9.005	6.976	0.808
Jun/Jul	30.505	10.993	7.857	0.742
Jul/Aug	24.552	8.669	6.198	0.812
Aug/Sep	23.033	8.297	5.688	0.835
Sep/Oct	19.481	6.911	4.580	0.899
Oct/Nov	29.056	10.521	6.564	0.746
Nov/Dec	30.678	11.404	7.471	0.613
Mean	24.506	9.113	6.332	0.795

performance. After conducting preliminary tests, the configuration of the ANN is chosen as: three layers, 10 neurons, ReLU/sigmoid activation functions, and Adam optimizer. Such configuration ensures more stable results for prediction with satisfactory accuracy and relatively less computational burden. Configurations with less layers and numbers demonstrate better performance statistics, as presented in Table 17, but enormous drops in prediction accuracy occur occasionally. After removing outliers and holidays, which is done similarly for the TBATS method above, the results were recalculated as shown in Table 19. Prediction accuracy

Table 19. Performance evaluation measures of ANN forecast of consecutive months (without outliers and holidays)

Trained/Predicted Month	RMSE	CV (%)	MAPE (%)	R-Squared
Jan/Feb	21.144	8.203	5.621	0.844
Feb/Mar	23.248	8.972	6.181	0.801
Mar/Apr	21.106	8.084	6.000	0.845
Apr/May	23.198	8.910	6.358	0.812
May/Jun	22.869	9.005	6.976	0.808
Jun/Jul	30.505	10.993	7.857	0.742
Jul/Aug	24.552	8.669	6.198	0.812
Aug/Sep	23.033	8.297	5.688	0.835
Sep/Oct	19.481	6.911	4.580	0.899
Oct/Nov	23.636	8.443	5.508	0.842
Nov/Dec	23.953	8.715	6.148	0.803
Mean	23.339	8.655	6.101	0.822

significantly increases for April, November, and December. Comparing with results from the TBATS method using the same strategy, that is, trained with one month and predict the following month, the ANN model achieves better accuracy in most of the months throughout the year. ANN is approximately 5~20% more accurate than TBATS evaluating from RMSE, CV, MAPE, and R^2 in terms of the average of all the months throughout the year.

In addition, training with multiple months of data was conducted in a similar way performed above for the TBATS method. October was selected as the sample month for examination. Up to nine preceding months of data was used while the number of neurons ranged from 10 to 25. In contrast to the TBATS method, prediction accuracy decreases as more historical data is introduced to the model, as Table 20 presents. Evaluating with the performance metrics, the

Table 20. Performance evaluation measures of ANN forecast of multiple months

Trained/Predicted Month	Configuration (layer-neuron)	RMSE	CV (%)	MAPE (%)	R-Squared
Jan~Sep/Oct	3-10	29.845	10.588	8.179	0.764
Jan~Sep/Oct	3-15	28.900	10.252	7.607	0.779
Jan~Sep/Oct	3-20	28.611	10.150	7.436	0.783
Jan~Sep/Oct	3-25	28.858	10.237	7.513	0.779
Mar~Sep/Oct	3-10	27.433	9.732	7.127	0.801
Mar~Sep/Oct	3-15	25.988	9.219	6.478	0.821
Mar~Sep/Oct	3-20	26.590	9.433	6.652	0.813
Mar~Sep/Oct	3-25	27.722	9.834	6.851	0.797
May~Sep/Oct	3-10	24.364	8.643	6.085	0.842
May~Sep/Oct	3-15	24.037	8.527	5.927	0.847
May~Sep/Oct	3-20	25.331	8.986	6.167	0.830
May~Sep/Oct	3-25	25.345	8.991	6.218	0.830
Jul~Sep/Oct	3-10	20.100	7.130	4.800	0.893
Jul~Sep/Oct	3-15	20.100	7.130	4.800	0.893
Jul~Sep/Oct	3-20	21.560	7.648	5.133	0.877
Jul~Sep/Oct	3-25	21.841	7.748	5.258	0.874
Sep/Oct	3-10	19.868	7.048	4.660	0.895
Sep/Oct	3-15	20.007	7.097	4.738	0.894
Sep/Oct	3-20	20.341	7.216	4.715	0.890
Sep/Oct	3-25	20.692	7.341	4.915	0.887

accuracy difference between the best and worst performing configuration ranges from roughly 15% to 40%. Training with just one preceding month of data leads to best prediction performance, using the configuration of three layer and ten neurons.

CHAPTER 5. METHODS COMPARISON

To investigate whether the model has properly captured the information in the data, the residuals of all three forecasting methods are checked. Residuals are the difference between the observations and the corresponding predicted values. The residuals yielded by a good forecasting method will have the following properties [38]:

1. The residuals are uncorrelated. If there are correlations between residuals, some information in the residuals is left unused for the forecasting model.
2. The residuals have zero mean. If the mean is nonzero, the forecast is biased.

It is also beneficial for the residuals to have additional properties given as follows:

1. The residuals have constant variance
2. The residuals have normal distribution

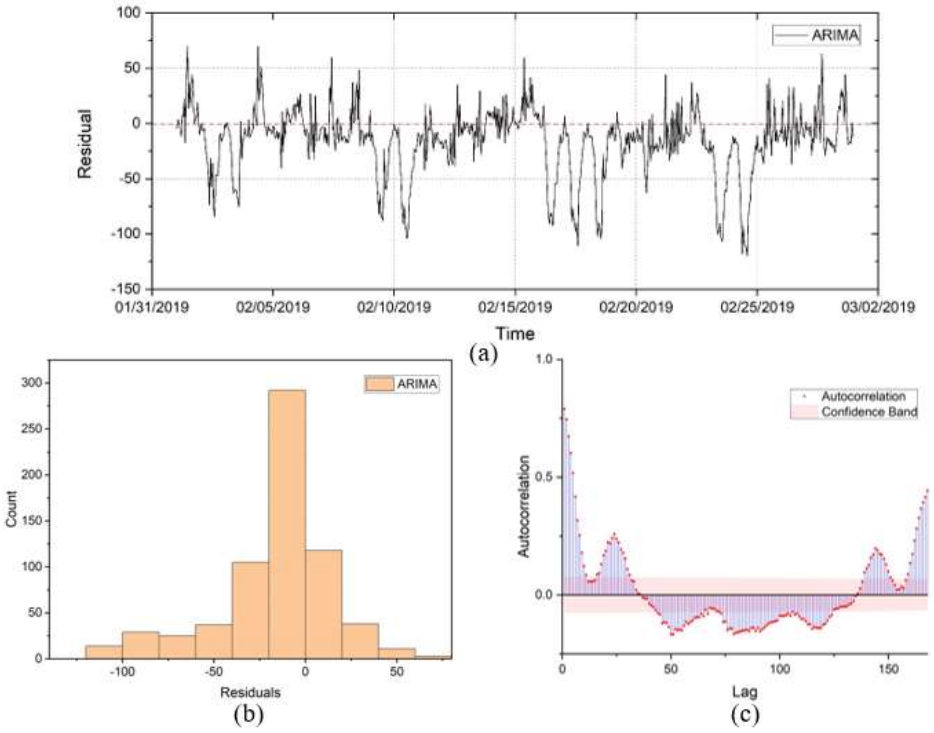

Figure 19. (a) Residuals from ARIMA forecasting, (b) histogram of residuals from ARIMA forecasting, (c) ACF plot of residuals from ARIMA forecasting

The residual diagnostics of all three forecasting methods for February are presented in Figure 19, Figure 20 and Figure 21. As the figure indicates, the residuals from ARIMA forecasting does not have zero mean. The mean is -15.892, which is relatively far from zero. In addition, the variance of the residuals has several drastic changes, which are caused by the incapability of ARIMA to capture hourly power pattern during weekends. In contrast, residuals from TBATS and ANN forecasting appear to have zero mean and constant variance according to the figures except a few outliers around February 18th, where a sudden drop in power occurs. The means of the residuals are 3.169 and 0.908 for TBATS and ANN method respectively, which are virtually zero. From histograms of the residuals, it can be seen that residuals from ARIMA forecasting are not normally distributed while the distribution of residuals from the other two methods are close to normal distribution. The Autocorrelation Function (ACF) plots are used to inspect whether the residuals are correlated. It can be seen that correlation between residuals exist for all three

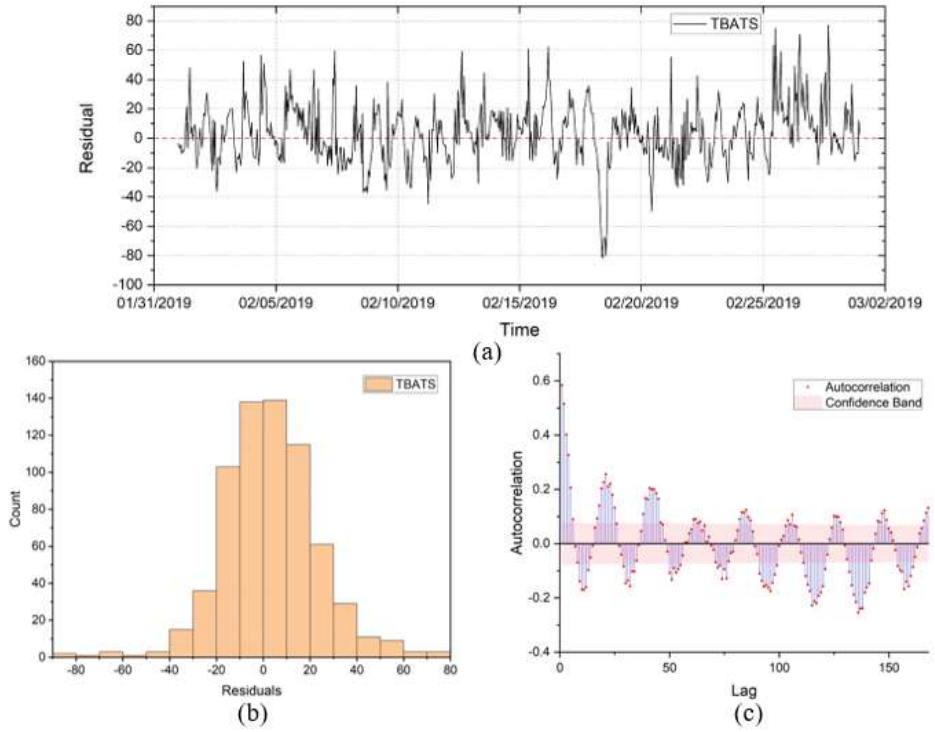

Figure 20. (a) Residuals from TBATS forecasting, (b) histogram of residuals from TBATS forecasting, (c) ACF plot of residuals from TBATS forecasting

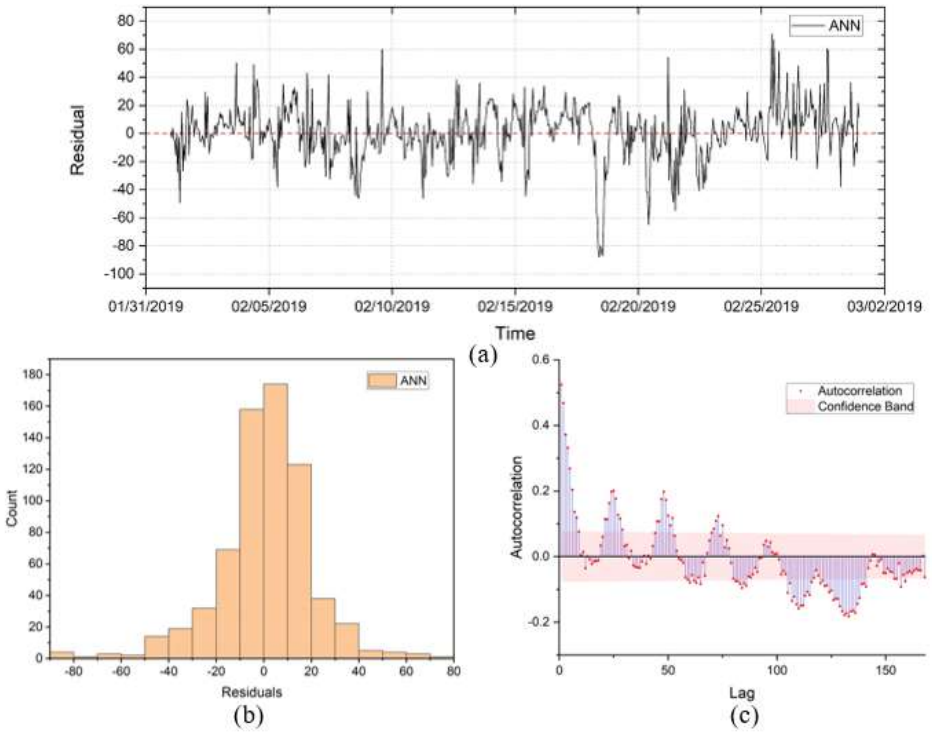

Figure 21. (a) Residuals from ANN forecasting, (b) histogram of residuals from ANN forecasting, (c) ACF plot of residuals from ANN forecasting

forecasting methods since a large portion of the autocorrelation coefficients fall outside of the confidence band. In other words, the prediction models have not perfectly captured the information in the data. Weather condition parameters could serve to assist load forecasting due to their correlations with energy consumption, but they had been excluded in previous analysis. Operation and activity schedules of BioSci3 building may be helpful and their effectiveness can be examined in future study.

The hourly electric power prediction of each methods for February is taken as an example for analysis as shown in Figure 22. ARIMA fails to capture the load pattern in weekends while TBATS and ANN successfully recognize the pattern, although forecasting errors are still relatively large during weekends. It is worth mentioning that spikes in power during noon are rarely captured by the prediction models. Comparing the average evaluation metrics of TBATS

and ANN shown in Table 9 and Table 19, the ANN model is approximately 5~20% more accurate than TBATS model.

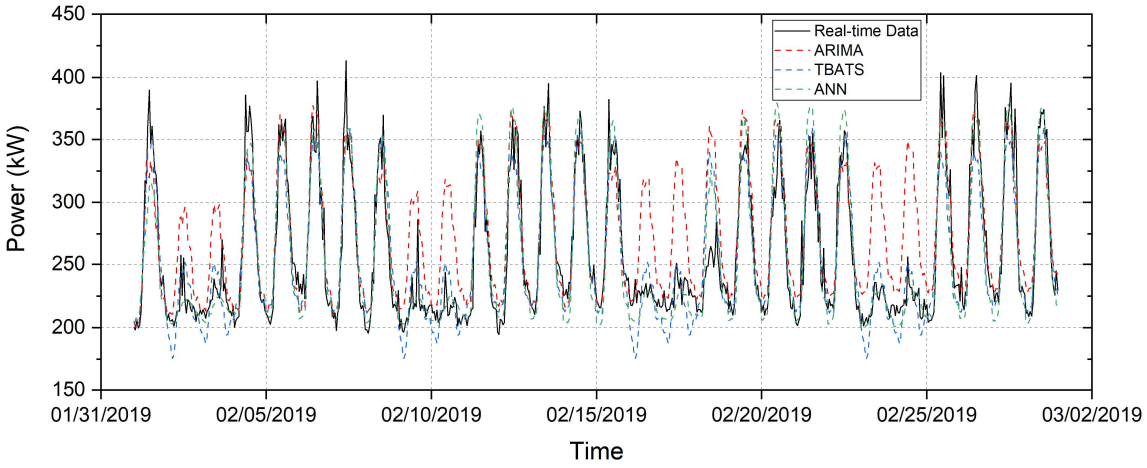

Figure 22. Comparison between real and predicted hourly electric power in February 2019

CHAPTER 6. CONCLUSION

A case study was developed for an actual LEED-certified building, BioSci3, on UCI campus to investigate its real-time energy consumption and prediction using various data-driven methods. BioSci3 is a multi-purpose laboratory with lecture halls and the average power consumption is roughly 300 kW throughout the year of 2019. Various energy saving measures, including occupancy-sensor lighting and demand-controlled ventilation, have been applied to BioSci3, for which it has earned a LEED Platinum certification. Hourly electric power demand data of the building in 2019 was obtained through a web-based monitoring system using existing meters for forecasting in this study. A total of 8,760 data points was used, representing hourly electric power loads in 2019. To compare various methods, RMSE, CV, MAPE and R^2 were calculated for performance evaluation. AIC was introduced as a guideline to determine the optimal model for certain forecasting methods.

For polynomial regression, least squares estimation was utilized to determine the coefficients of the fitted polynomial function based on MATLAB. Data of January was selected for the method and data of each day was fitted separately because polynomial regression cannot handle a full month of data. In total, 31 days of data was fitted and stitched together for a whole month of fitted values. The polynomial obtained through the data fitting of each day was then tested for forecasting the power of the next day. Because of the large order of the polynomial, the values exponentially increase and hence forecasting was not feasible. Using similar approach, data of a week was fitted, and forecasting was attempted. The attempt failed due to the same reason of daily data fitting that the order of the polynomial caused the forecasted values with increased inputs blow up.

ARIMA is a widely used approach for time series prediction. It makes predictions based solely on past values. An automatic algorithm of Python was utilized to decide the optimal model for the data. The determined model for forecasting is denoted as $\mathrm{SARIMA}(3,0,0)(4,1,3)_{24}$, meaning that the model has three autoregressive terms, four seasonal autoregressive terms, one seasonal differencing, and three seasonal moving average terms with a seasonality period of 24. January's data was fed to the model to forecast power in February. The forecasting accuracy yielded by the SARIMA model is unsatisfactory because ARIMA method cannot handle multiple seasonalities in the hourly power data.

To address the problem of ARIMA, TBATS is introduced. Similar to ARIMA, TBATS does not require exogenous parameters to forecast. The TBATS model was obtained through a Python library package that realizes the algorithm for the model determination process described in the particular paper. The definitive model for predicting power in February with January's data is $\mathrm{TBATS}(0, NA, 2,4, \{24,8\}, \{168,6\})$. The forecasting accuracy of the TBATS model is relatively satisfactory as it addressed the problem of ARIMA by properly capturing the load patterns of weekends. The data of the rest of the months were then predicted using the preceding month's data. After eliminating outliers and holidays, the average RMSE, CV, MAPE, and R^2 were found to be 25.295, 9.384%, 7.417%, and 0.789, respectively. Up to nine preceding months of data was used to predict the power of three different months to investigate the performance. It was found that prediction accuracy was substantially improved by 5~20% depending on evaluation metrics when multiple months of data were used. However, extensive examination cannot be conducted due to limited access to data.

In comparison, backpropagation ANN was employed to forecast electric loads. A spectrum of candidates was introduced as the inputs of the ANN. To identify effective inputs, each input

candidate was examined by fitting January power data and comparing performance metrics. A temporary ANN configuration was used for testing purpose. In the end, five inputs were considered essential for the network: total hour, hour, day of week, weekday/weekend, and previous week same hour power. In a similar way, the activation functions of the network were found to be ReLU/sigmoid combination, as well as the number of hidden layers and neurons, which are three and 10, respectively. Adam was chosen as the optimizer due to its capability of fast convergence. The hourly electric load in each month in 2019 starting from February was forecasted by this ANN with one preceding month's data, showing satisfactory prediction. The average RMSE, CV, MAPE and R^2 are 23.339, 8.655, 6.101, and 0.822, respectively with outliers and holidays removed. In addition, training with multiple months' data was introduced and tested using October data as the sample. In contrast to the TBATS method, prediction with multiple months' data worsened the accuracy and best performance was achieved by only training with data of only one preceding month. ANN can also address holidays in the data by introducing a holiday indicator input, which cannot be achieved by TBATS.

Residuals were inspected for all the forecasting methods through the diagnostics figures using prediction for February power as the sample. The residuals have a nonzero mean and changing variance for ARIMA method while the other two have approximately zero mean and constant variance. The residuals in the ARIMA forecast are not normally distributed while those from TBATS and ANN forecast are nearly normal distribution. The residuals in all three methods appear to have correlations as implied by the ACF plots, which suggests that the models have not comprehensively captured the information in the data.

Overall, the ANN's forecasting accuracy is found to be about 5~20% better than TBATS' when only using one month's data for training. They both satisfy and tremendously exceed the

requirement by ASHRAE, i.e. lower than 30% CV for hourly granularity. For future study, operation and activity schedules can be introduced as ANN inputs to further improve its forecasting accuracy.

REFERENCES

[1] U.S. Energy Information Adminstration, 2019, "Global Energy Consumption Driven by More Electricity in Residential, Commercial Buildings" [Online]. Available: https://www.eia.gov/todayinenergy/detail.php?id=41753. [Accessed: 18-May-2020].

[2] California Public Utilities Commission, 2017, "Zero Net Energy" [Online]. Available: https://www.cpuc.ca.gov/ZNE/. [Accessed: 14-May-2020].

[3] Chauhan, A., and Saini, R. P., 2014, "A Review on Integrated Renewable Energy System Based Power Generation for Stand-Alone Applications: Configurations, Storage Options, Sizing Methodologies and Control," Renew. Sustain. Energy Rev., **38**, pp. 99–120.

[4] Wang, Y., Chen, K. S., and Cho, S. C., 2013, *PEM Fuel Cells: Thermal and Water Management Fundamentals*, Momentum Press.

[5] Wang, Y., Chen, K. S., Mishler, J., Cho, S. C., and Adroher, X. C., 2011, "A Review of Polymer Electrolyte Membrane Fuel Cells: Technology, Applications, and Needs on Fundamental Research," Appl. Energy, **88**(4), pp. 981–1007.

[6] Ham, S. W., Jo, S. Y., Dong, H. W., and Jeong, J. W., 2015, "A Simplified PEM Fuel Cell Model for Building Cogeneration Applications," Energy Build., **107**, pp. 213–225.

[7] Ellamla, H. R., Staffell, I., Bujlo, P., Pollet, B. G., and Pasupathi, S., 2015, "Current Status of Fuel Cell Based Combined Heat and Power Systems for Residential Sector," J. Power Sources, **293**, pp. 312–328.

[8] Wang, Y., and Cho, S. C., 2014, "Analysis and Three-Dimensional Modeling of Vanadium Flow Batteries," J. Electrochem. Soc., **161**(9), pp. A1200–A1212.

[9] Zheng, M., Sun, J., Meinrenken, C. J., and Wang, T., 2019, "Pathways toward Enhanced Techno-Economic Performance of Flow Battery Systems in Energy System Applications,"

J. Electrochem. Energy Convers. Storage, **16**(2).

[10] Kear, G., Shah, A. A., and Walsh, F. C., 2012, "Development of the All-Vanadium Redox Flow Battery for Energy Storage: A Review of Technological, Financial and Policy Aspects," Int. J. Energy Res., **36**(11), pp. 1105–1120.

[11] D'Agostino, R., Baumann, L., Damiano, A., and Boggasch, E., 2015, "A Vanadium-Redox-Flow-Battery Model for Evaluation of Distributed Storage Implementation in Residential Energy Systems," IEEE Trans. Energy Convers., **30**(2), pp. 421–430.

[12] Azharuddin Shamshuddin, M., Babu, T. S., Dragicevic, T., Miyatake, M., and Rajasekar, N., 2017, "Priority-Based Energy Management Technique for Integration of Solar PV, Battery, and Fuel Cell Systems in an Autonomous DC Microgrid," Electr. Power Components Syst., **45**(17), pp. 1881–1891.

[13] Adroher, X. C., and Wang, Y., 2011, "Ex Situ and Modeling Study of Two-Phase Flow in a Single Channel of Polymer Electrolyte Membrane Fuel Cells," J. Power Sources, **196**(22), pp. 9544–9551.

[14] Amasyali, K., and El-Gohary, N. M., 2018, "A Review of Data-Driven Building Energy Consumption Prediction Studies," Renew. Sustain. Energy Rev., **81**(March 2017), pp. 1192–1205.

[15] Fang, T., and Lahdelma, R., 2016, "Evaluation of a Multiple Linear Regression Model and SARIMA Model in Forecasting Heat Demand for District Heating System," Appl. Energy, **179**, pp. 544–552.

[16] Bagnasco, A., Fresi, F., Saviozzi, M., Silvestro, F., and Vinci, A., 2015, "Electrical Consumption Forecasting in Hospital Facilities: An Application Case," Energy Build., **103**, pp. 261–270.

[17] Wang, L., Kubichek, R., and Zhou, X., 2018, "Adaptive Learning Based Data-Driven Models for Predicting Hourly Building Energy Use," Energy Build., **159**, pp. 454–461.

[18] Solomon, D. M., Winter, R. L., Boulanger, A. G., Anderson, R. N., and Wu, L. L., 2011, "Forecasting Energy Demand in Large Commercial Buildings Using Support Vector Machine Regression," Dep. Comput. Sci. Columbia Univ. Tech. Rep. CUCS-040-11.

[19] Brożyna, J., Mentel, G., Szetela, B., and Strielkowski, W., 2018, "Multi-Seasonality in the Tbats Model Using Demand for Electric Energy as a Case Study," Econ. Comput. Econ. Cybern. Stud. Res., **52**(1), pp. 229–246.

[20] Edwards, R. E., New, J., and Parker, L. E., 2012, "Predicting Future Hourly Residential Electrical Consumption: A Machine Learning Case Study," Energy Build., **49**, pp. 591–603.

[21] Yokoyama, R., Wakui, T., and Satake, R., 2009, "Prediction of Energy Demands Using Neural Network with Model Identification by Global Optimization," Energy Convers. Manag., **50**(2), pp. 319–327.

[22] Fan, S., Member, S., and Hyndman, R. J., 2010, "Short-Term Load Forecasting Based on a Semi-Parametric Additive Model," IEEE Trans. Power Syst., (August), pp. 1–8.

[23] Zhao, D., Zhong, M., Zhang, X., and Su, X., 2016, "Energy Consumption Predicting Model of VRV (Variable Refrigerant Volume) System in Office Buildings Based on Data Mining," Energy, **102**, pp. 660–668.

[24] Catalina, T., Iordache, V., and Caracaleanu, B., 2013, "Multiple Regression Model for Fast Prediction of the Heating Energy Demand," Energy Build., **57**, pp. 302–312.

[25] Leung, M. C., Tse, N. C. F., Lai, L. L., and Chow, T. T., 2012, "The Use of Occupancy Space Electrical Power Demand in Building Cooling Load Prediction," Energy Build., **55**,

pp. 151–163.

[26] Borges, C. E., Penya, Y. K., Ferńandez, I., Prieto, J., and Bretos, O., 2013, "Assessing Tolerance-Based Robust Short-Term Load Forecasting in Buildings," Energies, **6**(4), pp. 2110–2129.

[27] Neto, A. H., and Fiorelli, F. A. S., 2008, "Comparison between Detailed Model Simulation and Artificial Neural Network for Forecasting Building Energy Consumption," Energy Build., **40**(12), pp. 2169–2176.

[28] Marino, D. L., Amarasinghe, K., and Manic, M., 2016, "Building Energy Load Forecasting Using Deep Neural Networks," IECON Proc. (Industrial Electron. Conf., pp. 7046–7051.

[29] Bektas Ekici, B., and Aksoy, U. T., 2011, "Prediction of Building Energy Needs in Early Stage of Design by Using ANFIS," Expert Syst. Appl., **38**(5), pp. 5352–5358.

[30] Zhang, F., Deb, C., Lee, S. E., Yang, J., and Shah, K. W., 2016, "Time Series Forecasting for Building Energy Consumption Using Weighted Support Vector Regression with Differential Evolution Optimization Technique," Energy Build., **126**, pp. 94–103.

[31] Jain, R. K., Smith, K. M., Culligan, P. J., and Taylor, J. E., 2014, "Forecasting Energy Consumption of Multi-Family Residential Buildings Using Support Vector Regression: Investigating the Impact of Temporal and Spatial Monitoring Granularity on Performance Accuracy," Appl. Energy, **123**, pp. 168–178.

[32] Liu, H., Chen, J., Hissel, D., and Su, H., 2019, "Remaining Useful Life Estimation for Proton Exchange Membrane Fuel Cells Using a Hybrid Method," Appl. Energy, **237**, pp. 910–919.

[33] Esmaeili, M., Shayeghi, H., Nejad, H. M., and Younesi, A., 2017, "Reinforcement

Learning Based PID Controller Design for LFC in a Microgrid," COMPEL - Int. J. Comput. Math. Electr. Electron. Eng., **36**(4), pp. 1287–1297.

[34] Wang, Y., Seo, B., Wang, B., Zamel, N., Jiao, K., and Adroher, X. C., 2020, "Fundamentals, Materials, and Machine Learning of Polymer Electrolyte Membrane Fuel Cell Technology," Energy AI, **1**, p. 100014.

[35] UCI Environmental Health&Safety, "UC Irvine Smart Labs Initiative" [Online]. Available: https://www.ehs.uci.edu/programs/energy/Bio Sci 3 Fact sheet.pdf. [Accessed: 20-Apr-2020].

[36] UCI Sustainability, "Green Buildings" [Online]. Available: https://sustainability.uci.edu/sustainablecampus/green-buildings/. [Accessed: 28-Mar-2020].

[37] Landsberg, D. R., Shonder, J. A., Barker, K. A., Haberl, J. S., Judson, S. A., Jump, D. A., Koran, W. E., Hall, R. L., Reindl, D. T., Anderson, J. R., Barnaby, C. S., Clark, J. A., Dunlap, J. F., Earley, J. W., Emmerich, S. J., and Graef, P. T., 2014, *Measurement of Energy, Demand, and Water Savings*.

[38] Hyndman, R. J., and Athanasopoulos, G., 2018, *Forecasting: Principles and Practice*, OTexts, Melbourne.

[39] Brockwell, P. J., and Davis, R. A., *Introduction to Time Series and Forecasting*, Springer.

[40] Rob J. Hyndman, and Yeasmin Khandakar, 2008, "Automatic Time Series Forecasting: The Forecast Package for R," J. Stat. Softw., **27**(3), p. 22.

[41] de Livera, A. M., Hyndman, R. J., and Snyder, R. D., 2011, "Forecasting Time Series with Complex Seasonal Patterns Using Exponential Smoothing," J. Am. Stat. Assoc., **106**(496), pp. 1513–1527.

[42] Neapolitan, R. E., and Neapolitan, R. E., 2018, “Neural Networks and Deep Learning,” Artif. Intell., pp. 389–411.

[43] Dolinar, M., Vidrih, B., Kajfež-Bogataj, L., and Medved, S., 2010, “Predicted Changes in Energy Demands for Heating and Cooling Due to Climate Change,” Phys. Chem. Earth, **35**(1–2), pp. 100–106.

[44] Ruder, S., 2016, “An Overview of Gradient Descent Optimization Algorithms,” pp. 1–14.