IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 7, 2021, accepted February 15, 2021, date of publication February 18, 2021, date of current version March 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3060290

On Short-Term Load Forecasting Using Machine Learning Techniques and a Novel Parallel Deep LSTM-CNN Approach

BEHNAM FARSI[1], MANAR AMAYRI[2], NIZAR BOUGUILA[1], (Senior Member, IEEE), AND URSULA EICKER[3]

[1]Concordia Institute for Information Systems Engineering(CIISE), Concordia University, Montreal, QC H3G 1M8, Canada
[2]G-SCOP Lab, Grenoble Institute of Technology, 38185 Grenoble, France
[3]Department of Building, Civil and Environmental Engineering, Concordia University, Montreal, QC H3G 1M8, Canada

Corresponding author: Behnam Farsi (behnam.farsibalouch@mail.concordia.ca)

ABSTRACT Since electricity plays a crucial role in countries' industrial infrastructures, power companies are trying to monitor and control infrastructures to improve energy management and scheduling. Accurate forecasting is a critical task for a stable and efficient energy supply, where load and supply are matched. This article discusses various algorithms and a new hybrid deep learning model which combines long short-term memory networks (LSTM) and convolutional neural network (CNN) model to analyze their performance for short-term load forecasting. The proposed model is called parallel LSTM-CNN Network or PLCNet. Two real-world data sets, namely ''hourly load consumption of Malaysia '' as well as ''daily power electric consumption of Germany'', are used to test and compare the presented models. To evaluate the tested models' performance, root mean squared error (RMSE), mean absolute percentage error (MAPE), and R-squared were used. In total, this article is divided into two parts. In the first part, different machine learning models, including the PLCNet, predict the next time step load. In the second part, the model's performance, which has shown the most accurate results in the first part, is discussed in different time horizons. The results show that deep neural networks models, especially PLCNet, are good candidates for being used as short-term prediction tools. PLCNet improved the accuracy from 83.17% to 91.18% for the German data and achieved 98.23% accuracy in Malaysian data, which is an excellent result in load forecasting.

INDEX TERMS Electricity, smart grids, load consumption, short-term load forecasting, deep learning, time series, regression, convolutional neural networks, long short-term memory.

NOMENCLATURE

ANN :	Artificial Neural Network
ARIMA :	Autoregressive Integrated Moving Average
CNN :	Convolutional Neural Network
DNN :	Deep Neural Network
ETS :	Exponential Smoothing
LSTM	Long Shot-Term Memory
LTLF :	Long-Term load forecasting
MLR :	Multiple Linear Regression
MTLF :	Medium-Term Load Forecasting
RNN :	Recurrent Neural Network
SARIMA :	Seasonal ARIMA
STLF :	Short-Term Load Forecasting
SVR :	Support Vector Machine

The associate editor coordinating the review of this manuscript and approving it for publication was Li Zhang.

I. INTRODUCTION

According to the IEA report [1], in 2017, world electricity consumption reached 21,372 TWh, which is 2.6% higher than 2016 electricity consumption. Such an annual increase creates a new problem: how to reduce consumption? Nowadays, many companies are working on this problem and trying to solve it. Demand Response Management, which is one of the main features in smart grids [2], helps to control electricity consumption with the focus on the customer side. It is also more essential to understand residential and non-residential building demand and the use of electricity. Carrying out a reduction in load consumption can lead to a high number of economic and environmental benefits. Since experts aim to create some automated tools that are able to deliver energy very efficiently, they introduced load forecasting methods as alternative solutions for electricity network augmentation as it can be useful to manage the electricity demand and

provide more energy efficiency [3]. In addition, improving power delivery quality and having secure networks is a critical task in smart grids to monitor and support advanced power distribution systems [4] and, in particular, to improve load forecasting. Since future consumption could be predicted, they can be considered as tools to minimize the gap between electricity supply and user consumption. However, an inaccurate prediction may lead to a huge loss. For instance, a small percentage of increase in forecast error was predicted in 1985, which led to more than 10 million pounds of yearly detriment in the thermal UK power systems [5]. Thus, many big companies have focused on accurate load forecasting and load management so that the Energy Supply Association of Australia, as an instance, invested about 80% of its budget on grid upgrades.

Load forecasting approaches are categorized in three different groups concerning their functionalities for different purposes [7]: Short-term Load Forecasting (STLF) [8], Medium-term Load Forecasting (MTLF) and Long-term Load Forecasting (LTLF) [9]. STLF forecasts the following hour load to next week, while in MTLF, it is more than one week to few months, and LTLF forecasts next years load consumption. For each of these methods, there are diverse factors that influence the prediction.

Due to the ability of STLF approaches, they have tremendous importance in energy management. Hence, they have been used to provide proper management in electric equipment, and because of this contribution, they are known as an inevitable component in Energy Management Systems. An error in STLF can have an immediate impact on electrical equipment. Several factors affect the STLF, including the following ones: (1) Time factor [6], which is the most crucial factor for STLF because of the existence of some patterns such as daily patterns in a set of data (2) Climate, which contains temperature and humidity [6]. (3) Holidays can make considerable changes in electricity demand. However, this article focuses on time as a factor that influences electricity usage and can help achieve accurate predictions.

Besides, diverse approaches can be applied to time-series data to carry out accurate short-term forecasting. These approaches consist of statistical regression models, classic time-series models, and deep learning models. However, in addition to the factors as mentioned earlier, there are other factors such as the size of the house, the age of appliances and equipment [6], global factors like diseases, which can affect the load prediction for medium-term and long-term forecasting. Still, most approaches have the same attributes with some subtle differences. Load consumption data sets can be viewed as time series. time-series have specific attributes, such as Trend, Seasonality, and Noise, which will be discussed later. Due to numerous challenging problems when dealing with time-series data, researchers deployed Artificial Neural Networks (ANN) [18] which have structures like the human brain. They are available to be used in various areas such as Natural Language Processing (NLP) [10], audio recognition [11], medical [12] and load forecasting [19] and they have been successful to achieve impressive results. One of the challenging problems of ANN is that they need large scale data for training to learn the models. Therefore, in some cases, regression-based approaches can be more useful.

Different algorithms for load forecasting have been studied so far. The authors in [6] prepared an overview of different types of load forecasting methods. They focused on the difference between short-term, medium-term, and long-term load forecasting and the factors which affect them. The authors in [14] discussed a review of load forecasting with a focus on regression-based approaches. To forecast day-ahead hourly electricity load, they used two particular data sets from the University of New South Wales, one from the Kensington Campus and the Tyree Energy Technologies Building. The authors in [7] surveyed different deep learning models for short-term load forecasting. They evaluated seven deep learning models on three real-world data sets. They proved that there is no correlation between the complexity of models and golden results. Even in some cases, simpler models can achieve better results in short-term load forecasting. The authors in [15] proposed a hybrid deep neural network consisting of a CNN module, an LSTM module, and a feature-fusion module and tested it on the 2-year electric load data sets from the Italy-North area. Their model was compared with some other machine learning models, including: Decision Tree, Random Forest, DeepEnergy, LSTM, and CNN coming with better results. However, even though they achieved good results from their proposed model, their used data set was not challenging. The authors did not challenge the model with a more complex data set or prediction in different time horizons. The author in [16] proposed a parallel CNN-RNN model to predict one day ahead of load consumption. Temperatures, holidays, hours of day, and days of the week were used as features for the historical load series and achieved better results than regression-based models, DNN and CNN-RNN. However, due to the existing vanishing gradient descent problem in RNNs, they are not suitable enough to be used in load forecasting applications, and LSTM networks should replace them. RNNs and the specific type of their family, LSTM, use control theory in their structure. They can find the dependency between old data and new ones and become an interesting network for load forecasting applications in recent years. [17] has studied RNNs models well. The authors in [20] proposed a new DeepEnergy model that combines 1-D CNN to extract the features and fully connected network to forecast future load data. To forecast the next three days' data, they used an hourly electricity consumption data set from the USA. They compared the proposed model's result with five other machine learning techniques through RMSE and MAPE. The results showed that the DeepEnergy model could carry out an accurate short-term load forecasting compared to other models. After the DeepEnergy model, the Random Forest technique [21] had a good performance. However, as there is no LSTM network in this model, it will have some difficulty working with more complex time-series data, and it can be expected that this model

may fail to make an accurate prediction. In another study [23], the authors proposed a new model which consists of three algorithms including Variational Mode Decomposition (VMD), Convolutional Neural Network (CNN), and Gated Neural Network (GRU). For more convenience, they called the proposed model SEPNet. This model aimed to predict hourly electric price and evaluate the model, hourly data from city Newyork, the USA, which includes the hourly electricity price from 2015 to 2018. Compared to other models such as LSTM, CNN, VMD-CNN, the SEPNet model performed better where it improved the RMSE and MAPE by 25% and 19%, respectively. The existing problem with SEPNet is that CNN and GRU are designed consecutively in this model, so the processed data from CNN may affect the GRU network's performance. Some authors, such as in [24] used ANNs to forecast other types of load data like PV system output data. They proposed a powerful CNN based model called PVPNet, and they evaluated the proposed model by using daily data from 2015. They used three past days' information to predict the next 24h and their model has outperformed Random Forest (RF) regarding the mean absolute error (MAE) and RMSE. Besides, with technology development, many studies deployed machine learning models in IoT. In terms of the technical part, if these models are supposed to be used in IoT, they must perform online load forecasting. [27] presents some related machine learning methods which can be used in IoT through the cloud. They also implemented a novel hardware technology, including the Arduino microcontroller. They implemented the device in a research lab to predict total power consumption in the lab. Regarding the algorithms, Linear Regression, SVM Regression, Ensemble Bagged, Ensemble Boosted, Fine Tree Regression, and Gaussian Process Regression (GPR) have been used. All of the mentioned models have performed appropriately.

Even though some studies in recent years have discussed different models for short-term load forecasting [22], the lack of a comprehensive article to carry out a comparison between classic time-series models, regression-based models, and deep learning is completely obvious. Besides, time-series must be used correctly as input for machine learning models. In other words, some analysis of data is essential to compile machine learning models. This article 's contribution is that it focuses on different models that are appropriate to be used for load forecasting, and it also reviews some different methodologies to find out the most effective models for forecasting applications. Even though various deep learning models have been introduced for load forecasting in recent years, only some have succeeded in achieving state-of-the-art results. Moreover, this article consecutively proposes a new hybrid parallel CNN-LSTM and LSTM-Dense neural networks to improve load forecasting accuracy. Regarding the model's architecture, it consists of two different paths (CNN and LSTM). CNN path extracts the input data features, and the LSTM path learns the long-term dependency within input data. After passing through these two paths and merging their outputs, a fully connected path

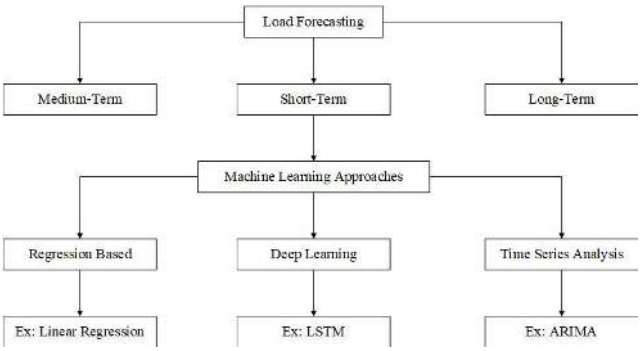

FIGURE 1. An overview of the article.

combined with an LSTM layer has been implemented to process the output to predict final load data. This article aims to evaluate various machine learning techniques in STLF tasks while there are no exogenous variables. In other words, it tries to find out a way to carry out STLF using just previous load data and compare all the results with each other. To extend this study, all the models are implemented to forecast daily and hourly ahead load consumption using two highly aggregated data sets, one of which is an hourly power consumption from the city of Johor in Malaysia [25]. The other one is a daily electric consumption which is collected from a power supply company in Germany. In order to evaluate the models, this article uses root mean squared error (RMSE) due to the ability to show how much predicted values spread around average and mean absolute percentage error (MAPE) since MAPE can present the accuracy of the models and R-Squared to show the correlation between predicted results and actual value.

The remainder of the article is organized as follows: section II discusses how time-series data and associated models work. Section III, in addition to discussing data pre-processing, elaborates the models and show the results. Finally, a conclusion is issued in section IV.

II. BACKGROUND OF STUDY

Load series data usually have particular attributes. Thus, before forecasting future load consumption, these attributes must be studied and discussed as follows.

A. DEFINITIONS

As it has been mentioned before, load consumption data are time series. Thus, to forecast future load consumption, some time-series analyses are needed. time-series have important attributes such as trend or noise. In order to predict future load consumption, some considerations of time-series are needed to be taken into account.

1) TREND

Some time-dependent data have a linear trend in the long term. It means there is an increase or decrease during the whole time, which may not be in the same direction throughout the given period. However, overall, it will be upward, downward or stable. Load series data are an excellent example of a kind of tendencies of movement.

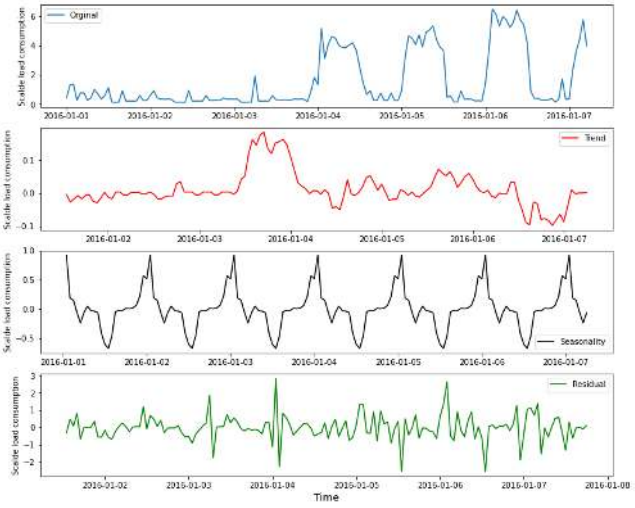

FIGURE 2. Original load series data and its decomposition. As this data is an hourly data, it has every 24 hours seasonality.

2) SEASONALITY

Data with seasonality or periodic fluctuations at a particular time repeat themselves. Many time-dependent data in a specific time have the same behavior. These kinds of data are called seasonal data, and studying the seasonality within time-series data is an important task.

3) RESIDUALS (NOISE)

The combination of trend, seasonality, and residual create the time dependent data, i.e., if the data decomposes to seasonality and trend, residuals (noise) will remain by subtracting both trend and seasonality.

4) STATIONARY

A stationary time-series does not depend on observed time. In other words, a stationary time-series does not have a pattern to predict the future by looking at it. If data is stationary, it is easier to be processed and predict the future load data.

Most load series data have all trends, seasonal, noise attributes simultaneously. For instance, figure 2 shows decomposition of a seasonal load series data. The blue plot shows original data, and the red plot shows trend, the black plot shows seasonality of data, and the green plot is noise. A library from Python called *seasonal* − *decompose*() had been used to decompose all the seasonal data in this article. This function returns an object array including seasonal, trend, and residuals. There is a *freq* variable in this function, which refers to the input data frequency, and it is important to assign a number to this variable. For instance, the *freq* is 24 for hourly data.

B. MACHINE LEARNING MODELS

Due to the importance of STLF, many authors have discussed how an accurate prediction of future load consumption can be obtained; thus, different methods have been introduced for this purpose such as Auto-Regressive Integrated Moving Average (ARIMA), Seasonal Auto-Regressive Integrated Moving Average (SARIMA), Regression, Artificial Neural Networks (ANN), etc. However, in recent years, ANNs have become widespread in STLF, and they achieved good results. Table 1 shows some studies on deep neural networks with different architectures to carry out an STLF task with some cities' historical electric load consumption. Different models including ANN, regression, and classic time-series analysis approaches will be discussed in the following.

TABLE 1. Different Neural Networks Architectures That are Widely Deployed for STLF With Datasets From Around the World. CNN: Convolutional Neural Networks, FTS: Fuzzy Time Series, LSTM: Long Short-Term Memories, DNN: Deep Neural Networks, FNN: Feedforward Neural Networks, GRU: Gated Recurrent Unit.

Reference	Model	City(Dataset)
[25]	FTS-CNN	Johor, Malaysia
[25]	CNN-RNN	North Italy
[13]	Parallel CNN-RNN	North China
[25]	LSTM	Johor, Malaysia
[13]	DNN-FNN	New york, USA
[26]	Seq2Seq	New England
[26]	GRU	New England

1) AUTO REGRESSIVE MODELS

Auto-regressive models predict future value using the correlation between future value and past value. An important forecasting method, Box-Jenkins methodology combines the Auto-Regressive model (AR) with Moving Average (MA). This integrated model is called Auto Regressive Moving Average (ARMA). When a differencing order is added to this model to remove non-stationary within data, it is called Auto-Regressive Differencing Moving Average (ARIMA). Some studies done by authors such as in [28] discuss the Box-Jenkins method for short-term load forecasting. However, some of these methods have been modified to achieve accurate results. [28] used a modified ARIMA to forecast hourly load consumption and have completed better results than standard ARIMA. The authors used the load data and temperatures from operators in Iran, and MAPE was between 1.5% and 2.0% while MAPE for standard ARIMA was higher (between 2.0% and 4.5%). In the following, ARIMA models are discussed.

AR: In the Auto-regressive model, the future variable will be predicted from the past variables. This model has an order, p, which is the number of immediately preceding values in the series used to predict the value at a time t. AR can be formalized as follows:

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \ldots + \beta_p y_{t-p} + \mu_t \quad (1)$$

where μ_t is the mean of series, β_p are the parameters of models and y_t is data at time t.

MA: Moving Average is an indicator of technical analyst and used widely to smooth noise based on lagging. The order q in MA models refers to q previous errors. MA can be formalized as following:

$$X_t = \theta_0 + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_n \epsilon_{t-q} + \epsilon_t \quad (2)$$

where the θ_q are the parameters of models and ϵ_t are the errors until time t.

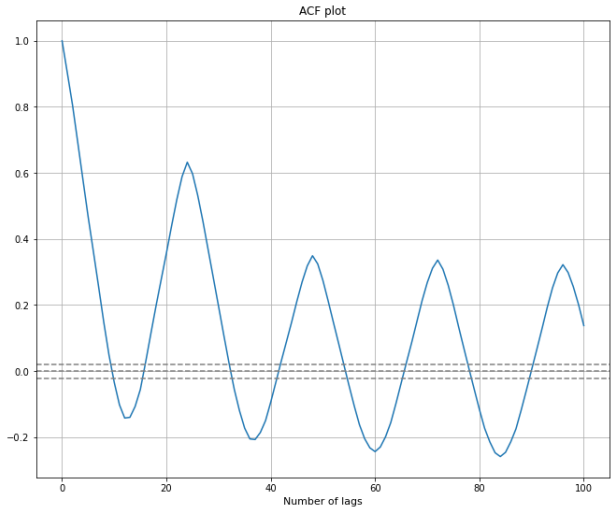

FIGURE 3. ACF plot for an example data. X axis shows number of lags, Y axis shows amount of auto-correlation.

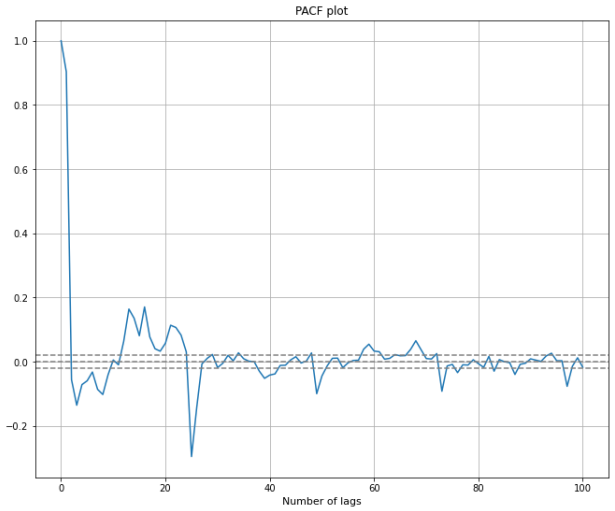

FIGURE 4. PACF plot for same data in figure 3.

Integration: To predict future load consumption with ARIMA model, a stationary time-series should be used. There are different methods to stabilize a time series, such as logarithm or differencing. These operators reduce time-series changes with trend elimination; in other words, they convert non-stationary data to stationary data. An ARIMA model usually is written as ARIMA(P,D,Q) to show the needed orders, which should be used to achieve the best results from this model. D represents the number of integration used, P and Q represent the orders of AR and MA part of ARIMA. To find out the values of P, D, and Q, there are different approaches. However, many experts suggest using auto-correlation (AC) and Partial auto-correlation (PAC) plots to figure out the values of P and Q. Nevertheless, first of all, it is necessary to find out what AC is precise. AC is the degree of similarity between a time-series data and its lags (see figure 3), and it takes a value in the range $[-1,1]$. If there is any seasonality in data, remarkable spikes in the AC plot are shown. For instance, figure 3 shows the AC plot (or auto-correlation function (ACF) plot) of an hourly load consumption from a smart building in France. This data is an hourly load consumption data, and because of this hourly attribute, a seasonal approach every 24 hours can be seen in this figure.

Likewise, P or, in other words, the order of AR which is a part of ARIMA model can be found by plotting PAC plot (or partial auto-correlation function (PACF) plot). Figure 4 shows the PAC plot for same data in figure 3.

However, there are some other tests to find the best values for D. To figure out whether the data is stationary or not, two different tests are proposed: The rolling statistic plot test and the Dickey-Fuller test. The rolling statistic plot test is a chart analysis technique to examine collected data by plotting Rolling Average. Figuring out the existence of a trend in the Rolling average is the primary objective. Provided there is not any trend, data is determined as stationary. In figure 5, the blue plot shows the original data, and the red plot shows the rolling average of data. Since there is no trend in the

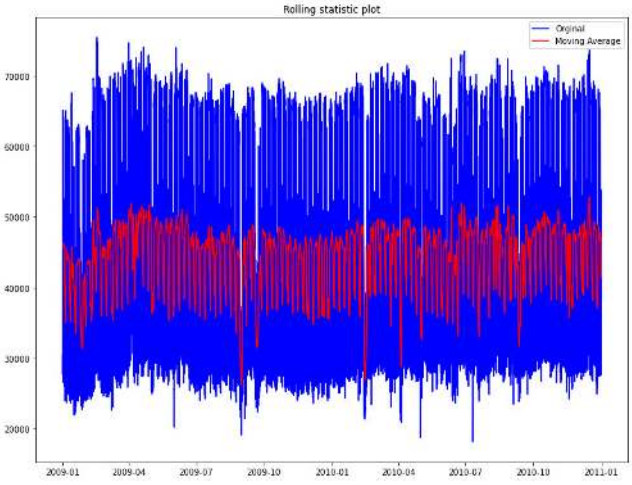

FIGURE 5. An example of Rolling test which proves that data is stationary.

rolling average plot (red plot), the data is stationary. Besides, a Dickey-Fuller test has been applied to this data. This test is based on a null hypothesis in which the nature of the series (i.e., stationary or not) could be determined by evaluating the p-value received by the Dickey-Fuller test. The p-value is considered as a critical value for rejecting the null hypothesis. Thus, the smaller p-value provides more robust evidence to accept the alternative hypothesis. In this example, the confidence interval is supposed 5%, and after applying the test, the obtained p-values are less than 0.05, so data can be considered stationary.

Hence, one way to obtain the best values for D, after any integration of data, Rolling tests and Dickey-Fuller test can be applied. If these tests prove data is stationary, there is no need to carry out another integration. However, in case the results were different, it demonstrates that data need more integration. It must be said that in some instances achieving stationary data is not possible. Therefore, this type of data cannot work with ARIMA models.

Seasonal ARIMA or SARIMA is another kind of statistical model that is widely used in seasonal data cases. In addition to the same parameters with ARIMA (P,D,Q)) four other parameters for the seasonal part of these models are p, d, q and m. Like ARIMA, p represents the order of Auto-regressive for the seasonal part, d represents the order of integration for the seasonal part, and q represents the order of Moving Average for the seasonal part. Besides, m shows the time horizon of seasonality. For example, for hourly data, m will be 24, and for daily data, it will be 7. Therefore, SARIMA formulation is usually presented as SARIMA (P,D,Q)(p,d,q,m).

2) EXPONENTIAL SMOOTHING

Exponential Smoothing (ETS) is a well-known time-series forecasting model for power systems. It can be used as an alternative to ARIMA models, in addition to its ability to be used for STLF, MTLF, and LTLF. It uses a weighted sum of past observations to make the prediction. The difference between ETS and ARIMA models is that ETS uses an exponential decreasing weight for previous observations. It means recent observations have a higher weight than past observations. Therefore the accuracy depends on some coefficients. The authors in [29] studied exponential smoothing for load forecasting application using different coefficients. They used six different data sets collected from China to evaluate their model, and as it was assumed, they achieved a high range of MAPE for different coefficient values. There are various types of ETS models that are used due to the complexity of data. Equation (3) indicates the formula of the simple Exponential Smoothing

$$F_{t+1} = \alpha A_t + (1-\alpha)F_t \tag{3}$$

where F_t and F_{t+1} indicate, predicted value in time t and $t+1$ respectively, A_t indicates actual value at time t and α is the smoothing factor ($0 \leq \alpha \leq 1$).

3) LINEAR REGRESSION

Regression-based approaches are interesting techniques, and among all these techniques, linear regression has an inevitable role. Some studies tried to use linear regression for time-series or specifically for load forecasting. The author in [30] studied RGUKT, R.K valley campus for STLF and achieved MAPE = 0.029 and RMSE = 2.453. In another study, the authors in [31] used it with different linear regression models, including multiple linear regression (MLR), Lasso, Ridge for hourly load data.

Linear regression is a statistical method to find the relation among variables. This method is useful to estimate a variable using influence parameters. The most straightforward linear regression equation is as below:

$$Y_i = \beta_0 + \beta_1 X_i + \mu_i \tag{4}$$

where Y is the dependent variable, β_0 is an interceptor, β_1 is the slope, X is the independent variable, and μ_i is residual of the model, which is distributed with zero mean and constant variance. By increasing the number of variables, this model is called multiple linear regression (MLR). In order to evaluate this model, the Least Squared Error (LSE) technique is used. The primary goal is that to find the best coefficients to minimize LSE. LSE evaluates the model by adding squares of error between two variables, which in this case, is between actual values and forecasted ones. Equation (5) shows LSE formula:

$$LSE = \sum_{i=1}^{n} (Y_i - X_i)^2 \tag{5}$$

where X is predicted value, Y is the actual value.

In order to use linear regression for load forecasting, some parameters such as temperature, humidity, time are needed to be used as independent variables. Likewise, the load consumption data are used as dependent variables in linear regression models. With this approach, it is possible to use linear regression to forecast future load consumption. However, there are some ways to forecast load consumption without using exogenous variables. Lags can be used as the independent variable for load forecasting to predict linear regression without using exogenous data. Usually, more than one lag is used as an independent variable, so MLR is used instead of simple linear regression. AC plot is a useful tool for time-series analysis with linear regression. In this approach, those lags in which their auto-correlation values are more than a certain threshold can be used as an independent variable in linear regression. For instance, according to figure 3 lags [1, 2, 3, 24, 25] are chosen as independent variables with amount 0.6 for threshold. In total, in this model, lags are independent variables, and actual load consumption is the dependent variable. An alternative way to increase the model's accuracy is that exogenous variables such as humidity, holiday, and weather are added to the model. Figure 6 shows the process of preparing data and choosing parameters for linear regression models. This approach also is used for SVR and fully connected models too. According to the diagram, data preparation refers to finding missing values and data normalization. As discussed after plotting the AC plot, those lags with higher auto-correlation value than the selected threshold are used as the models' parameters. After choosing lags and parameters, the models can predict again, however, if they are not accurate enough, the amount of the threshold must be changed, and the process is also re-started.

4) SUPPORT VECTOR REGRESSION (SVR)

Support vector machine (SVM) is an approach that is used for classification and regression problems. SVM has become an exciting model among machine learning techniques due to this model's ability in different issues such as text or image analysis. For instance, the authors in [14] studied SVM for supervised learning methods. However, the first objective of SVM was classification. Nonetheless, this model has been extended to regression problems after a while, called support vector regression (SVR). SVR has the same procedure as SVM, with some differences. This model's objective is to find the most appropriate hyperplane with minimum acceptable

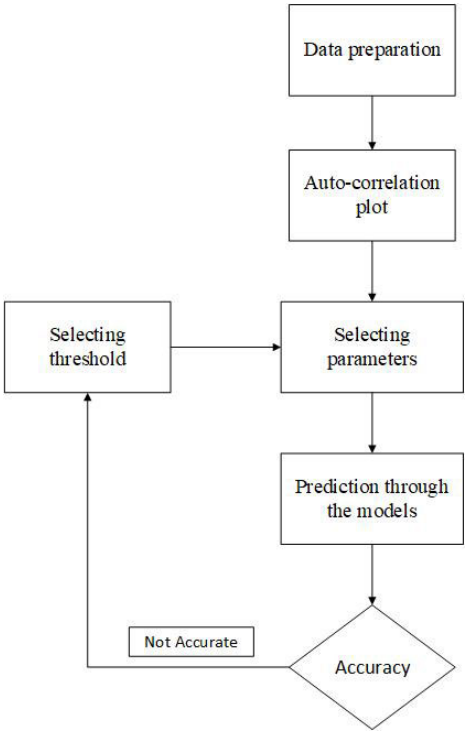

FIGURE 6. The diagram of preparing load data for linear regression, SVR and fully connected models.

error from training samples. In other words, the best fit hyperplane has the maximum number of data points. The main objective is to minimize the coefficients through L2-norm while it is entirely in contrast with the LSE function in linear regression. As it can be seen in figure 7, there is a decision boundary (two red lines) which have ϵ distance with a hyperplane. The accuracy of the model depends on ϵ, so adjusting ϵ, the desired accuracy. Assuming equation (6) indicates hyperplane (in this case, it is a linear equation).

$$y_i = wx_i + b \tag{6}$$

Therefore, the solutions and constraints are as equations (7)-(9):

Solution:

$$min\frac{1}{2}||w||^2 \tag{7}$$

Constraints:

$$y_i - wx_i - b \leq \epsilon \tag{8}$$

$$wx_i + b - y_i \leq \epsilon \tag{9}$$

where x is input, y is target and w is the weight.

SVR also can be used for load forecasting problems. Authors in [32] proposed a new SVR for short-term load forecasting. They evaluated their model using two data sets, ISO New England and North-American Utility. They forecasted 24-hour and 1-hour ahead and achieved reasonable MAPE between 0.75% and 2.25% for test and validation sets. In another study [33], authors applied SVR on electricity load demand recorded every half an hour from 1997 to 1998.

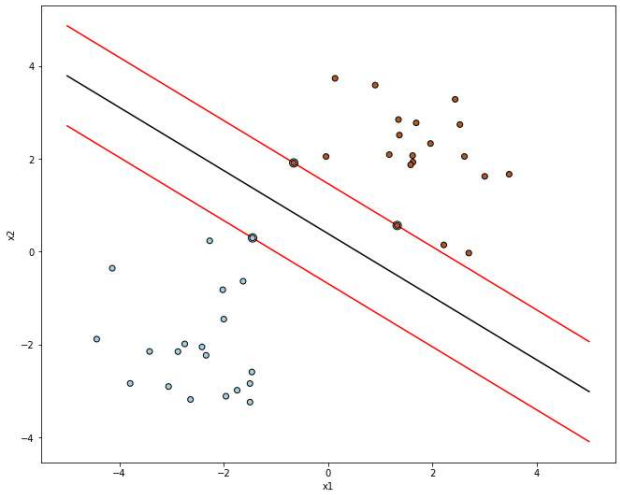

FIGURE 7. An illustrative example SVR. Red line shows the boundary lines, black line shows hyper plane.

They evaluated their model using an exogenous variable (temperature) and without it. They trained the model once with winter data and then with Jan-Feb data. MAPE for different times and variables have been between 1.95% and 3.5%. However, they concluded that it is better to predict future load consumption without using temperature data because it is difficult to predict future temperature, leading to a higher error.

5) FULLY CONNECTED NEURAL NETWORKS

Nowadays, numerous neural networks such as fully connected [34] ones have been introduced. However, it is difficult to train a fully connected network for load forecasting due to the overfitting problem. Therefore, the same approach with discussed linear regression predicts future load consumption through fully connected neural networks.

In neural networks, there are three different layers: the input layer, hidden layer, and output layer. The depth of the network depends on the number of layers in the hidden layer. In fully connected neural networks, all the neurons in each layer are connected to the next layer's neurons. In other words, every output of layers uses as input for the next layer while each neuron has an activation function (usually a non-linear function). Figure 8 shows a simple network with just one hidden layer. Each neuron has a specific weight, and for every layer, a bias term is considered. In total, outputs of layers are computed as:

$$a_l = W_l h_{l-1} + b_l \tag{10}$$

$$h_l = f(a_l) \tag{11}$$

where l indicates layer number, f is activation function like Relu, Softmax, linear, sigmoid. W_l is weighted matrix of layer l, h_l is output of activation function and b_l is bias term of layer l. It is obvious if W is $r \times 1$ matrix, a and h will have $r \times 1$ dimension. Therefore, the transpose of W should be used. If $P(\alpha)$ is considered as predicted output from neural

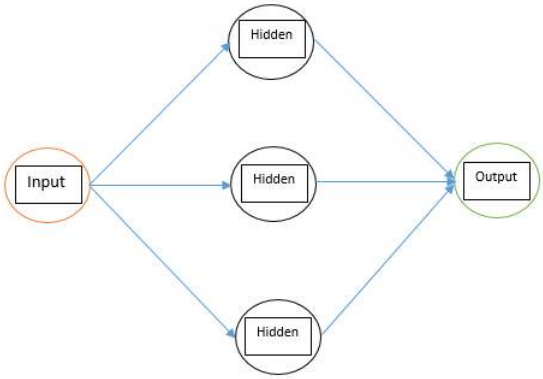

FIGURE 8. A one-layer neural network example.

networks, α represents parameters of neural network and y is the actual value, for N input-output, the loss function is:

$$L = \frac{1}{N}\Sigma_{i=1}^{N}(y_i - P(\alpha))^2 \tag{12}$$

The primary goal is to optimize the parameters of the neural network. For the same purpose, the loss function must be minimized as much as possible. A regularization penalty term is usually used to avoid overfitting (see equation (13)). Overfitting refers to the production of analysis from a statistical model that is extremely trained. The problem is that when overfitting happens, the model learns very well the parameters but it is not able to predict well (i.e. weak generalization ability) [35].

$$L = \frac{1}{N}\Sigma_{i=1}^{N}(y_i - P(\alpha))^2 + \Omega(\alpha) \tag{13}$$

where $\Omega(\alpha) = \lambda||\alpha||^2$, using norm-2 and a hyperparameter to control the regularization strength. For the learning processing there are different algorithms such as RMSprop [36], SGD, ADAM [37]. However, due to its ability to work with non-stationary data, ADAM is the most appropriate choice for load forecasting.

6) LONG SHORT-TERM Memory(LSTM)

Long short-term memory (LSTM) [38] is a particular case of RNNs. RNNs are based on control theory, and because of this reason, they are used to process a sequence of inputs [39]. However, experimental results have proven that RNNs cannot perform well if a long time interval is used as input due to the gradient vanishing problem. To overcome this disadvantage, in many recent pieces of research, RNNs were replaced by LSTM. In load forecasting, many studies used LSTM and improved their approaches by finding the dependency within load series data. The authors in [40] used the LSTM network to carry out load forecasting in different time horizons, including 24 hours, 48 hours, 7 days, and 30 days and compared LSTM with some traditional models such as SARIMA and ARMA. The authors in [41] also studied STLF by using an architecture including LSTM and fully connected layers. Moreover, they used historically as well as prediction data as input for their model. In addition to these particular research efforts of using LSTM for load forecasting, LSTM has been widely used in various hybrid models such as the one in [16].

LSTM consists of 3 different gates, namely input gate, forget gate, output gate. The input gate determines if cell C_t at time t should be updated by the input X_t or not, forget gate determines if the state of cell C_{t-1} should be forgotten, and the output gate controls the output of $h[t]$ to determine which part of cell C_t should be used. The following equations show the computation of LSTM in details:

$$i[t] = \psi(W_i * h[t-1] + b_i) \tag{14}$$

$$f[t] = \psi(W_f * h[t-1] + b_f) \tag{15}$$

$$O[t] = \psi(W_o * h[t-1] + b_o) \tag{16}$$

$$C[t] = f[t] \odot C[t-1] + i[t] \odot (\phi(W_c * h[t-1] + b_c)) \tag{17}$$

$$h[t] = \phi(C[t]) \odot O[t] \tag{18}$$

where W_i, W_f, W_o are parameters to be learned, b_i, b_f, b_o, b_c are biased vectors, ϕ is hyperbolic tangent (or can be any non-linear function), ψ is sigmoid activation function, $f[t]$ is forget gate, $i[t]$ is input gate, $O[t]$ is output gate, $C[t]$ is the state of this cell to encode information from the input sequence, $h[t]$ is network output and all of $[t]$ symbol refers at time t and finally, $\odot$ is used as a symbol for Hadamard product.

7) CONVOLUTIONAL NEURAL NETWORKS (CNN)

CNNs are a big family of artificial neural networks [42] designed to filter and extract input data features. They have been widely used in various areas thanks to their ability to handle data with different dimensionalities. For instance, two dimensional and three-dimensional CNNs are recognized as a powerful network for performing image processing, and classification [43], as well as computer vision tasks [44]. Moreover, in recent years they have been deployed in different other fields including Natural Language Processing (NLP) [10], audio recognition [45], medical [46] and load forecasting [13]. Existing diversities among the load profiles using CNN networks may come up with some difficulty. The complexity of human behaviors, working days, time, and weather data affect the load profiles [47] directly. To overcome the complexity of load profiles, CNNs need to have huge input data as the training set to learn all parameters.

From a technical point of view, CNNs are based on a discrete mathematics operator called convolution, as shown in equation 19. In equation 19, Y is used as an output and x is the input. In addition, w represents the kernel. The i-th output is given as follows:

$$Y(i) = \Sigma_{i,j}x(i-j)w(j) \tag{19}$$

where j is ranging from 0 to $k-1$ and then it makes Y to have $n-k+1$ dimensions, and n is the input's dimension.

Even though convolution operation is a simple mathematical formula, CNNs work a little differently. Figure 9 shows

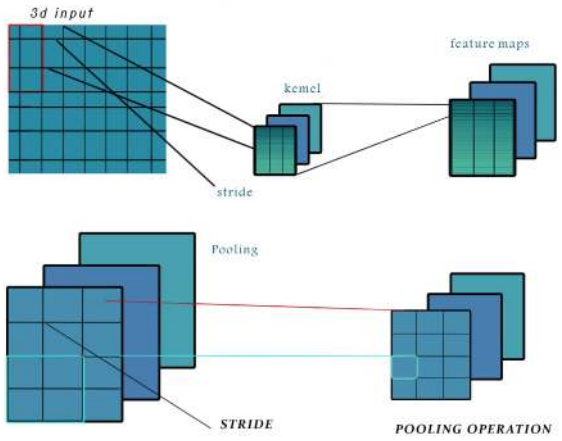

FIGURE 9. 2D-Convolution and Maxpooling operations. In this instance figure the filter size is [3,2], and for the sliding part the size is chosen [2,2].

the inner structure of this neural networks family, and as it can be seen in this figure, convolution filter slides over the whole input data to extract the features. According to [51], in convolution operation, firstly kernel and filter are convolved, and the result of this operation is added to a bias term. This mathematical operation is finished when a complete feature map is achieved. Equations (20) and (21) show the complete convolution operation in artificial networks:

$$Y_{ij}^{m} = sum(k_m \circledast xf_{ij}) + b_m \tag{20}$$

$$f^{m} = activation(Y^{m}) \tag{21}$$

where Y^m indicates the output, m represents the m-th feature maps, i, j indicate the vertical and horizontal steps of filter respectively, xf_{ij} is the filter matrix, k_m represents the kernel matrix, b_m is the bias term, and finally f^m is the activation function's output. It must be said that equation 20 shows the convolution operation formula while equation 21 shows the activation function for the m-th output.

In terms of the CNNs architecture, there are usually convolutional layers, pooling layers, and fully-connected layers. Pooling layers are used after CNN layers to carry out a downsampling operation while keeping the input data quality. This dimension-reduction operation is useful because it makes the model prepared to learn the parameters through the back-propagation algorithm. Finally, a fully connected layer is used to perform the final prediction by combining all features. However, according to the nature of load data, this article focuses on one-dimensional CNN.

8) PARALLEL LSTM-CNN NETWORK (PLCNet) MODEL

This article discusses a new methodology combined with CNN and LSTM, called parallel LSTM-CNN Network (PLCNet), to carry out load prediction. Despite other efforts, such as those reviewed in the introduction that combined both approaches, the methodology presented here is completely different. For instance, the authors in [48] proposed a CNN-LSTM model so that CNN is first used to extract the features of input data, and then output from CNN is used as LSTM input. The problem within this model is that extracted features affect the training of LSTM. In order to solve this problem, in the PLCNet, LSTM and CNN networks are used in two different paths without any correlation between those two paths. Figure 10 shows the frame-work of the proposed methodology. As shown, input signals are first entered into two paths to be processed by LSTM and CNN paths. These two paths extract the features and the long dependency within data and prepare the input data to make the final prediction. A fully connected path, including dense and dropout layers, has been implemented and finally predicted actual values to compare data to carry out the final prediction.

In the CNN path, capturing the feature of local trend is the main objective. In this path, the data is convoluted through a Conv-1D layer within 64 units and filter size 2. After the convolution layer, the Maxpooling layer is used to reduce the data's dimensionality by downsampling while keeping its quality. In the next layer, another Conv-1D layer, but within 32 units, is implemented. The data in the final layer continue through flatten layer. All of the units are activated by Rectified Linear Unit (ReLU).

The LSTM path is used to capture the long-term dependency within data, and data go through a flatten layer to start working with the LSTM network. After passing through the flatten layer, input data is ready to be entered as an LSTM layer input. An LSTM layer with 48 units and the activation function is ReLU.

After passing through LSTM and CNN paths, the processed data is ready to be entered into the fully connected layer. As it was mentioned before, there is no correlation between the two paths. Thus to prepare data for prediction, the outputs are concatenated in a merge layer. The merged data are entered into an LSTM layer with 300 units and ReLU activation function to learn the long-dependency within output data from two paths, and then the output of the LSTM layer will feed the next dense layer. After that, a dropout (30%) [49] layer is implemented to avoid any overfitting. Two other dense layers are used to prepare the data for final prediction. Since this model aims to predict two data sets and various time horizons, the number of units in each dense layer is different. However, all the existing units in the fully connected path are activated by the sigmoid function. Concerning the fact that the PLCNet model also must be evaluated for different time horizons, the number of units in the LSTM-Dense path can be different.

Figure 11 shows a diagram that discusses how the PLCNet model is working and how the input data are being processed. According to the diagram, the number of data in each batch can be different, and it depends on the purpose of the prediction. In other words, for different time horizons, the number of look back steps is different. After choosing the look back number, the load data are batched with the same size. For example, if the number of look back steps is chosen 24, the first batch will contain data point 0 to 23, then the second batch will have data point 1 to 24, the third batch includes

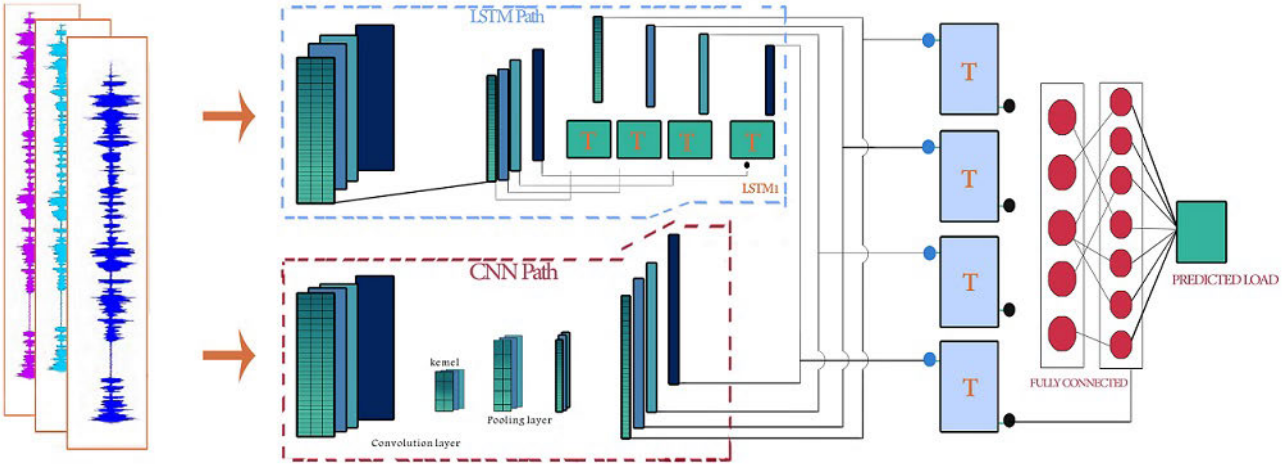

FIGURE 10. The framework of the PLCNet.

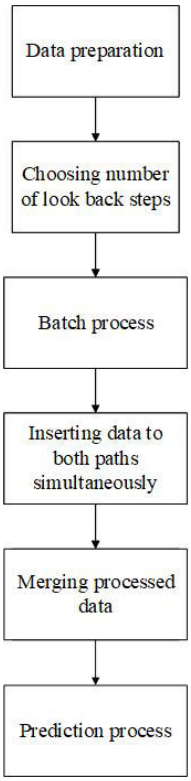

FIGURE 11. The workflow of the PLCNet model.

2 to 25, and so on. Likewise, due to the time horizon of the prediction, the target data can be different.

III. EXPERIMENTAL RESULTS

Malaysian data is divided into two sets, the training set, which contains the year 2009 load consumption, and the year 2010 load consumption used as the test set. German data set is also divided, so that 2012-2015 data are used as the training set, and 2016-2017 ones are used as the test set. All the models are implemented in Python. This article used Keras library with the back end of TensorFlow to implement deep neural networks (DNN). Besides, Scikit-learn, Statsmodels, and Pmdarima libraries were used for regression and time-series modeling and analysis.

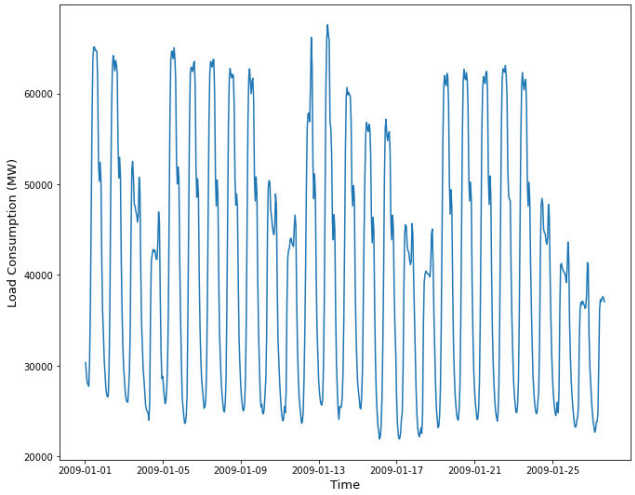

FIGURE 12. The illustration of Malaysian data.

A. CASE STUDIES

Two different data sets are used to carry out STLF. The authors in [25] used load consumption of the city of Johor in Malaysia to predict day-ahead load consumption (hourly prediction) using a model that combines neural network and fuzzy time series. They used a new model, which was a combination of Fuzzy time-series and CNN (FTS-CNN). They first created a sparse matrix through fuzzy logic and then, through CNN, extracted features and carried out STLF. They also tried other models, including SARIMA, different LSTM models, different probabilistic weighted fuzzy time series, and weighted fuzzy time series. Their proposed model (FTS-CNN) could achieve better results than other models for two different years of Malaysia data, and RMSE was 1777.99, 1702.70, respectively. This data is from a power company in this city for the years 2009 and 2010 and consists of hourly electric consumption in MW. It has 17518 rows, which show the aggregated load consumption of these two years in this city. Figure 12 illustrates part of this hourly data, and figure 13 shows a Boxplot of the whole data set how the loads are distributed among days of a week.

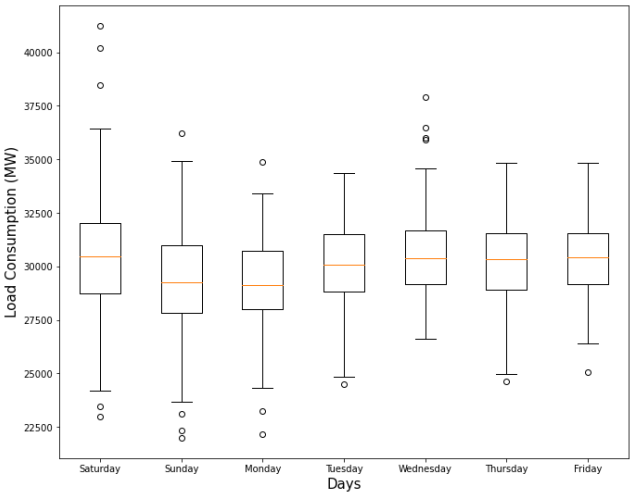

FIGURE 13. Boxplot of load consumption during a week in Malaysia 2009-2010.

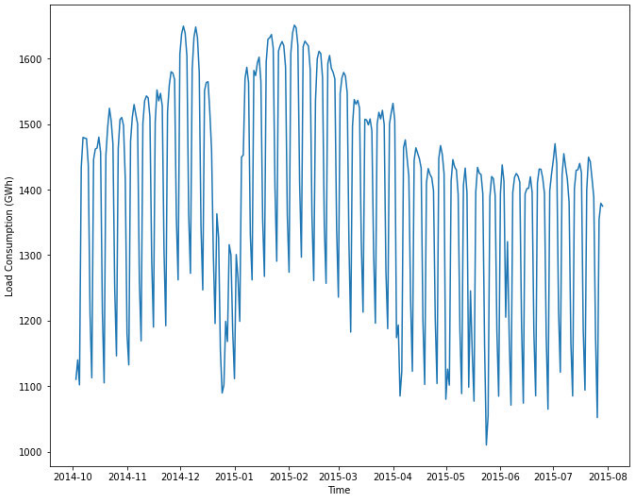

FIGURE 14. The illustration of German load data.

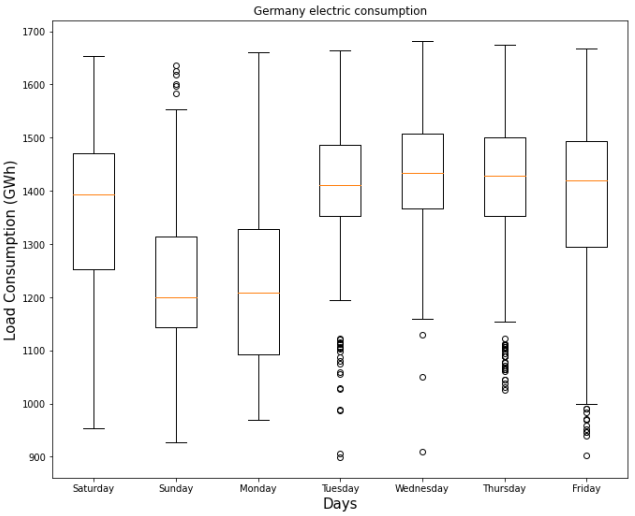

FIGURE 15. Boxplot of electric consumption in Germany 2012-2017.

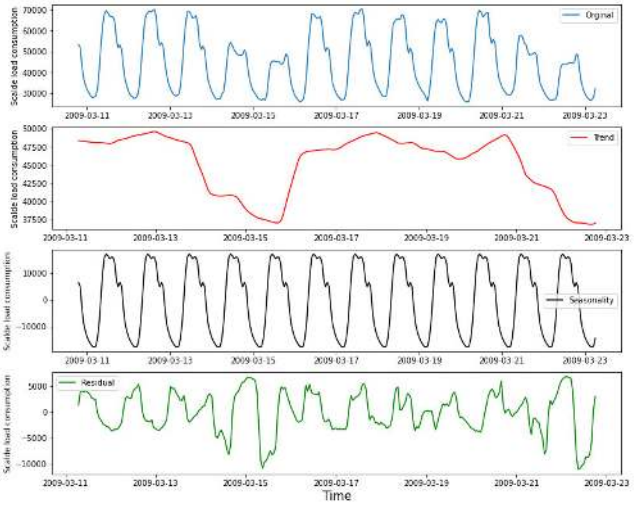

FIGURE 16. Part of the decomposition of Malaysian data.

Another data is Germany country-wide daily aggregated electric consumption since 2006 to 2017 in GWh. This data is provided by Open Power System Data (OPSD) and is used to predict day ahead load consumption. This data has 2186 recorded electric consumption in Germany. Figure 14 shows part of the German load data for almost 9 months and figure 15 shows the Boxplot of this data during a week.

Part of Malaysian data and German data have been decomposed into seasonal, trend and noise. Figures 16 and 17 show the original data and their decomposition. Black plots in both figures show the seasonal part of each data.

B. DATA NORMALIZATION

The acquired results from practical experiments proved that to work with deep learning models, data should be prepared well [50], and results showed pre-processing is more significant than the training process. As discussed before, in load forecasting, even though some parameters such as holidays, temperature, humidity, etc., affect the model, the article 's goal is to carry out STLF using just previous load consumption data. Therefore, data must be prepared specifically for each model. Data are scaled between 0 and 1 through equation (22) which is written as follows:

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{22}$$

C. EVALUATION METRICS

In order to evaluate models performance, root mean squared error (RMSE), mean absolute percentage error (MAPE) and coefficient of determination (R^2) are used.

$$RMSE = \sqrt{(\frac{1}{N})\sum_{i=1}^{N}(A_i - F_i)^2} \tag{23}$$

$$MAPE = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{A_i - F_i}{A_i}\right| \tag{24}$$

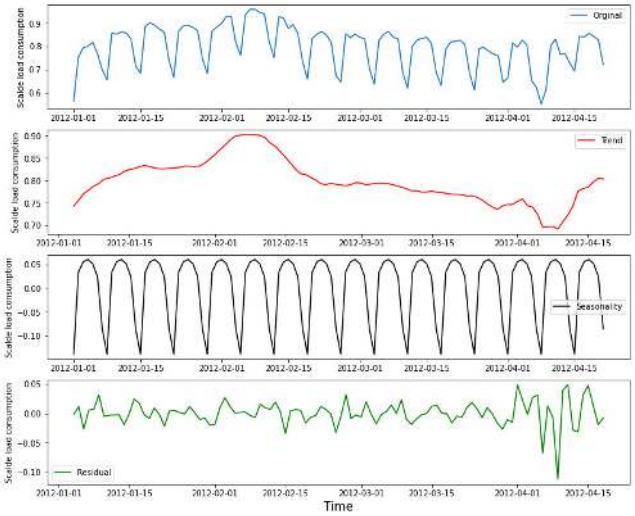

FIGURE 17. Part of the decomposition of German data.

$$R^2 = 1 - \frac{SSR}{SST} \tag{25}$$

$$SSR = \sum_{i=1}^{N} (A_i - F_i)^2 \tag{26}$$

$$SST = \sum_{i=1}^{N} (A_i - \bar{A}_i)^2 \tag{27}$$

where A_i and F_i refer to actual and predicted value of i-th data, N is the data size, $\bar{A}$ is the average of actual data. In addition, SSR stands for sum squared regression and SST stands for total sum of squared.

D. IMPLEMENTATION

In this section, the results of all the discussed models are presented.

1) THE EVALUATION OF ARIMA

In ARIMA models, time-series data decompose into m series to eliminate hourly/daily seasonality within data. According to figures 10, 11 there is a daily seasonality in Malaysian data (every 24 hours), and weekly seasonality in German data (every 7 days). Therefore, instead of using simple ARIMA, seasonal ARIMA (SARIMA) is being used to carry out STLF. In order to find the parameters of SARIMA, Auto-arima function from pmdarima library in python was used and it tries to find the best number for parameters by carrying out a comparison among different parameters. For German data, ARIMA (5,1,0)(5,0,5,[7]) became the final models and ARIMA (1,0,1)(2,0,0,[24]) achieved best results for Malaysian data. Figures 18 and 19 illustrate predicted results for both data from ARIMA model.

As can be seen, ARIMA carried out short-term load forecasting for both daily and hourly data well. However, the problem is that for tuning the best parameters for ARIMA model, some complex computations must be solved. It leads to a considerable amount of RAM involvement in addition to the fact that it takes time.

FIGURE 18. Actual and predicted results from SARIMA, Malaysian data.

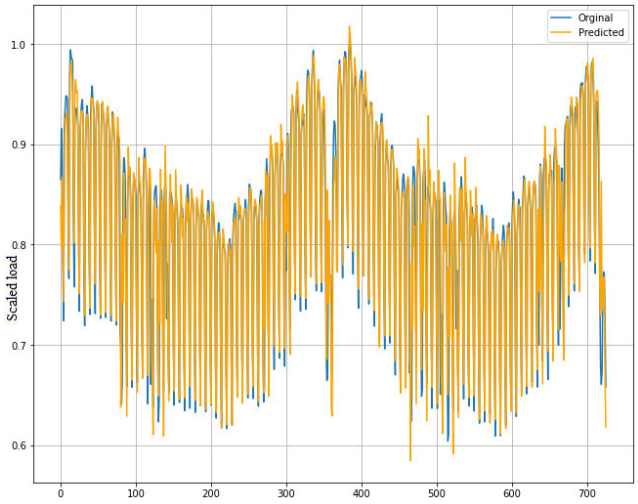

FIGURE 19. Actual and predicted results from SARIMA, German data.

2) THE EVALUATION OF EXPONENTIAL SMOOTHING:

Exponential Smoothing (ETS) is an alternative approach for load forecasting. Training and test sets of two data sets are applied to this model to carry out $t + 1$ forecasting. Besides, figures 20 and 21 show predicted and actual test set for both data. These plots prove that ETS fails to perform accurately in STLF.

3) THE EVALUATION OF LINEAR REGRESSION

For the linear regression model, the ACF plot is used to find out how many lags can be used as linear regression variables (independent data). In figure 22, ACF plot of Malaysia is illustrated. For this data set, the threshold is 0.75. Lags [1, 2, 23, 24, 25, 47, 48, 49, 71, 72] become the independent data and actual load consumption are used as targets (dependent data). Scaled data is divided into training and test set. As 10 lags are used as variables, the shape of train set is (8723,10) which started from the first day of 2009 to the first

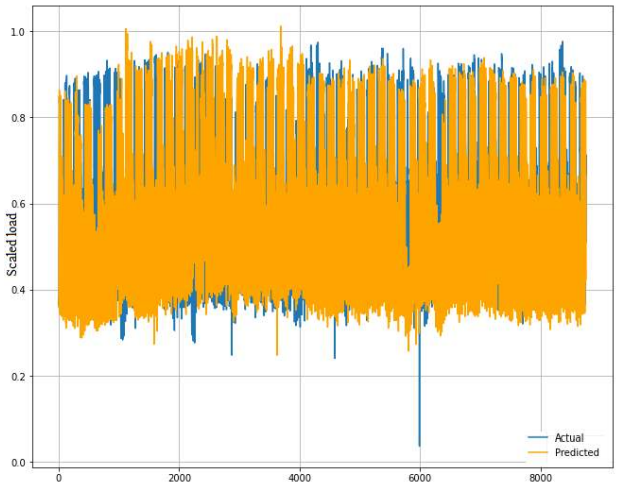

FIGURE 20. Actual and predicted results from ETS, Malaysian data.

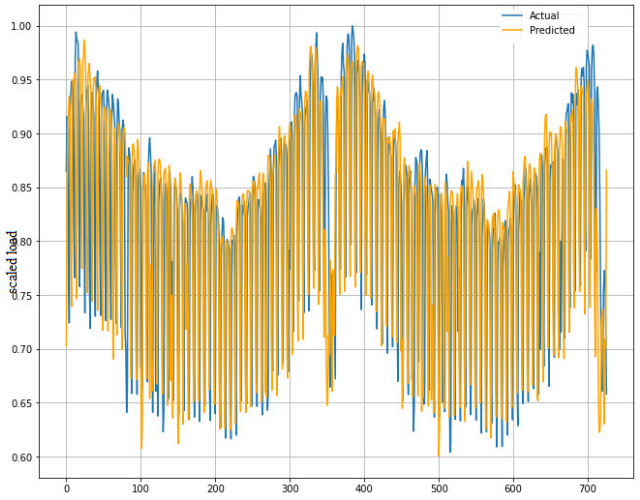

FIGURE 21. Actual and predicted results from ETS, German data.

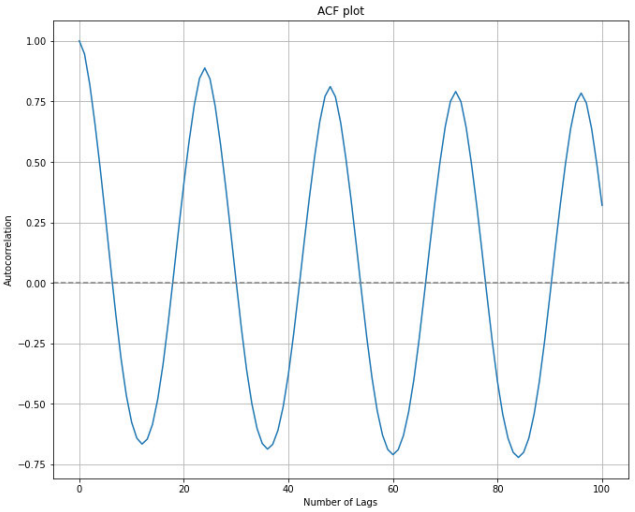

FIGURE 22. AC plot of Malaysian data.

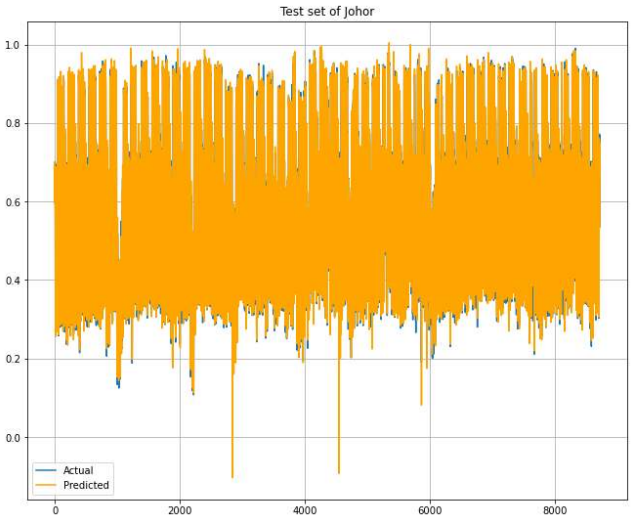

FIGURE 23. Actual and predicted results from linear regression, Malaysian data.

day of 2010 and test set has the shape of (8723,10) from first day of 2010 to the end of this year.

Figure 23 shows predicted and actual data of load consumption for year 2010 in Malaysia. As can be seen, linear regression achieved accurate results for this data set.

However, for the German data set, there are some differences. The first difference is that 0.69 is chosen for the threshold. Figure 24 shows AC plot of German data. According to this plot and threshold, lags [7, 14, 21, 28] are being used as independent variables for MLR. Historical data from 2012 to the end of 2015 are used as the training set. The shape of training data is (1456,4), and the test set is from 2016 to the end of 2017 with a shape of (702,4). As this data is daily, the model predicted daily load consumption but as not good as predicted results from Malaysian data. Figure 25 shows actual and predicted results from test set. According to these figures, while linear regression can predict hourly load series accurately, it fails to forecast accurately future load consumption of daily load series. The difference in the number of lags as variables explains why the results are not similar. The number of variables (lags) for Malaysian data is 10, while it is 4 for German data. This point is the main weakness of linear regression. Even though this model is high-speed, it fails to achieve accurate results if there is not much auto-correlation in input data.

4) THE EVALUATION OF SVR

As SVR is a regression-based approach, the same training and test sets for linear regression in the previous section are used to evaluate the model. Various parameters affect SVR to perform well. Among all these parameters, choosing the appropriate kernel has the most importance. For Malaysian data, the 'linear' kernel had the best performance compared to other kernels, and the German case selects 'radial bias function (rbf)' as kernel according to its well-performance with this data set. In addition, figures 26 and 27 show the predicted load consumption from SVR for both data sets, respectively.

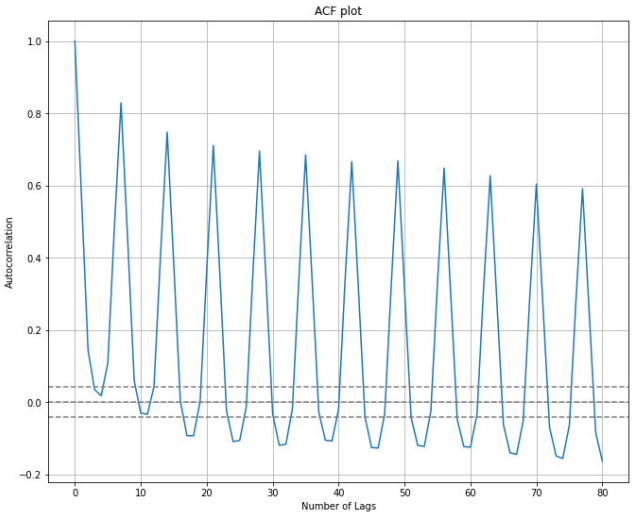

FIGURE 24. AC plot of German data.

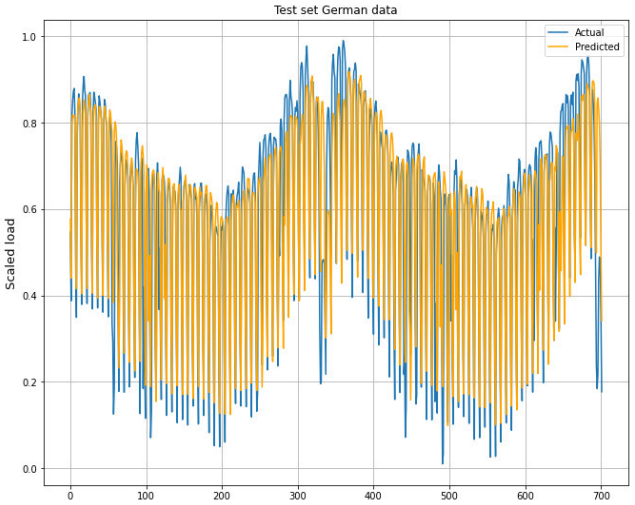

FIGURE 25. Actual and predicted results from linear regression, German data.

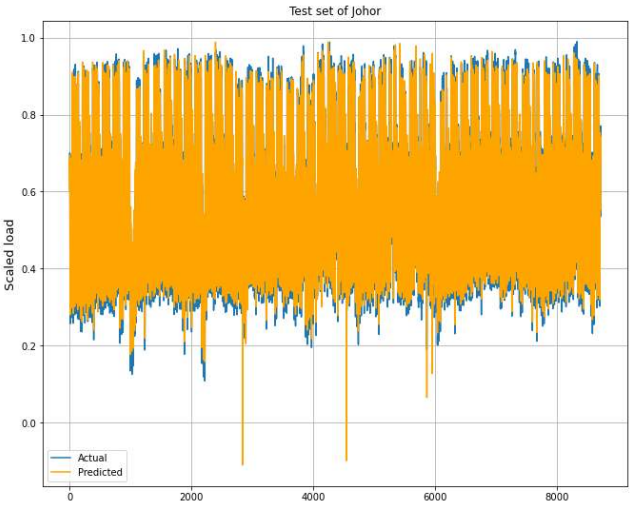

FIGURE 26. Actual and predicted results from SVR, Malaysian data.

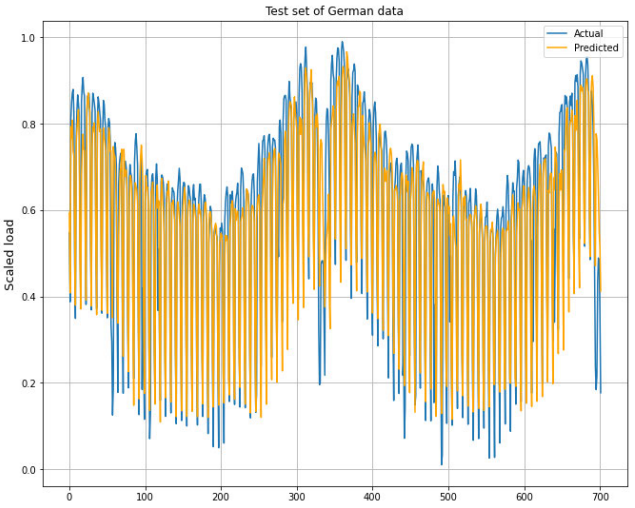

FIGURE 27. Actual and predicted results from SVR, German data.

As seen from the figures and tables, SVR predicted future load consumption of Malaysia with less accuracy than linear regression. However, SVR has achieved more accurate results than linear regression for German data. SVR has better results than linear regression in German data because only 4 lags are considered independent and are not enough for linear regression. Nevertheless, it can be concluded that SVR is not the right candidate for STLF.

5) THE EVALUATION OF FULLY CONNECTED NEURAL NETWORKS

A fully connected neural network has been used for the same training and testing sets used in two previous sections. This network has 3 hidden layers in addition to input and output layers. The first layer is the input layer, and hidden layers consist of 27, 18, and 18 dense layers, respectively. To avoid overfitting, before the output layer, a dropout (20%) layer is used. As this network is supposed to forecast load consumption, one dense is used in the output layer. This model learns the parameters through ADAM optimizer in 20 epochs, and the size of each batch is 1. Besides, for whole layers, ReLU is used as the activation function. ReLU stands for Rectified Linear Unit, and it works like a linear function with a difference, which is its output for negative inputs is zero. This attribute helps DNN models to avoid vanishing gradient problem. The mathematical formula is given below, while figure 28 shows this function:

$$F(x) = max(0, x) \tag{28}$$

As it could be assumed, the results from fully connected neural network (see figures 29 and 30) with ReLU would be close to results from linear regression while this model is more complex and needs more time to forecast future load consumption. Especially, when training and test sets are the same for both models.

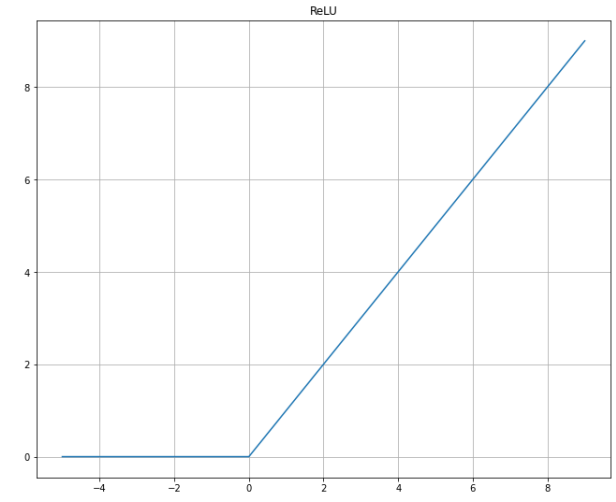

FIGURE 28. ReLU illustrative function.

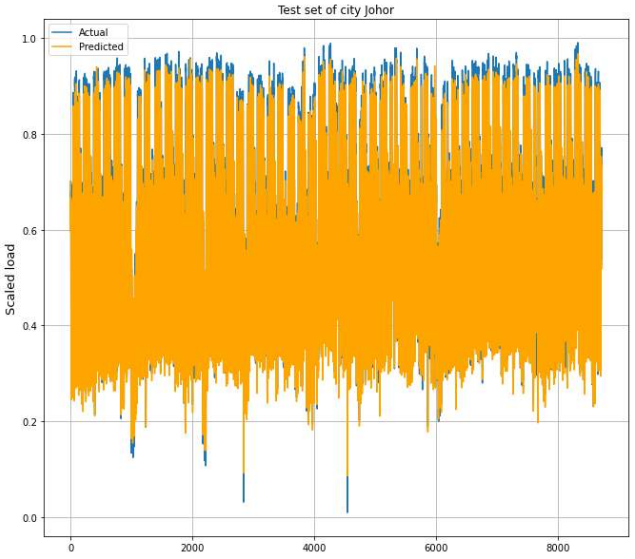

FIGURE 29. Actual and predicted results from fully connected, Malaysian data.

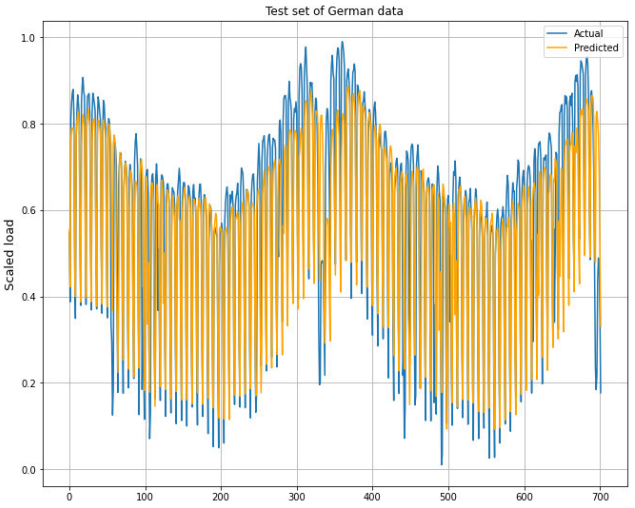

FIGURE 30. Actual and predicted results from fully connected, German data.

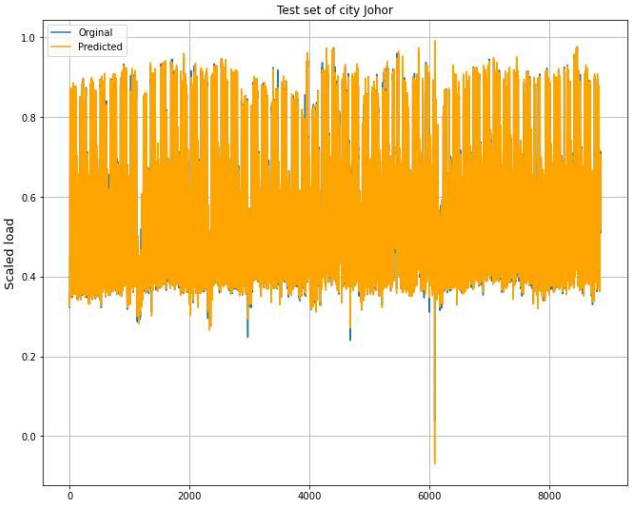

FIGURE 31. Actual and predicted load consumption from LSTM, Malaysian data.

6) THE EVALUATION OF LSTM

In this article, the LSTM model studies the last 24 hours load consumption and predict the next hour consumption in Malaysian data. In contrast, in German data, in order to predict next-day load consumption, it studies the last 7 days and predicts next day data. It has one LSTM layer in terms of architecture, while one dense layer is used as output. This type of architecture calls Vanilla LSTM, a well-known network and widely used in different areas. Same with the fully connected network in the previous section, the used activation function is ReLU. The model is also trained by ADAM optimizer for Malaysian data in 200 epochs and RMSprop for German data in 150 epochs. This model proves that LSTMs are a powerful tool for STLF due to the accurate results that the LSTM model has achieved as well as their independence to auto-correlation of input data. The results are illustrated in figures 31 and 32.

7) THE EVALUATION OF CNN-LSTM AND PLCNet

As discussed before, CNNs are well-known networks for feature extraction. LSTM also showed its ability to predict short-term load consumption. Therefore, a hybrid model of CNN and LSTM can come with several advantages. In order to work with this hybrid model, CNN layers should be implemented first to apply historical data. In the next step, extracted features from CNNs are used as input for LSTM layers. This section uses a 7-layer model to apply on the same test and training set used for the LSTM model in the previous section. In layer #1 and layer #2, 1-D CNNs with the ReLU activation function and filters = 64 and kernel size = 3 are implemented. After that, a Maxpooling and Flatten layer are used to prepare data for the LSTM layer. In layer #5, 200 LSTM neurons with ReLU activation function are implemented. Two dense layers with 200 and 1 neurons are implemented to analyze the results and predict load consumption, while ReLU is used as

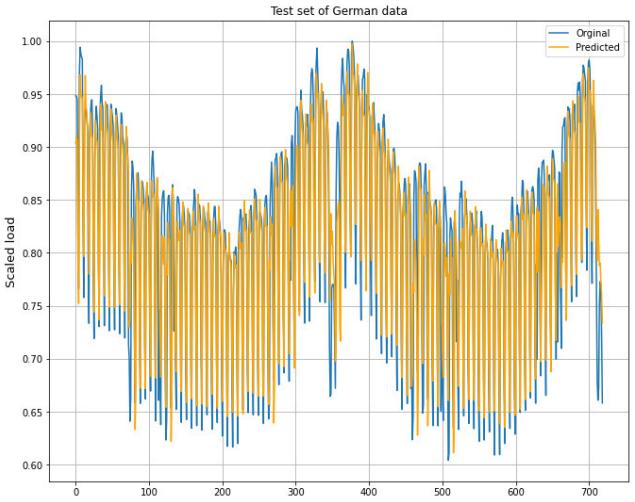

FIGURE 32. Actual and predicted load consumption from LSTM, German data.

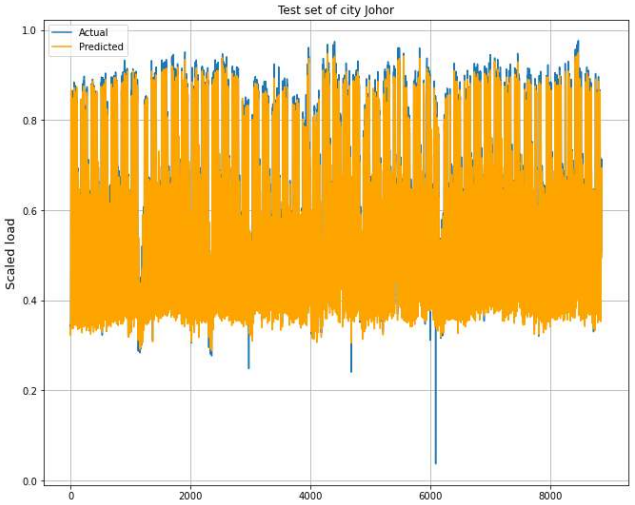

FIGURE 33. Actual and predicted results from CNN-LSTM, Malaysian data.

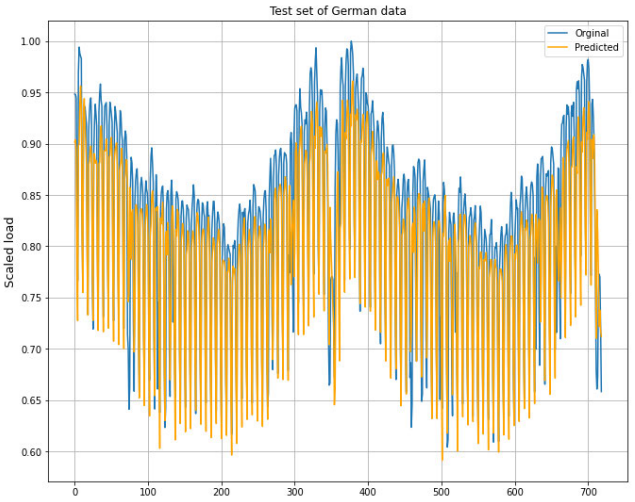

FIGURE 34. Actual and predicted results from CNN-LSTM, German data.

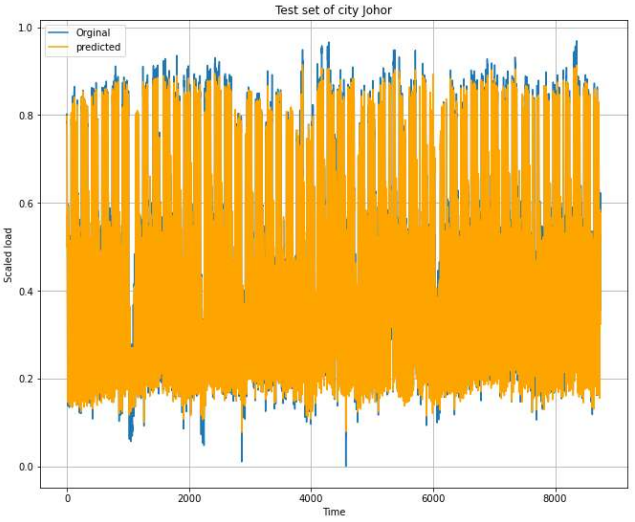

FIGURE 35. Actual and predicted results from PLCNet, Malaysian data.

the activation function. Like other DNN models, the ADAM optimizer has the role in compiling the model for Malaysian data, and the RMSprop optimizer is used for German data. Figures 33, 34 the results of forecasted data with CNN-LSTM model.

As it mentioned before, the PLCNet includes two different parallel paths, CNN path and LSTM path, and these two paths are fed simultaneously by historical load data. According to the figures 35 and 36, the model has a good performance for both data sets.

E. RESULTS

The detailed experimental results are presented numerically in tables 2 and 3. As shown in these two tables, the PLCNet's MAPE and RMSE are the smallest, while the R^2 score is the highest. Regarding the largest error value, the MAPE and RMSE of ETS have the highest error value in both German and Malaysian data sets, where it has got 0.36 and 8.81 for RMSE and MAPE for Malaysian data and 0.316 and 33.63 for RMSE and MAPE for German data. According to the MAPE and RMSE values, the short-term electric load forecasting accuracy of tested models in descending order is as follows: the PLCNet, LSTM-CNN, LSTM, ARIMA, linear regression, DNN, SVR, and ETS.

Besides, it can be seen in the figures that the PLCNet has performed far better than other models, especially in the German data set. Since the Malaysian data is hourly data, many samples are available; thus, all the models can be trained well, while German data is a daily one, so all the models have been trained with fewer samples. It leads to letting PLCNet model shows its power more with an accuracy of 91.18%, in the German case. After that, LSTM has performed well, and its accuracy is 83.17%. Likewise, the accuracy of the PLCNet model for Malaysian data is the highest, too, 98.23%. However, in this case, there is no remarkable difference between the most accurate one and the second one, where LSTM-CNN accuracy is 97.49%.

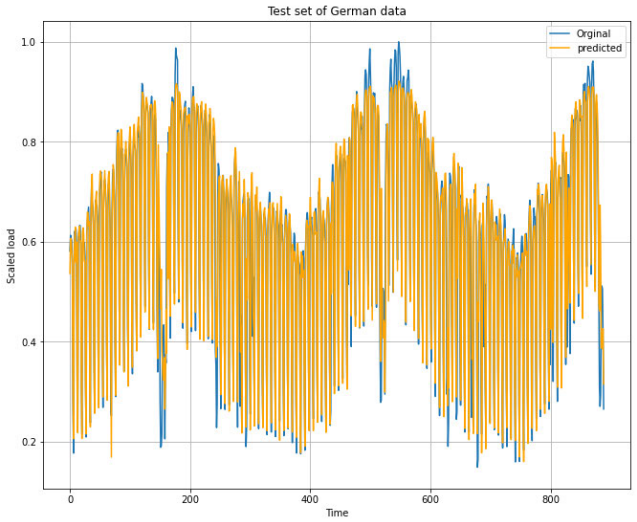

FIGURE 36. Actual and predicted results from PLCNet, German data.

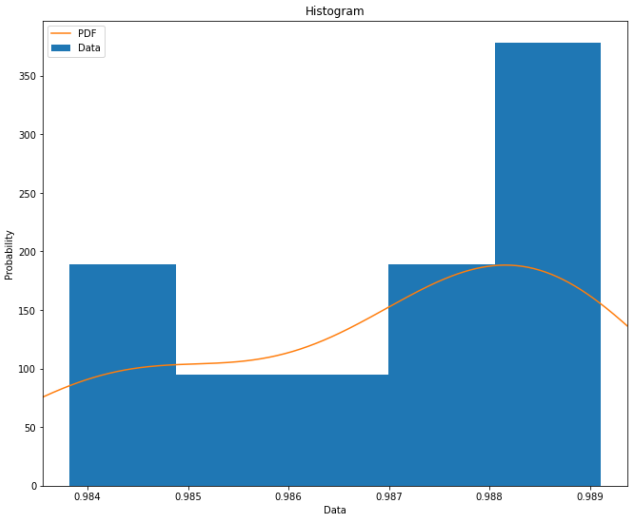

FIGURE 37. Histogram plot of the PLCNet results for Malaysian data.

TABLE 2. Models Performance for Malaysian Data.

Model	Performance metrics		$R^2 score$	Runtime (s)
	RMSE	MAPE		
ARIMA	0.102	3.56	94.19%	451.12
ETS	0.36	8.81	90.06%	380.35
Linear Regression	0.092	2.335	95.50%	12.41
SVR	0.272	7.63	90.40%	10.23
DNN	0.128	3.62	95.38%	199.12
LSTM	0.097	3.11	96.63%	902.56
LSTM-CNN	0.053	2.43	97.49%	487.33
PLCNet	0.031	2.08	98.23%	92.47

TABLE 3. Models Performance for German Data.

Model	Performance metrics		$R^2 score$	Runtime (s)
	RMSE	MAPE		
ARIMA	0.201	18.40	80.04%	179.89
ETS	0.316	33.63	70.1%	167.03
Linear Regression	0.214	19.12	79.86%	4.32
SVR	0.247	22.41	74.39%	3.11
DNN	0.25	26.47	73.47%	80.35
LSTM	0.197	13.20	83.17%	431.11
LSTM-CNN	0.207	15.02	79.75%	180.22
PLCNet	0.061	2.08	91.18%	65.34

TABLE 4. The Training Time Per Epoch of Deep Learning Models.

Model	Runtime per epoch (s)	
	Malaysian data	German data
DNN	9.95	3.2
LSTM	4.61	1.87
LSTM-CNN	9.74	3.01
PLCNet	4.5	0.93

Regarding the run time in the tables, linear regression and SVR are the fastest in German and Malaysian cases. However, they are outperformed by deep learning models. Besides, ARIMA and ETS are two computational techniques that take significant time for training. Even though all deep learning models in the tables took much time to be trained compared to regression-based approaches, their acquired accuracy is acceptable. The difference between the training time in deep learning models depends on the number of epochs considered. Table 4 indicates the runtime per epoch of each deep learning model for both Malaysian and German data sets. According to the tables 2 and 3, LSTM has the highest runtime, but the main reason is that this model needs more epochs to be trained and predict future load. It can be seen that in table 4 LSTM is faster than LSTM-CNN and DNN models in both data sets. However, the PLCNet results show that this model has the highest accuracy and lowest error amount. It is also the fastest model between deep learning models where the runtime per epoch in Malaysian data is 4.5(s) and in German data is 0.93(s).

Therefore, it is proven that the novel hybrid STLF algorithm proposed in this article is practical and useful. Although LSTM has a good performance when dealing with time-series, its accuracy in German data set, which does not have many samples, is not good enough. Therefore, the LSTM is not suitable for this kind of prediction. Finally, the experimental results show that the PLCNet provides the best electricity load forecasting results.

F. STATISTICAL ANALYSIS

A common approach to comparing the performance of the machine learning models is using statistical methods to select the best one. This section aims to compare the PLCNet, and LSTM results since both achieved acceptable results in both German and Malaysian cases. To provide statistical analysis of the PLCNet and LSTM results, these two models were run 10 times. In terms of visualization, figures 37 and 38 show the results histogram of the PLCNet and LSTM for Malaysian data set, respectively.

In this section, the t-test is used to understand the achieved results are just some stochastic results or they are trustful to perform statistical analysis. This analysis works based on the null hypothesis. The null hypothesis is that two models (the PLCNet and LSTM) are similar to each other. There is no difference between them, while the alternative hypothesis is that the two models perform differently. The considered significance level is 5%, so if the acquired P-value is less than

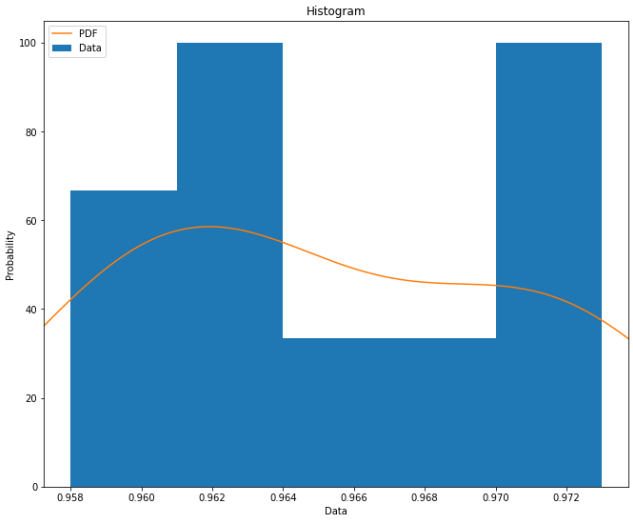

FIGURE 38. Histogram plot of LSTM results for Malaysian data.

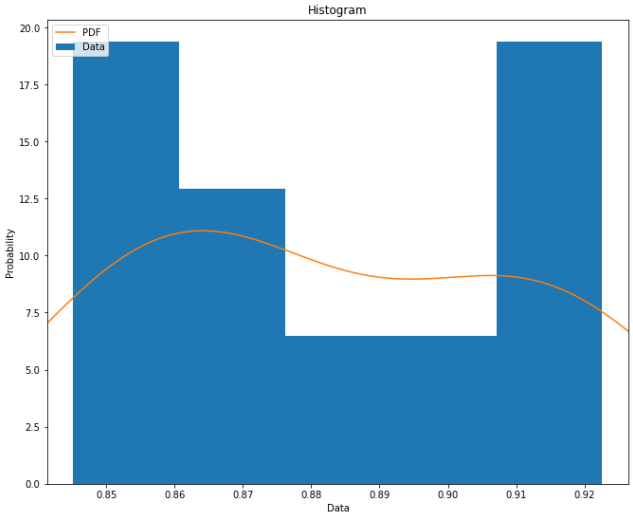

FIGURE 39. Histogram plot of the PLCNet results for German data.

5%, the null hypothesis can be rejected. It can be concluded that the PLCNet performs better than the LSTM model. After carrying out some statistical analysis, the obtained P-value is 0.0411 (or 4.11%), less than 5%. Likewise, after applying the statistical analysis to the German data case, it can be concluded that the results from the PLCNet are not stochastic, where the obtained P-value is 0.03019 (or 3.019%). Figures 38 and 39 illustrate the histogram plot of the results.

G. DIFFERENT TIME HORIZONS

Previous sections discussed some machine learning techniques to predict next time step load data consumption. In other words, at time t, they predicted the load data at time $t + 1$. Since the Malaysian data is hourly data and German data is a daily one, all the models predicted the next hour load of Malaysian data and the next-day German data load. This section aims to challenge the PLCNet in different horizons. In the Malaysian case, the PLCNet will predict the next 24 hours, next 48 hours, and next 10 days load data,

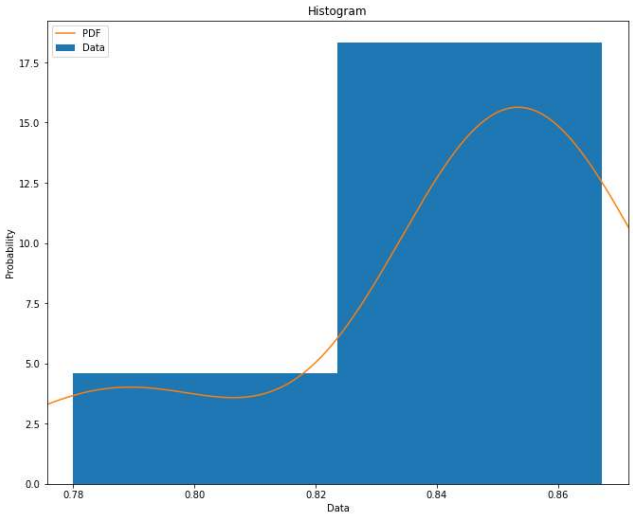

FIGURE 40. Histogram plot of LSTM results for German data.

and in the German case, it will predict next 7 days, next 10 days, and next 30 days. RMSE and R^2 scores are the two metrics used to evaluate the model's performance in terms of evaluation. Because of the existing soft computing errors, the model is tested 5 times for each horizon, and the average value is calculated. Same as mentioned before, the model uses the year 2009 as training set and year 2010 as the test set in Malaysian data, and years 2012-2015 are used as the training set and 2016-2017 as the test set in the German data set.

1) MALAYSIAN CASE

Since in previous sections, the prediction of next hour load data through the PLCNet was discussed thoroughly, this section studies the next one day, 2 days, and 10 days load data in the following.

The Malaysian data is hourly data, so predicting one day ahead load data next 24 time steps should be predicted, leading to a subtle modification in the model's architecture. The last layer of the model, which is a dense layer, will have 24 neurons to provide the next 24 hours prediction. To predict one-day data, the model looks back to 72 hours ago to train the algorithm within data and then predict the next 24 hours. Figure 41 shows the results of the prediction.

In order to forecast the next 2 days load data, the next 48 time steps should be predicted, so another modification is needed to make the model able to forecast the next days' load data. Thus, the model will have 48 neurons in its last layer (a dense layer). In terms of the training procedure, the model looks back to 4 days ago ($4days \times 24h$) to be trained, and then it predicts the next 2 days. Figure 42 illustrates the prediction and actual load data.

This paragraph aims to predict the next 10 days of load data, an MTLF task but a big challenge for the model. Since the samples have been recorded hourly in Malaysian data, the model must predict 240 values ($10\ days \times 24h$). There are 240 neurons in the last layer. The model looks back to 10 days ago and predicts 10 days ahead load data in the training process. According to figure 43, the results show that

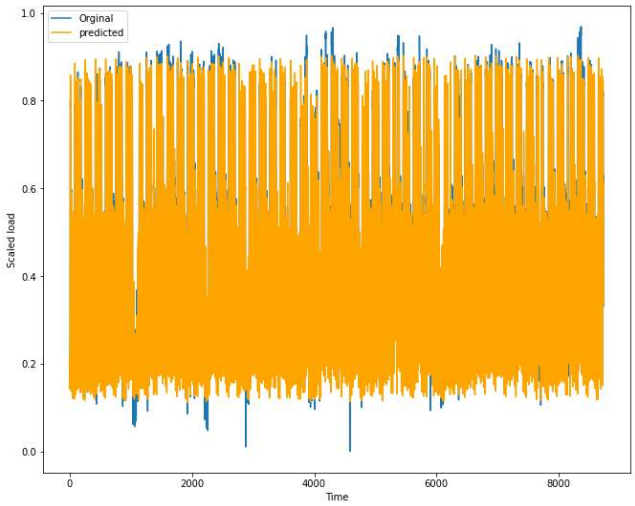

FIGURE 41. 1 day ahead results using the PLCNet, Malaysian data.

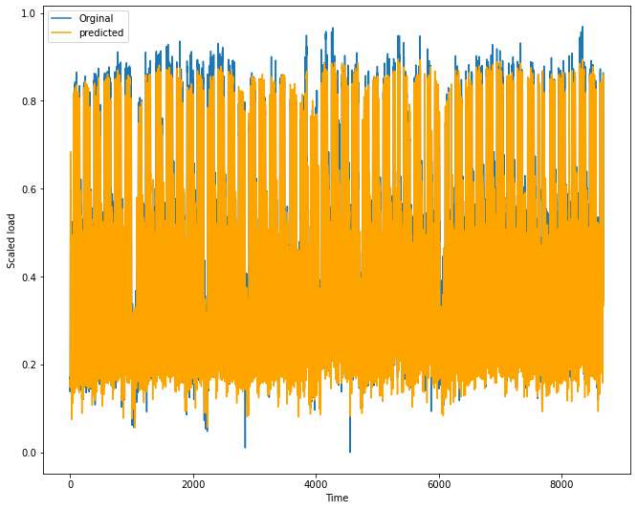

FIGURE 42. 2 days ahead results using the PLCNet, Malaysian data.

the PLCNet has an acceptable performance in this task where the results are close to next hour, one day and 2 days ahead outputs.

2) RESULTS

Tables 5 and 6 show the results of next days prediction. However, to have comprehensive knowledge in terms of the model's performance, the results of the next hour prediction are added to these tables again. In this table, $A1$, $A2$, $A3$ and $A4$ represent next hour, next day, next 2 days and next 10 days results.

Even though forecasting future load data in longer time horizons is a challenging task, according to these tables, there is almost 4% difference between the accuracy of the next hour prediction and the next 10 days prediction which is an acceptable difference and the model has a good performance in Malaysian case.

Likewise, in order to make sure the PLCNet has better performance rather than two other discussed deep learning

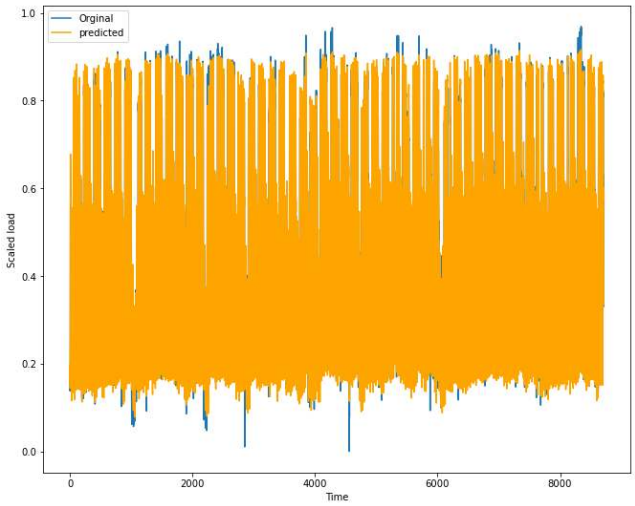

FIGURE 43. 10 days ahead results using the PLCNet, Malaysian data.

TABLE 5. The Experimental Results of Malaysian Data in Terms of R^2 Score.

Model	Test-1	Test-2	Test-3	Test-4	Test-5	Average
A1	98.06%	98.10%	98.12%	98.19%	98.23%	98.14%
A2	97.09%	97.60%	97.56%	97.63%	97.89%	97.55%
A3	96.80%	96.20%	96.31%	97.01%	96.88%	96.64%
A4	94.10%	93.89%	94.25%	94.35%	94.24%	94.16%

TABLE 6. The Experimental Results of Malaysian Data in Terms of RMES.

Model	Test-1	Test-2	Test-3	Test-4	Test-5	Average
A1	0.0341	0.0337	0.0335	0.0328	0.0316	0.0331
A2	0.0412	0.0374	0.0382	0.0376	0.0355	0.0379
A3	0.0410	0.0413	0.0401	0.0387	0.0398	0.0401
A4	0.0609	0.0590	0.0582	0.05476	0.05477	0.0575

TABLE 7. The Comparison Table for Malaysian Data in Terms of R^2 Score.

Model	DeepEnergy	LSTM	CNN-LSTM	PLCNet
A1	94.88%	96.63%	97.49%	98.14%
A2	87.92%	95.21%	96.62%	97.55%
A3	87.86%	92.65%	94.31%	96.64%
A4	90.02%	92.03%	92.88%	94.16%

TABLE 8. The Comparison Table for Malaysian Data in Terms of RMSE.

Model	DeepEnergy	LSTM	CNN-LSTM	PLCNet
A1	0.055	0.097	0.053	0.033
A2	0.0852	0.121	0.069	0.0379
A3	0.0854	0.189	0.0782	0.0401
A4	0.077	0.197	0.082	0.0575

models including DeepEnergy [20], LSTM and CNN-LSTM, tables 7 and 8 are provided to compare the results of all these models in different time horizons in terms of RMSE and R^2 score.

3) GERMAN CASE

Same as Malaysian data, the model is challenged through German data set but in different time horizons. It predicts the next 7 days, next 10 days, and next 30 days load data.

The model looks back to 7 days ago to perform 7 days ahead prediction and to do this task it needs 7 neurons in its last layer. Figure 44 shows the result of using the PLCNet to forecast Germany's next 7 days load data.

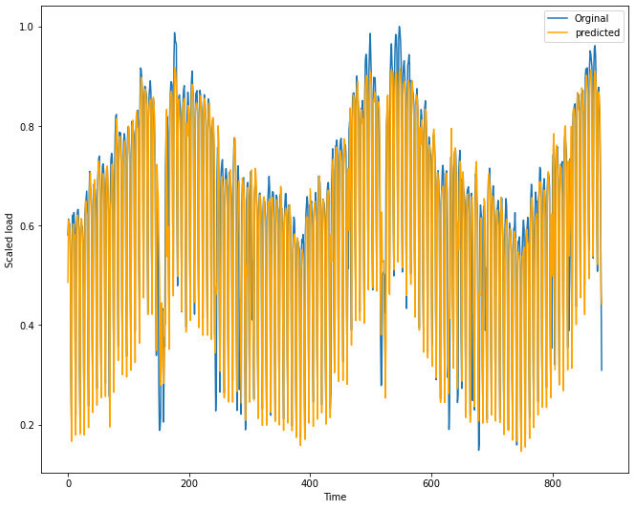

FIGURE 44. 7 days ahead results using the PLCNet, German data.

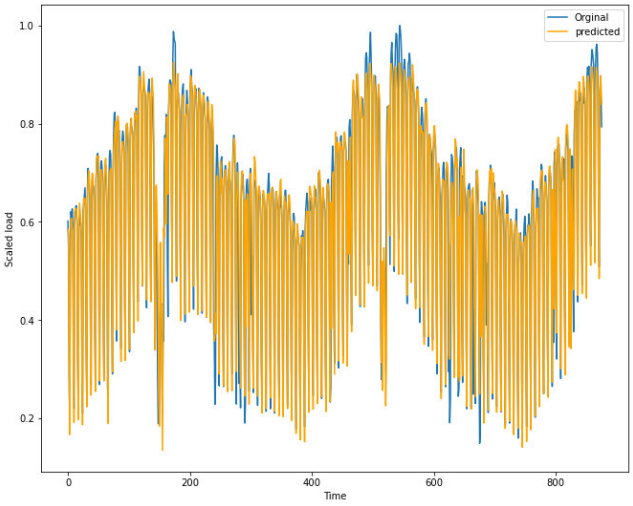

FIGURE 45. 10 days ahead results using the PLCNet model, German data.

This paragraph discusses the results of 10 days ahead prediction. As the model is supposed to predict next 10 days, there are 10 neurons in the last layer of the model. Besides, since forecasting the next days data is a bit harder than next 7 days, the model looks back to 10 days ago data to understand the algorithm within load series better. The results are shown in figure 45.

If the PLCNet can carry out an LTLF task, it can be introduced as a well-performance tool in load forecasting applications. To evaluate the LTLF task model's performance, this section aims to predict the next 30 days of German load data, and the results are shown in figure 46. Like previous sections, there are 30 neurons in the last layer of the model to predict future load data in terms of the model architecture.

4) RESULTS

So far, the German data set's illustrative results have been shown, and in the following, the numerical results are available. Besides, the one day ahead prediction results are

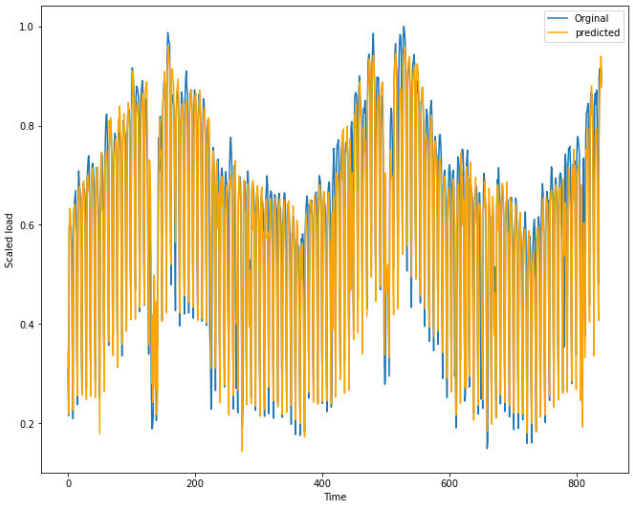

FIGURE 46. 30 days ahead results using the PLCNet, German data.

TABLE 9. The Experimental Results of German Data in Terms of R^2 Score.

Model	Test-1	Test-2	Test-3	Test-4	Test-5	Average
B1	90.76%	92.26%	91.18%	90.47%	91.88%	91.31%
B2	89.75%	89.92%	89.63%	88.82%	89.57%	89.53%
B3	89.02%	89.06%	89.59%	89.26%	88.99%	89.18%
B4	83.05%	83.25%	82.27%	81.84%	82.08%	82.49%

TABLE 10. The Experimental Results of German Data in Terms of RMES.

Model	Test-1	Test-2	Test-3	Test-4	Test-5	Average
B1	0.0659	0.0606	0.0622	0.0686	0.0612	0.0637
B2	0.0741	0.0734	0.0753	0.0787	0.0745	0.0752
B3	0.077	0.0768	0.0748	0.0733	0.0788	0.0761
B4	0.132	0.124	0.155	0.183	0.16	0.117

TABLE 11. The Comparison Table for German Data in Terms of R^2 Score.

Model	DeepEnergy	LSTM	CNN-LSTM	PLCNet
B1	82.79%	79.75%	83.17%	91.31%
B2	80.12%	78.88%	80.87%	89.53%
B3	78.86%	78.02%	79.65%	89.18%
B4	78.02%	74.14%	76.88%	82.49%

available in tables 10 and 9 to demonstrate the comparison between different horizons for same data sets. The modeling's name $B1$, $B2$, $B3$, and $B4$ represent the modeling for 1, 7, 10, and 30 days ahead, respectively. These two tables indicate that there are not many differences between one day ahead prediction and 10 days ahead prediction. Still, it is a big challenge for the model to predict the next 30 days load series because the average accuracy for the next day prediction is 91.31% while it is 82.49% for the next 30 days prediction. The problem is that the German data is not a big data set, and it has only 2186 recorded samples, so it is difficult for the model to learn all the parameters well while looking back 30 previous steps and predicting the next 30 steps.

Same as the Malaysian data, a comparison table (see tables 11 and 12) is provided for German results to prove that the PLCNet not only has a better performance in one step ahead prediction but also it can perform better than other deep learning models in different time horizons.

TABLE 12. The Comparison Table for German Data in Terms of RMSE.

Model	DeepEnergy	LSTM	CNN-LSTM	PLCNet
B1	0.0827	0.207	0.197	0.063
B2	0.102	0.215	0.201	0.0752
B3	0.281	0.231	0.209	0.0761
B4	0.198	0.312	0.279	0.117

IV. CONCLUSION

With smart grids on the rise, the importance of short-term load forecasting highly increases. To predict the future load consumption, some factors such as weather can affect the results. The lack of future weather is a challenging problem for load forecasting. In this article, the previous consumption was used as a parameter to predict the load one step ahead. Some non-deep learning approaches like linear regression or ARIMA have proven powerful tools for accurate load forecasting. However, regression-based methods come with some disadvantages. In order to use these models, such as SVR and linear regression, lags are used as parameters through auto-correlation (AC) values. As the threshold value is subjective, the number of lags as regression-based models' parameters can be different. Fully connected networks also use the same approach as regression-based approaches. Because there is no constant threshold to find those lags that are suitable to be used as variables, this procedure (finding lags through AC plot) may lead to higher errors. ARIMA and ETS also are two well-known time-series analysis approaches. However, some parameters need to be tuned to work with these methods. This procedure needs numerous trials to find the best values for them. Furthermore, in time-series methods, data must be analyzed to find out if they are stationary or not. In contrast, LSTM can achieve good results whether data is stationary or not. CNN-LSTM also is a hybrid model, which is used in various load forecasting studies. The PLCNet achieves the best results between all the discussed models where the accuracy increase from 83.17% to 91.18% for load data in a German case study. Likewise, for a Malaysian data set, the model's obtained accuracy is 98.23% which is very high for time-series results and the RMSE is very low at 0.031. In summary, the PLCNet improves the results remarkably for the German data set. Besides, while all the models have acceptable Malaysian data performance, the most accurate results come from the PLCNet. It is faster than other deep learning models to train both German and Malaysian data in terms of runtime. This improvement and highly accurate results, as well as a quick training process, prove that this novel hybrid model is a good choice for STLF tasks. The PLCNet model was also evaluated in different horizons, and it performed better than other deep learning models. The accuracy of the PLCNet in Malaysian experiments for different horizons is between 94.16% and 98.14%. In German data, it is between 82.49% and 91.31%, which are acceptable results compared to other deep learning models' results.

The interest in using artificial neural networks for electric load forecasting is winning ground in research and industries, especially when deployed in IoT applications. According to the discussed results, deep learning models can be the right choice for IoT compared to other techniques. Thus further work could be devoted to using deep learning models such as the PLCNet in this article for online load forecasting tasks.

REFERENCES

[1] *World Energy Outlook 2019*, IEA, Paris, France, 2019, doi: 10.1787/caf32f3b-en.

[2] H. Daki, A. E. Hannani, A. Aqqal, A. Haidine, and A. Dahbi, "Big data management in smart grid: Concepts, requirements and implementation," *J. Big Data*, vol. 4, no. 1, pp. 1–19, Dec. 2017, doi: 10.1186/s40537-017-0070-y.

[3] S. Acharya, Y. Wi, and J. Lee, "Short-term load forecasting for a single household based on convolution neural networks using data augmentation," *Energies*, vol. 12, no. 18, p. 3560, Sep. 2019.

[4] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid—The new and improved power grid: A survey," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 4, pp. 944–980, 4th Quart., 2012.

[5] J. Sharp, "Book reviews," *Int. J. Forecasting*, vol. 2, no. 2, pp. 241–242, 1986.

[6] V. Gupta, "An overview of different types of load forecasting methods and the factors affecting the load forecasting," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 5, no. 4, pp. 729–733, Apr. 2017.

[7] L. Ekonomou, C. A. Christodoulo, and V. Mladenov, "A short-term load forecasting method using artificial neural networks and wavelet analysis," *Int. J. Power Syst.*, vol. 1, pp. 64–68, Jul. 2016.

[8] F. Javed, N. Arshad, F. Wallin, I. Vassileva, and E. Dahlquist, "Forecasting for demand response in smart grids: An analysis on use of anthropologic and structural data and short term multiple loads forecasting," *Appl. Energy*, vol. 96, pp. 150–160, Aug. 2012, doi: 10.1016/j.apenergy.2012.02.027.

[9] C. Xia, J. Wang, and K. Mcmenemy, "Short, medium and long term load forecasting model and virtual load forecaster based on radial basis function neural networks," *Int. J. Electr. Power Energy Syst.*, vol. 32, no. 7, pp. 743–750, Sep. 2010.

[10] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," 2017, *arXiv:1705.03122*. [Online]. Available: http://arxiv.org/abs/1705.03122

[11] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*. [Online]. Available: https://arxiv.org/abs/1609.03499

[12] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: An overview and application in radiology," *Insights Imag.*, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/s13244-018-0639-9.

[13] A. Gasparin and C. Alippi, "Deep learning for time series forecasting: The electric load case," 2019, *arXiv:1907.09207*. [Online]. Available: https://arxiv.org/abs/1907.09207

[14] B. Yildiz, J. I. Bilbao, and A. B. Sproul, "A review and analysis of regression and machine learning models on commercial building electricity load forecasting," *Renew. Sustain. Energy Rev.*, vol. 73, pp. 1104–1122, Jun. 2017.

[15] C. Tian, J. Ma, C. Zhang, and P. Zhan, "A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network," *Energies*, vol. 11, no. 12, p. 3493, Dec. 2018.

[16] W. He, "Load forecasting via deep neural networks," *Procedia Comput. Sci.*, vol. 122, pp. 308–314, 2017, doi: 10.1016/j.procs.2017.11.374.

[17] C. Gallicchio and A. Micheli, "Deep echo state network (DeepESN): A brief survey," 2017, *arXiv:1712.04323*. [Online]. Available: http://arxiv.org/abs/1712.04323

[18] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *Int. J. Forecast.*, vol. 14, pp. 35–62, Mar. 1998.

[19] M. Q. Raza and A. Khosravi, "A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings," *Renew. Sustain. Energy Rev.*, vol. 50, pp. 1352–1372, Oct. 2015.

[20] P.-H. Kuo and C.-J. Huang, "A high precision artificial neural networks model for short-term energy load forecasting," *Energies*, vol. 11, no. 1, p. 213, Jan. 2018, doi: 10.3390/en11010213.

[21] A. Liaw and M. Wiener, "Classification and Regression by random Forest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[22] C. Gerwig, "Short term load forecasting for residential buildings—An extensive literature review," in *Intelligent Decision Technologies. IDT* (Smart Innovation, Systems and Technologies), vol. 39, R. Neves-Silva, L. Jain, and R. Howlett, Eds. Cham, Switzerland: Springer, 2017.
[23] L. Hu and G. Taylor, "A novel hybrid technique for short-term electricity price forecasting in UK electricity markets," *J. Int. Council Electr. Eng.*, vol. 4, no. 2, pp. 114–120, Apr. 2014, doi: 10.5370/JICEE.2014.4.2.114.
[24] C.-J. Huang and P.-H. Kuo, "Multiple-input deep convolutional neural network model for short-term photovoltaic power forecasting," *IEEE Access*, vol. 7, pp. 74822–74834, 2019, doi: 10.1109/ACCESS.2019.2921238.
[25] H. J. Sadaei, P. C. D. L. E. Silva, F. G. Guimarães, and M. H. Lee, "Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series," *Energy*, vol. 175, pp. 365–377, May 2019.
[26] F. M. Bianchi, E. Maiorino, M. C. Kampmeyer, A. Rizzi, and R. Jenssen, *Recurrent Neural Networks for Short-Term Load Forecasting* (SpringerBriefs in Computer Science). Springer, 2017, doi: 10.1007/978-3-319-70338-1.
[27] R. M. Pratapa and A. Laxmi, "IOT based online load forecasting using machine learning algorithms," *Procedia Comput. Sci.*, vol. 171, pp. 551–560, 2020, doi: 10.1016/j.procs.2020.04.059.
[28] N. Amjady, "Short-term hourly load forecasting using time-series modeling with peak load estimation capability," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 798–805, Nov. 2001, doi: 10.1109/59.962429.
[29] P. Ji, D. Xiong, P. Wang, and J. Chen, "A study on exponential smoothing model for load forecasting," in *Proc. Asia–Pacific Power Energy Eng. Conf.*, Mar. 2012, pp. 1–4.
[30] M. D. Reddy and N. Vishali, "Load forecasting using linear regression analysis in time series model for RGUKT, RK valley campus HT feeder," *Int. J. Eng. Res. Technol.*, vol. 6, no. 5, pp. 624–625, 2017.
[31] G. Dudek, "Pattern-based local linear regression models for short-term load forecasting," *Electr. Power Syst. Res.*, vol. 130, pp. 139–147, Jan. 2016.
[32] E. Ceperic, V. Ceperic, and A. Baric, "A strategy for short-term load forecasting by support vector regression machines," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4356–4364, Nov. 2013.
[33] B.-J. Chen, M.-W. Chang, and C.-J. Lin, "Load forecasting using support vector machines: A study on EUNITE competition 2001," *IEEE Trans. Power Syst.*, vol. 19, no. 4, pp. 1821–1830, Nov. 2004.
[34] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: A review and evaluation," *IEEE Trans. Power Syst.*, vol. 16, no. 1, pp. 44–55, Feb. 2001.
[35] X. Ying, "An overview of overfitting and its solutions," *J. Phys., Conf. Ser.*, vol. 1168, Feb. 2019, Art. no. 022022, doi: 10.1088/1742-6596/1168/2/022022.
[36] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, *arXiv:1609.04747*. [Online]. Available: http://arxiv.org/abs/1609.04747
[37] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015. [Online]. Available: http://arxiv.org/abs/1412.6980
[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[39] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, pp. 1–12, Dec. 2018.
[40] S. Muzaffar and A. Afshari, "Short-term load forecasts using LSTM networks," *Energy Procedia*, vol. 158, pp. 2922–2927, Feb. 2019.
[41] B.-S. Kwon, R.-J. Park, and K.-B. Song, "Short-term load forecasting based on deep neural networks using LSTM layer," *J. Electr. Eng. Technol.*, vol. 15, no. 4, pp. 1501–1509, Jul. 2020.
[42] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*. Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105.
[44] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[45] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Appl. Intell.*, vol. 42, no. 4, pp. 722–737, 2015.
[46] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 565–571.
[47] B. Stephen, X. Tang, P. R. Harvey, S. Galloway, and K. I. Jennett, "Incorporating practice theory in sub-profile models for short term aggregated residential load forecasting," *IEEE Trans. Smart Grid*, vol. 8, no. 4, pp. 1591–1598, Jul. 2017, doi: 10.1109/TSG.2015.2493205.
[48] A. Z. Hinchi and M. Tkiouat, "Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network," *Procedia Comput. Sci.*, vol. 127, 2018, Art. no. 123132.
[49] N. Srivastava, R. Salakhutdinov, A. Krizhevsky, I. Sutskever, and G. Hinton, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[50] J. Sola and J. Sevilla, "Importance of input data normalization for the application of neural networks to complex industrial problems," *IEEE Trans. Nucl. Sci.*, vol. 44, no. 3, pp. 1464–1468, Jun. 1997, doi: 10.1109/23.589532.
[51] J. Li, X. Li, and D. He, "A directed acyclic graph network combined with CNN and LSTM for remaining useful life prediction," *IEEE Access*, vol. 7, pp. 75464–75475, 2019, doi: 10.1109/ACCESS.2019.2919566.

BEHNAM FARSI received the bachelor's degree from the Electrical Engineering Department, Ferdowsi University, Iran, in 2019. He is currently pursuing the master's degree with the Concordia Institute for Information Systems Engineering (CIISE) Department, Concordia University, Canada. His master's thesis topic was on load forecasting. His research interests include machine learning, deep learning, load forecasting, and remaining useful life prediction.

MANAR AMAYRI received the bachelor's degree in power engineering from Damascus University, Syria, the master's degree in electrical power systems from the Power Department, Damascus University, the master's degree in smart grids and buildings from ENES3, INP-Grenoble (Institute National Polytechnique de Grenoble), in 2014, and the Ph.D. degree in energy smart-buildings from the Grenoble Institute of Technology, in 2017. She is currently an Assistant Professor with ENES3, INP-Grenoble, G-SCOP Lab (Sciences pour la conception, l'Optimisation et la Production).

NIZAR BOUGUILA (Senior Member, IEEE) received the Engineer degree from the University of Tunis, Tunis, Tunisia, in 2000, and the M.Sc. and Ph.D. degrees in computer science from Sherbrooke University, Sherbrooke, QC, Canada, in 2002 and 2006, respectively. He is currently a Professor with the Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, QC, Canada. His research interests include machine learning, data mining, computer vision, and pattern recognition applied to different real-life problems.

URSULA EICKER has held leadership positions as a German Physicist with the Centre for Sustainable Energy Technologies, Stuttgart University of Applied Sciences. Since June 2019, she has been leading an ambitious research program to establish transformation strategies toward zero-carbon cities. She is currently the Canada Excellence Research Chair (CERC) for Next Generation Cities with Concordia University, Montreal. Her current research interests include urban scale modeling, zero carbon buildings, renewable energies, and circular economy strategies.