energies

MDPI

Article

High-Resolution Load Forecasting on Multiple Time Scales Using Long Short-Term Memory and Support Vector Machine

Sizhe Zhang, Jinqi Liu and Jihong Wang *

School of Engineering, Univeristy of Warwick, Coventry CV4 7AL, UK
* Correspondence: jihong.wang@warwick.ac.uk

Abstract: Electricity load prediction is an essential tool for power system planning, operation and management. The critical information it provides can be used by energy providers to maximise power system operation efficiency and minimise system operation costs. Long Short-Term Memory (LSTM) and Support Vector Machine (SVM) are two suitable methods that have been successfully used for analysing time series problems. In this paper, the two algorithms are explored further for load prediction; two load prediction algorithms are developed and verified by using the half-hourly load data from the University of Warwick campus energy centre with four different prediction time horizons. The novelty lies in comparing and analysing the prediction accuracy of two intelligent algorithms with multiple time scales and in exploring better scenarios for their prediction applications. High-resolution load forecasting over a long range of time is also conducted in this paper. The MAPE values for the LSTM are 2.501%, 3.577%, 25.073% and 69.947% for four prediction time horizons delineated. For the SVM, the MAPE values are 2.531%, 5.039%, 7.819% and 10.841%, respectively. It is found that both methods are suitable for shorter time horizon predictions. The results show that LSTM is more capable of ultra-short and short-term forecasting, while SVM has a higher prediction accuracy in medium-term and long-term forecasts. Further investigation is performed via blind tests and the test results are consistent.

Keywords: load prediction; SVM; LSTM; multiple time scales

Citation: Zhang, S.; Liu, J.; Wang, J. High-Resolution Load Forecasting on Multiple Time Scales Using Long Short-Term Memory and Support Vector Machine. *Energies* **2023**, *16*, 1806. https://doi.org/10.3390/en16041806

Academic Editor: Silvio Simani

Received: 28 December 2022
Revised: 5 February 2023
Accepted: 9 February 2023
Published: 11 February 2023

1. Introduction

In 2019, the UK amended the previous Climate Change Act of 2008 and set a more ambitious target to achieve net-zero carbon emissions by 2050 [1]. Correspondingly, the electricity market in the UK has seen significant reforms to accommodate more power generation from renewable energy sources for emission reductions [2]. Figure 1 shows the percentage change in power generation from different energy sources in the UK since 1998 [3]. It can be found from Figure 1 that the share of renewable energy generation has significantly increased since 2010, which threatens the stability of the grid and makes it more challenging to maintain a balance between generation and demand. It is well known that power generation must be equal to the load demand. If the gap in between is over a certain range, action must be quickly taken to reduce the gap to an allowable margin within a few or a few ten seconds; otherwise, the power grid stability may not be maintained, and blackout may be triggered [4]. The high penetration of unpredictable power generation from intermittent renewable energy sources makes grid balance maintenance very challenging and costly.

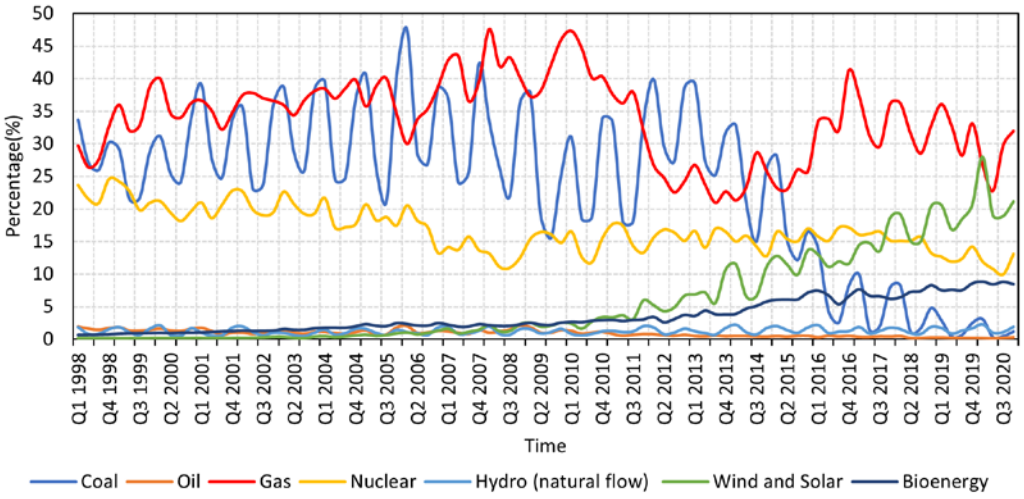

Figure 1. The percentage of different sources in terms of electricity generation from 1998 to 2020.

The challenges are escalating with the integration of new electric devices and new types of usages due to future electrification in heating systems and transportation [5]. According to the UK National Grid electricity system operator, the cost of balancing the UK grid reached £2.65 billion in 2021, up 48% year-on-year [6]. The increased grid balancing costs cause a significant increase in electricity prices, while depriving renewable energy of its potential economic advantages. To minimise the gap between the power generation and usage, effective load prediction is essential, which will allow the operator to plan ahead of scheduling the power generation.

Through predicting future electricity demand, energy companies and utilities can better accordingly plan their generation capacity. This helps minimise the risk of power outages due to overloading the grid or high excess power generation over the load demand. Load forecasting also helps smooth pricing so that electricity can be provided at a competitive rate. It allows energy providers to optimise their operations by managing peak load times more effectively through better scheduling, maintenance plans and investment in infrastructure improvements. The historical load demand used for forecasting is the time series data [7]. Time series data is a set of values that are sequentially listed in time order. Time series data often contain trends and seasonal fluctuations, which need to be considered when analysing the data. Time series data analysis has been popular in recent years and has been applied in areas such as stock price prediction and power management [8]. The functional modelling approach is used for predicting electricity demand and electricity price, which produces superior forecasting results. Results have shown that functional modelling performs better than non-functional techniques, such as autoregressive (AR) [9,10]. The short-term load forecasting problem is addressed through an ensemble learning scheme. The prediction results produced by three base learning algorithms used by a top-level method are more accurate compared to state-of-the-art techniques available [11]. Two customised ARIMA (p,D,q) were used to predict stock prices using Netflix stock historical data over five years. ARIMA (1,1,33) achieved accurate results, which shows the potential for stock forecasting [12]. Figure 2 demonstrates the popular methods used in the last two decades [13].

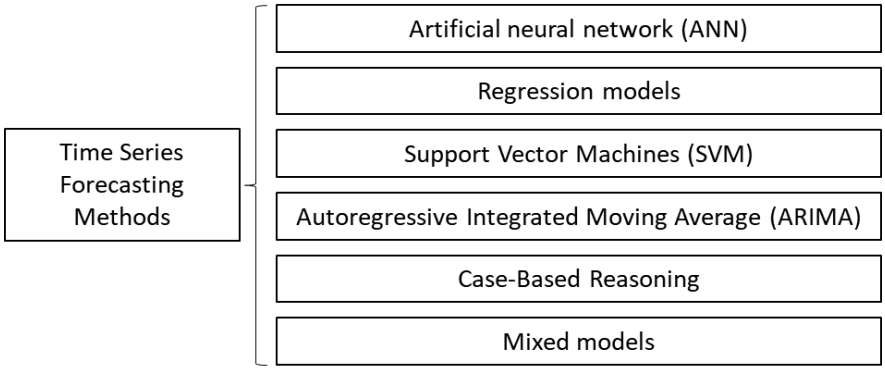

Figure 2. The methodology for prediction time series.

An artificial neural network (ANN) is a type of artificial intelligent algorithm that mimics the behaviour of a biological neuron. They are composed of interconnected neurons, which work together to process information and solve problems. ANNs can be used for time series prediction in a variety of ways. One approach is to use a recurrent neural network (RNN), which is an ANN that uses a sequence of inputs to make predictions. Another approach is to use a multilayer perceptron (MLP), which is an ANN that can be used to analyse time series data and make predictions. LSTM is a type of RNN that has become increasingly popular in recent years due to its ability to effectively process sequential data. It was first introduced by Hochreiter and Schmidhuber in 1997 [14]. A total of 861 NVDI images in two selected regions were used for making the time series data, which was then used for the future vegetation dynamics forecast with LSTM [15]. The previous top-down and bottom-up long-term load forecast methods were unable to incorporate different levels of information. Dong proposed a hybrid LSTM method using sequence prediction for this classic and important task [16]. Long and short-term memory (LSTM), gated recurrent networks and convolutional neural networks were trained to predict daily electricity loads from one to seven days in advance [17]. The prediction results on the test set showed that the LSTM achieved the best performance. Aragón introduces load prediction as continuous input for optimisation models within an optimisation framework for short-term control of complex energy systems, and the LSTM model is used as it allows incremental training in an application with continuous real-time data [18]. Accurate and effective short-term power load forecasting is very important for power systems. Considering the temporal and non-linear characteristics of power load study the application of standard LSTM network and its two typical variants, the Gated Recurrent Unit and the Just Another Networking short-term power load forecasting [19]. A multiscale CNN-LSTM hybrid neural network short-term load forecasting model considering real-time electricity prices was proposed, achieving an accuracy of 98.78% [20]. The prediction results provided a new way for the development of power load forecasting. To overcome the limitations of previous studies and further strengthen prediction performance, a novel short-term power load prediction system, VMD-BEGA-LSTM (VLG), integrating a data pretreatment strategy, advanced optimisation technique and deep learning structure, was developed [21]. In order to improve the accuracy of short-term load forecasting of power systems, a combination model based on LSTM and light gradient boosting machine (LightGBM) was proposed. The experiment first decomposed historical load data by empirical mode decomposition, used historical weather data and load data decomposed by empirical mode decomposition to establish LSTM prediction model and LightGBM prediction model, respectively, and then these two predicted values were linearly combined to obtain the final predicted value [22]. An integrated evolutionary deep learning method based on complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN), an improved grasshopper optimisation algorithm and LSTM networks was proposed by Hu [23]. Experimental results showed that the integrated evolutionary deep learning method proposed in Hu's paper was an effective tool for STLF.

Alsharekh developed an innovative framework for short-term electricity load forecasting, which consists of two phases: data cleaning and a residual convolutional neural network with a multilayer LSTM architecture [24].

Support vector machine (SVM) was developed by Vapnik and Alexey in 1963 and is a supervised learning algorithm that uses labelled training data to identify the best hyperplane, which can act as a decision boundary for categorising new data points [25]. This hyperplane partitions the input space into two or more classes by maximising the margin between them. SVM can be used in both binary and multi-class classification problems, as well as regression problems where the output is real values. The SVM is combined with the fuzzy set theory to develop a novel fuzzy twin support vector machine model, which is applied to predict stock price trends. The fuzzy twin SVM performs better when facing outliers, accounting for its better performance when handling data containing noise [26]. M. Shao developed an SVM energy consumption prediction model to predict the energy consumption of hotel buildings, and the MSE value of the prediction result was 2.22%, and the R2 value was 0.94 after optimisation [27]. Francis proposed ε-Descending Support Vector Machines (ε-DSVMs) to solve the non-stationary input data and reduce the number of support vectors compared with conventional SVM [28]. A two-phases training SVM was introduced by Zhiwang [29]. The two phases correspond to two different linear programming models that improve prediction accuracy and performance. Particle swarm optimisation (PSO) is integrated into the support vector machine to predict the thermal load, and the PSO is utilised to find the optimal SVM parameters [30].

SVM and LSTM have been evaluated as effective time series load forecasting methods. SVM has the advantage of handling non-linear data patterns and robustness against outliers, which enables it to deal with high-dimensional datasets and requires less computational resources than other methods. LSTM can effectively capture long-term dependencies and remember information over extended periods, making it ideal for load forecasting applications. Additionally, LSTM networks are able to quickly and accurately handle large amounts of data without sacrificing accuracy. As such, they are often preferred over traditional models when dealing with high-dimensional datasets that contain complex patterns and temporal dynamics. Therefore, in this paper, two intelligent algorithms, LSTM and SVM, are applied for load forecasting; the University of Warwick campus energy data is used for algorithm refinement and verification. The novelty is to compare and analyse the prediction accuracy of two intelligent algorithms with multiple time scales and to explore better scenarios for their prediction applications. The high-resolution load forecasting over a long range of time is conducted in this paper. It is not common to predict load demand data for 48 × 7 and 48 × 30 sizes at one time. The results confirm the algorithms' effectiveness and their suitability for different time horizon load prediction tasks.

The rest of the article is organised as follows. Section 2 introduces the principles of the two intelligent algorithms and their development for load forecasting. Some pre-prediction work and a prediction flow chart of the two algorithms are shown in Section 3. Prediction results of the two algorithms with multiple time scales in various application scenarios are demonstrated and compared in Section 4. This chapter also presents a blind test. Finally, Section 5 contains the conclusion.

2. Introduction and Development of Intelligence Algorithms for Load Forecasting

2.1. The Principle of LSTM

LSTM (Long Short-Term Memory) is a type of ANN that is employed in deep learning applications. It is designed to remember information for long periods and is especially effective in recognising patterns in data sequences [14]. The unique design makes it well-suited for tasks such as language translation, natural language processing and time series prediction [31]. The LSTM architecture is capable of learning both short-term and long-term dependencies, allowing the network to store information from previous inputs in its

memory cells. This information can then be used to influence the current output. The memory cells also act as a form of memory for the network, allowing it to remember information from prior inputs [32].

As shown in Figure 3, unlike traditional recurrent neural networks, which can struggle to remember long-term dependencies, LSTMs use gates to store and forget information. These gates act as filters, allowing the model to focus on essential information while allowing it to forget the rest [33]. This makes them better suited to dealing with long-term dependencies than traditional recurrent neural networks. The gates in an LSTM consist of various components, including an input gate, an output gate, a forget gate and a memory cell [34]. The input gate is responsible for controlling the flow of information into a cell by determining which parts of the current input should be added to the cell's state. The output gate controls the flow of information from the cell state to the output. It takes input from both the cell state (the current memory) and the hidden layer, combining them together to generate an output that can be used by other layers or passed on as an external prediction. The forget gate controls which information will be retained or forgotten. It takes into account the current input, as well as past inputs and outputs, to decide whether to reset a cell's state (forget) or keep it unchanged (remember). Figure 3 shows the LSTM network structure. As can be seen, a single LSTM unit consists of forget gate G_t^f, input gate G_t^i, output gate G_t^o and candidate states $\widetilde{c_t}$. The external input of the current moment and the output and the cell state of the previous moment form the input of network. The equation of each gate is:

$$G_t^f = \sigma\left(a_f \bullet [h_{t-1}, x_t] + b_f\right) \tag{1}$$

$$G_t^i = \sigma(a_i \bullet [h_{t-1}, x_t] + b_i) \tag{2}$$

$$G_t^o = \sigma(a_o \bullet [h_{t-1}, x_t] + b_o) \tag{3}$$

where h_{t-1} and x_t are the output of last state and present input, respectively. The $a_{k(k=f,i,o)}$ and $b_{k(k=f,i,o)}$ are the weight and bias for each gate. The candidate states $\widetilde{c_t}$, cell states c_t and output h_t can be written as:

$$\widetilde{c_t} = tanh(a_c \bullet [h_{t-1}, x_t] + b_c) \tag{4}$$

$$c_t = G_t^f * c_{t-1} + G_t^i * \widetilde{c_t} \tag{5}$$

$$h_t = G_t^o * tanh(c_t) \tag{6}$$

With the weight a and bias b, and activation functions σ for each gate, the cost function (CF) can be adopted to optimise the LSTM models. The ANN output can be expressed as follows:

$$\tilde{y} = f(a_{k(k=f,i\,o)}, b_{k(k=f,i\,o)}, \sigma_{k(k=f,i\,o)}) \tag{7}$$

The $\tilde{y}$ is actual networks output. If the y is used to represent the expected output, the error e between the $\tilde{y}$ and y is:

$$e = |y - \tilde{y}| \tag{8}$$

The error in LSTM refers to the difference between the actual output and the expected output of the model, which is calculated by taking the difference between the predicted output and the actual output. Absolute error measures the magnitude of the error regardless of its direction. It is used to measure model accuracy and can be used as a metric for model selection and optimisation. Absolute error in LSTM is a measure of how far off the prediction is from the actual output value. It is a measure of the total error in the model, regardless of the direction, so it is not affected by the sign of the error. It is calculated by taking the absolute value of the difference between the predicted output and the actual

output. This is important because it allows us to compare models of different sizes and types while still providing an accurate measure of their performance.

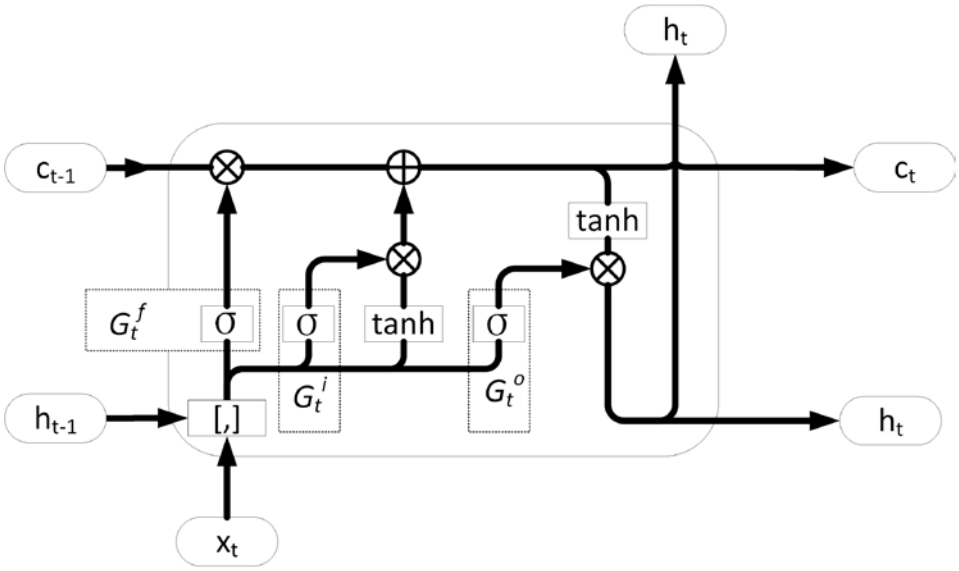

Figure 3. LSTM network structure.

With the error, gradient descent can be implemented. Gradient descent is an optimisation algorithm used to minimise some functions by iteratively moving in the direction of the steepest descent as defined by the negative gradient [35]. It is used to find the local minimum of a function. In machine learning, it is used to update the parameters of a model to minimise a loss function. The algorithm works by taking steps proportional to the negative gradient of the function at the current point. The size of the step is determined by a learning rate parameter which determines the size of the step taken in the direction of the negative gradient. This process of taking steps is repeated until a local minimum is achieved.

The factor that affects the forecasting accuracy with an LSTM model will depend on the data used and the complexity of the model [36]. Factors that can affect the accuracy include the length of the time series, the number of input and output variables, the type of data (e.g., time-series or non-time-series) and the amount of training data. Other factors include the hyperparameters used to train the model, such as the learning rate, the number of hidden layers and the size of the neural network [37]. Finally, the choice of optimiser and loss function can also affect the accuracy of the model [38].

2.2. The Principle of SVM

Support vector machine has been applied in classification and regression in many scenarios. The support vector regression (SVR) is adopted to conduct the load prediction, which contains the linear SVR and the non-linear SVR.

2.2.1. Linear Support Vector Regression

Suppose that the input series data at moment m is:

$$z_n = \{(x_1, y_1), (x_2, y_2) \cdots (x_m, y_m)\} \quad (9)$$

where x_i, $i = 1, \ldots\ldots m$ are the normalised input load value samples, y_i, $i = 1, \ldots\ldots m$ are the normalised actual load values corresponding to input sample i. The total number of samples are m. For SVR, a smoothly approximating function $f(x)$ through training input samples can be found to achieve the minimum error between actual load values and predicted output values [7]. Figure 4 shows the function $f(x)$ that needs to be found.

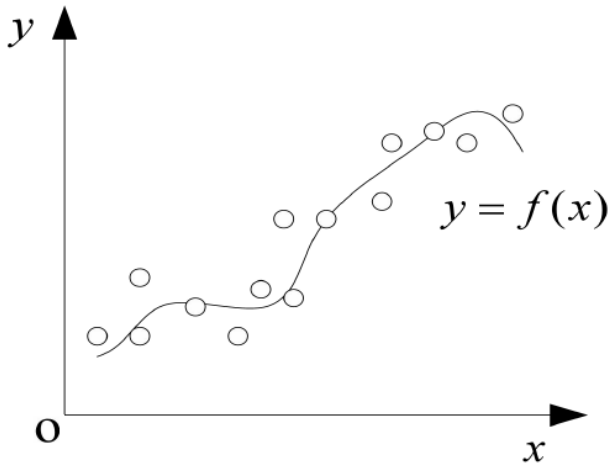

Figure 4. Regression function.

The SVR function can be written as follows:

$$f(x) = \langle w \cdot x_i \rangle + b \tag{10}$$

where w, b is the weighting factor and bias parameter, and $\langle \cdot \rangle$ represents the inner product in feature space. A minimum w needs to be determined to get a smoothly $f(x)$, which can be achieved through minimising the norm as the following form.

$$\min \frac{1}{2}\|w\|^2 \tag{11}$$

Subject to:

$$\begin{cases} y_i - \langle w \cdot x_i \rangle - b \le \varepsilon \\ \langle w \cdot x_i \rangle + b - y_i \le \varepsilon \end{cases} \tag{12}$$

In which ε is the insensitive loss function. Two parallel lines form the ε-region. If the predicted load value locates within the ε-region, the loss is ignored. Otherwise, the loss is the magnitude of the distance between the predicted value and the ε line.

To ensure that the (11) has the solution, two slack variables are introduced:

$$\min[\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)] \tag{13}$$

With the constraints:

$$\begin{gathered} y_i - \langle w \cdot x_i \rangle - b \le \varepsilon + \xi_i \\ \langle w \cdot x_i \rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{gathered} \tag{14}$$

where C is the regularisation parameter, which is also known as the penalty factor. It is used to control the trade-off between maximising the margin of separation between classes and minimising the misclassification errors. It represents the degree of punishment for misclassification. Since it is hard to find the solution of (13), four Lagrange multipliers, α_i, α_i^* and η_i, η_i^*, are introduced to transform the above problem into lagrangian function:

$$\begin{aligned} L = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) - \sum_{i=1}^{n}\alpha_i(\varepsilon + \xi_i - y_i + \langle w \cdot x_i \rangle + b) \\ - \sum_{i=1}^{n}\alpha_i^*(\varepsilon + \xi_i - y_i - \langle w \cdot x_i \rangle - b) - \sum_{i=1}^{n}(\eta_i\xi_i + \eta_i^*\xi_i^*) \end{aligned} \tag{15}$$

The following equations can be achieved based on KKT condition:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)x_i = 0 \tag{16}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \tag{17}$$

$$\frac{\partial L}{\partial \xi^{(*)}} = C - \alpha_i{}^{(*)} - \eta_i{}^{(*)} = 0 \tag{18}$$

Substituting (16)–(18) into (15) can transform it into a dual optimisation problem:

$$\begin{aligned} maximize\ L = -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)\left(\alpha_j - \alpha_j^*\right)\langle x_i \cdot x_j\rangle - \varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) \\ + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) \end{aligned} \tag{19}$$

Subject to:

$$\begin{cases} \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i - \alpha_i^* \in [0, C] \end{cases} \tag{20}$$

The equation of weighting factor and the $f(x)$ can be written as follows:

$$\begin{cases} w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\, x_i \\ f(x) = \sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)\langle x_i \cdot x\rangle + b \end{cases} \tag{21}$$

2.2.2. Non-Linear Support Vector Regression

Load demand data is usually non-linear, calling for a more complex regression model to deal with the more complex non-linear relationship between data. The non-linear SVR problem can be transformed into the linear SVR problem through mapping input samples x_i to a higher dimensional space to obtain $\varphi(x_i)$ [39]. Then, the solution of the non-linear SVR is similar to the linear SVR. The kernel function is introduced to replace the sophisticated inner product operation between input samples:

$$K(x_i, x_j) = \langle \varphi(x_i) \cdot \varphi(x_j)\rangle \tag{22}$$

In a high-dimensional feature space, the expression for the linear regression function is:

$$f(x) = \langle w \cdot \varphi(x_i)\rangle + b \tag{23}$$

Like the linear case, the dual optimisation problem for non-linear regression can be achieved:

$$\begin{aligned} maximize\ L = -\tfrac{1}{2}\textstyle\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)\left(\alpha_j - \alpha_j^*\right)K(x_i, x_j) - \varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \\ \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) \end{aligned} \tag{24}$$

Subject to:

$$\begin{cases} \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i - \alpha_i^* \in [0, C] \end{cases} \tag{25}$$

The solution of non-linear regression is:

$$\begin{cases} w = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)\, \varphi(x_i) \\ f(x) = \sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*) K(x_i \cdot x_j) + b \end{cases} \quad (26)$$

In the calculation of non-linear regression, the kernel function $K(x_i \cdot x_j)$ is used instead of getting the mapping function $\varphi(x_i)$. Selecting an optimal kernel function can significantly improve the prediction performance and several kernel functions are tested for this study. The Gaussian Radial Basis Function (RBF) kernel is used to perform the prediction in this paper, the formula is shown as follows:

$$K(x_i \cdot x_j) = \exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right) \quad (27)$$

2.2.3. Adapting the Algorithms for Load Forecasting

The non-linear regression case is used in this paper. When applying the algorism to load forecasting, the input and output of the model need to be identified before training is initiated; i.e., define the training sample array (x_m, y_m). In this application, x_m consists of the load values and selected features before the load values are predicted and y_m contains the load values to be predicted, the feature selection will be clarified in the following section. In the non-linear case, it is required to map x_i to a high-dimensional linear Hilbert space to get $\varphi(x_i)$. Then, the kernel function needs to be determined according to Equation (27); hence, the selection of the width coefficient of RBF σ is important. In addition, the α_i, α_i^* will be calculated through LIBSVM in Python; since the regularisation parameter *C* plays an important role in this process, how to determine *C* is vital. In this paper, the cross-validation method is addressed to select two parameters, which is presented in the following section. After determining these variables, the load value to be predicted can be achieved through Equation (26).

3. Applying Two Algorithms for Load Forecasting

3.1. Prediction Time Horizons Determination

Since the prediction with different time scales performs different roles in various applications, shorter time horizon forecasting is essential for electricity grid operators to ensure reliable and cost-effective power generation and delivery [40]. It also helps reduce costs associated with oversupply or undersupply of electricity, providing an efficient energy market. The longer time horizon load forecasting enables utilities and energy providers to plan ahead and make informed decisions regarding investments in infrastructure, generation capacity and supply contracts, which allows them to optimise their operations by providing timely information about expected customer needs [16]. Thus, four different prediction time horizons scenarios based on the half-hour resolution are addressed in this paper, and all prediction is implemented on Python software:

- Ultra-short-term load prediction (USTLP) is to forecast load value after half an hour
- Short-term load prediction (STLP) is to predict load values in the next day
- Medium-term load prediction (MTLP) is to predict load values in the next week
- Long-term load prediction (LTLP) is to predict load values in the next month

3.2. Data Collection and Processing

The half-hour recorded data were collected from the University of Warwick (UoW) campus energy consumption data for both 2020 and the first nine months of 2021, which were used to perform as training and test datasets, respectively.

The selection of features is vital for the prediction performance. Suitable feature selection can effectively improve prediction accuracy. Three sets of information are taken into consideration based on the previous study in this paper: Calendar information, Weather data and Historical load demands [41]. The calendar information encompasses four features: an integer from 1 to 365 representing each day of the year, an integer from 0 to 6 signifying Sunday through Saturday, a timeslot label with integers 0–47 for thirty-minute intervals throughout the day and three binary digits distinguishing different types of days such as weekdays, weekends, term time and holidays. Temperature, humidity and wind speed are chosen as weather data labels. The data used are presented in Figure 5 [42].

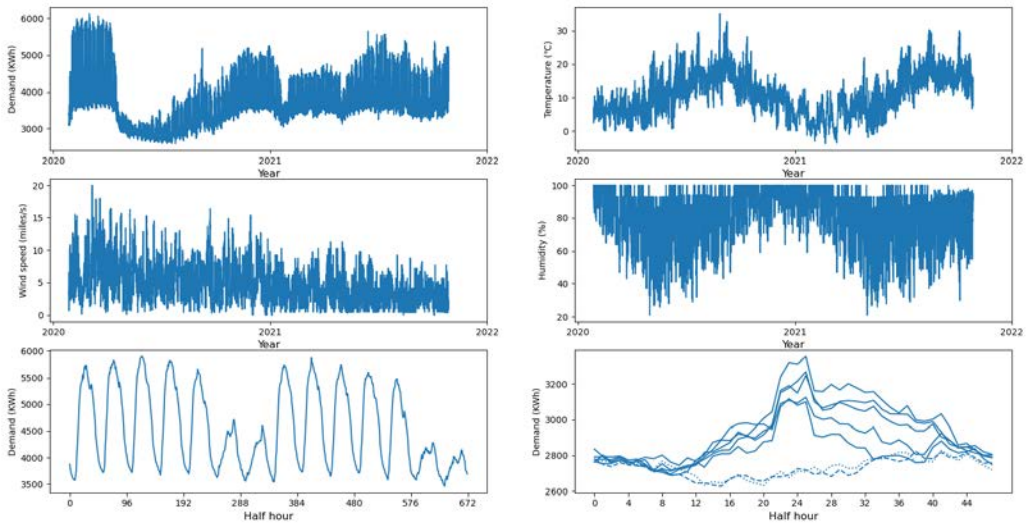

Figure 5. Load demand and features data. Load demand data for the period 1 January 2020–31 September 2021 (**top left**); Temperature data for the period 1 January 2020–31 September 2021 (**top right**); Wind speed data for the period 1 January 2020–31 September 2021 (**middle left**); Humidity data for the period 1 January 2020–31 September 2021 (**middle right**); daily and weekly periodicities in the period 2 March 2020–15 March 2020 (**bottom left**); daily load demand curves in a week for the period 18 May 2020–24 May 2020. Solid lines: weekdays; dashed lines: Saturdays; dotted lines: Sundays (**bottom right**).

Data normalisation is an essential pre-processing step before conducting load prediction. By transforming the data into a standard scale, it helps to improve the accuracy of machine learning algorithms. In addition, normalising data can reduce the computational complexity of training models by eliminating redundant features and facilitating faster convergence during optimisation processes. In this paper, data are mapped to [0, 1] interval through data normalisation using the equation as follows:

$$x_{i,scaled} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{28}$$

where x_{max} denotes the maximum value in the sample data and x_{min} stands for the minimum value. x_i and $x_{i,scaled}$ are load values before and after the data normalisation. The corresponding inverse normalisation is required after the prediction:

$$x_i = x_{min} + x_{i,scaled}(x_{max} - x_{min}) \tag{29}$$

In addition, there are some outliers in the load demand data due to the mismeasurement of utilities, which interfere with the training process, accuracy and reliability. The load demand is adjusted via linear fitting to exclude outliers.

3.3. Error Evaluation Index

Three evaluation metrics are chosen to evaluate the prediction performance of two algorithms. They are Root-Mean-Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE):

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(A_i - F_i)^2} \tag{30}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|A_i - F_i| \tag{31}$$

$$\mathrm{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|A_i - F_i|}{A_i} \times 100 \tag{32}$$

where the data size is n, A_i represents the actual load values for i times and F_i denotes the predicted values for i times.

3.4. Load Prediction Flow Chart of the Two Algorithms

Figures 6 and 7 show the load prediction flow charts of the two algorithms. For the LSTM, hyperparameters are set at first. Then, the weight and bias are randomly initialised. To minimise the discrepancy between actual values and predicted values, gradient descent is utilised. The error is dependent on both weight and bias. In this proposed LSTM technique, Mini-batch Gradient Descent (MBGD) is adopted for further optimisation purposes. For the SVM, the first step is to collect and pre-process load demand data. The training sample arrays (x_m, y_m) can be obtained afterwards based on the processed load data and the chosen features. Initial parameters and coefficients are then determined so that an initial model for load prediction can be established. The last step is to carry out the optimisation.

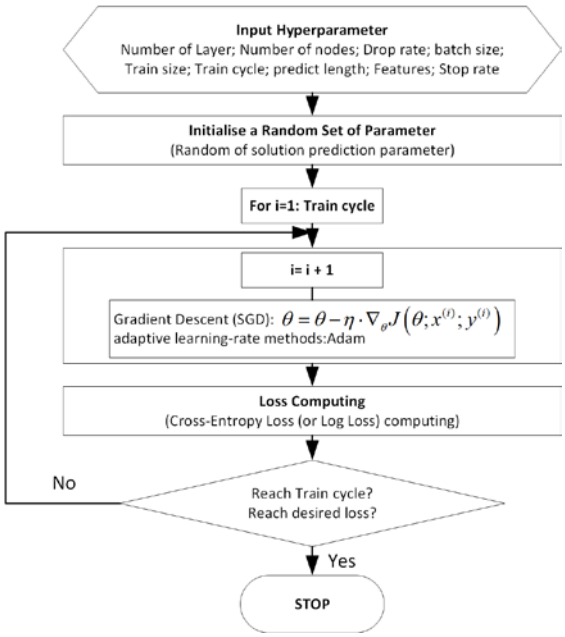

Figure 6. LSTM prediction flow chart.

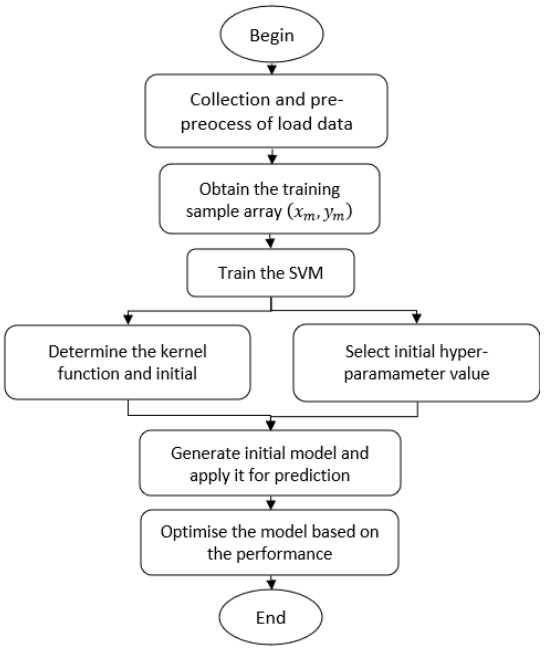

Figure 7. SVM prediction flow chart.

4. Prediction Results of Using the Two Algorithms

4.1. Algorithms Model Structure and Hyperparameters

In this paper, the table search method is addressed to the best combination of structures or hyperparameters for both models. Various combinations of hyperparameters are evaluated and compared to achieve optimal prediction performance. The optimal structure and hyperparameters after the epoch/training time are shown in Table 1. The training/epoch time is calculated on an Intel® Xeon Silver 4114 CPU 2.20 GHz and 64 GB RAM with Windows Server 2019 system. The USTLP case is chosen to show the tuning process for two algorisms, as in Figures 8 and 9.

Table 1. Model parameters and training/epoch time for LSTM and SVM.

		LSTM		
	Layer number	Nodes number	Batch size	Epoch time (s)
USTLP	10	10	39	132
STLP	10	40	13	403
MTLP	10	50	6	494
LTLP	10	100	4	2085
		SVM		
	C	$gamma$		training time (s)
USTLP	1	0.03		37
STLP	1	0.01		499
MTLP	0.2	0.01		613
LTLP	0.1	0.005		1192

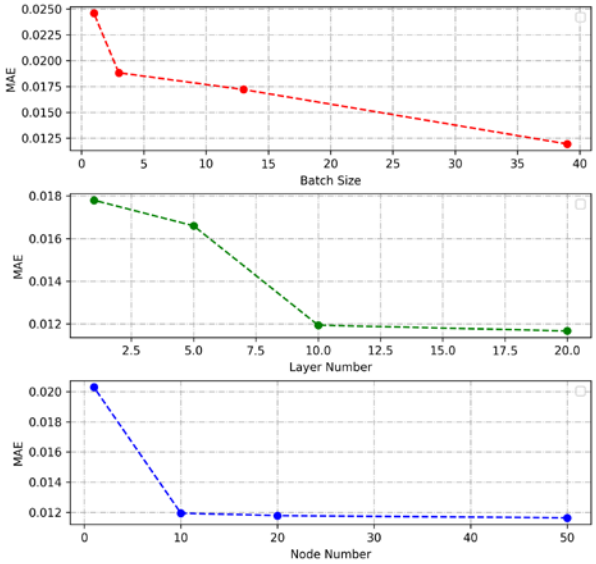

Figure 8. Tuning process of LSTM structure.

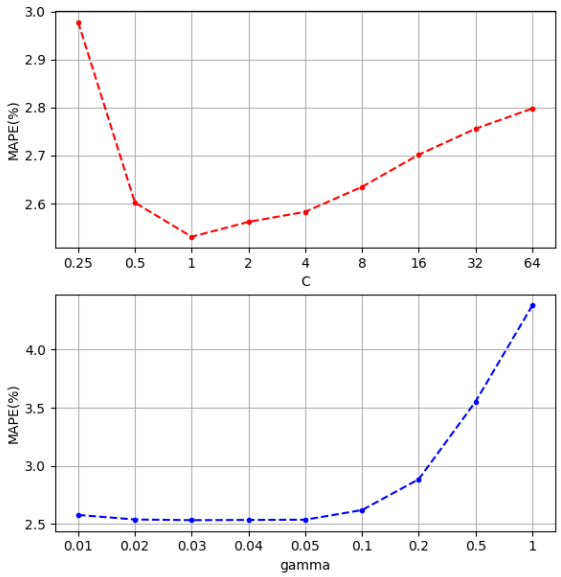

Figure 9. Tuning process of SVM hyperparameters.

The rolling window technique is adopted in this paper for well-trained models. The parameters of rolling window techniques are demonstrated in Table 2.

Table 2. Rolling window technique parameters for both algorithms.

	Window Size	Step Size
USTLP	48	1
STLP	48 × 7	48
MTLP	48 × 7 × 4	48
LTLP	48 × 7 × 4	48

For neural networks, differences in model structure have a significant impact on prediction results. The batch size determines the direction of the gradient descent of the model, and therefore, directly determines the direction of the optimisation of the model. Too small a batch size makes the update direction of the model uncertain, so the model

tends to be trapped in a slight gradient difference, and thus it cannot find an optimal solution. A large batch size has a similar effect. The update direction of the gradient depends on the gradient of multiple trainings, so the best gradient on the network is diluted. As shown in Figure 8, the larger batch size has higher performance on the proposed model. Both the number of layers and the number of nodes in the network determine the complexity of the model, so a higher number of layers and cells will generally result in higher training results. However, an increased number of layers and cells can also increase the training burden and resilience of the model. More layers and cells in the network may also lead to training failure. The proposed model performs better when the number of layers and nodes reaches 10, as shown in Figure 8. Increasing the number of layers and nodes further will significantly increase the training time of the model.

Regarding the SVM method, the tuning process of hyperparameters is demonstrated in Figure 9. Selecting appropriate parameters C and σ plays an important role in improving the prediction performance of model. If the regularisation parameter C is too large, the algorism will try to generate a hyperplane that endeavours to classify each data point correctly. However, it may lead to a loss in generalisation properties of the classifier. On the contrary, a small value of C may enhance the freedom of the model, leading to a large training error. For the σ parameter, a small value constraints the ability of the model to capture the complexity of the data, while a large σ may cause overfitting of the model [43].

4.2. Prediction Results Presentation

The prediction results of two algorithms for four different time horizons prediction are shown in Figures 10 and 11.

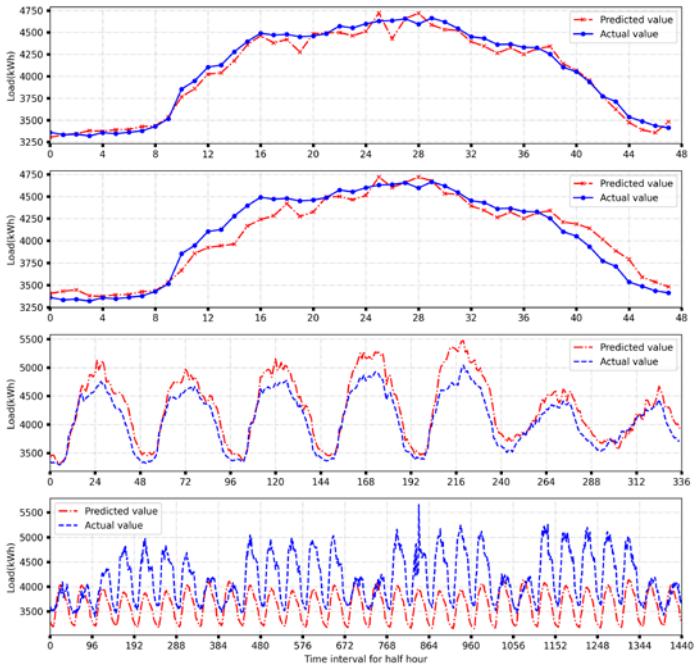

Figure 10. Prediction results for LSTM in four prediction scenarios. Predicted and actual load demand in USTLP scenario for 21 September 2021 (**first row**); Predicted and actual load demand in STLP scenario for 21 September 2021 (**second row**); Predicted and actual load demand in MTLP scenario for the period 11 October 2021–17 October 2021 (**third row**); Predicted and actual load demand in LTLP scenario for the period 1 May 2021–30 May 2021 (**fourth row**).

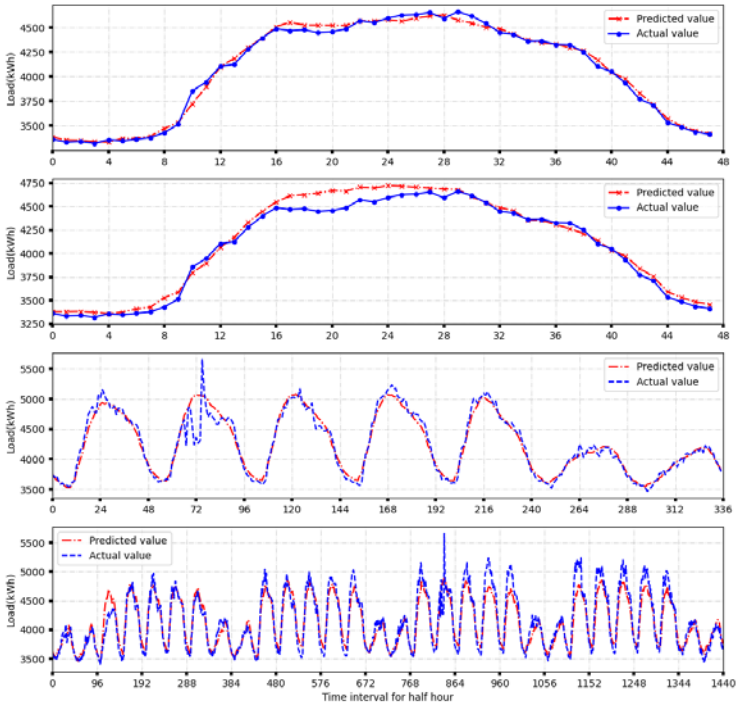

Figure 11. Prediction results for SVM in four prediction scenarios. Predicted and actual load demand in USTLP scenario for 21 September 2021 (**first row**); Predicted and actual load demand in STLP scenario for 21 September 2021 (**second row**); Predicted and actual load demand in MTLP scenario for the period 11 October 2021–17 October 2021 (**third row**); Predicted and actual load demand in LTLP scenario for the period 1 May 2021–30 May 2021 (**fourth row**).

A typical day, week and month in the test dataset were selected for four prediction scenarios to show the results. Four subplots in Figures 10 and 11, from top to bottom, represent USTLP, STLP, MTLP and LTLL, respectively. It can be found from the two figures that the discrepancy between the actual and predicted load demand becomes larger with a longer prediction time horizon, indicating a worse prediction performance. It can be concluded that both computational algorithms are more suitable for shorter time horizon load prediction.

4.3. Comparison of Prediction Performance for Two Algorithms

The prediction result comparison and values of error indexes in the test dataset are shown in Figure 12 and Table 3. It can be found from the figure and table that the LSTM has a better prediction performance according to the USTLP and STLP scenarios, while the SVM achieves better prediction accuracy in MTLP and LTLP cases. Since the deeper network level can fit more non-linear relationships and LSTM can investigate the hidden information in time series data, LSTM performs better for shorter time horizon prediction. The worse prediction accuracy for LSTM in MTLP and LTLP cases is due to the size of the data. Big-size data in weeks or months make the LSTM challenging to capture data features. However, the better generalisation ability and the kernel function enable the SVM algorithm to jump out of the local minimal to achieve the optimal global solution, which results in SVM's better prediction performance when facing larger-sized data [7].

Hence, to improve the LSTM accuracy in longer time horizon prediction scenarios, a deeper, more complex network (more nodes and more layers) needs to be established to enable LSTM to efficiently cope with larger data sizes. However, Table 1 shows that the epoch time in the LTLP case is 2085s, and the epoch time rapidly increases with the increase in data size, indicating that the exploration is limited by the existing computing

platform performance. Thus, more computing power is required for a more complex LSTM model, which reveals the dependence of the LSTM network on computing power. The LSTM algorithm prediction performance will be significantly improved with a more complex model, while SVM performs better in the exact implementation platform.

Table 3. Model metric for algorithms in the test dataset.

			LSTM	
	USTLP	STLP	MTLP	LTLP
RMSE	0.016	0.018	0.114	0.310
MAE	0.010	0.015	0.109	0.304
MAPE (%)	2.501	3.577	25.073	69.947
			SVM	
	USTLP	STLP	MTLP	LTLP
RMSE	0.015	0.032	0.048	0.063
MAE	0.011	0.022	0.034	0.047
MAPE (%)	2.531	5.039	7.819	10.841

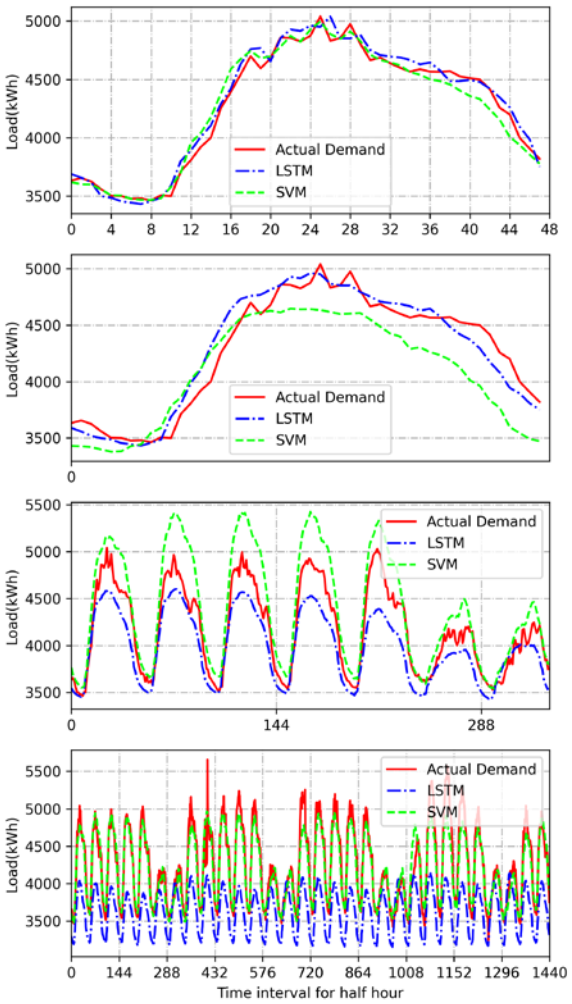

Figure 12. Comparison of prediction results in four prediction scenarios.

Apart from comparing the prediction accuracy of two algorisms, the effectiveness of two methods in various situations is verified in this section. Considering the impact of different seasons, weekdays/weekends, term-time/holiday on load profile. Four typical days belonging to the weekday, weekend, term time and holiday are respectively chosen

according to USTLP and STLP cases. For MTLP and LTLP cases, one typical week and month located in terms of time and holiday are determined. The model metric (RMSE, MAE and MAPE (%)) for all scenarios are presented in Tables 4 and 5, and Figure 13.

Table 4. Model metric for two algorithms in four seasons.

				LSTM		
		Spring	**Summer**	**Autumn**	**Winter**	**Average**
USTLP	RMSE	0.017	0.017	0.012	0.013	0.015
	MAE	0.014	0.013	0.09	0.010	0.031
	MAPE	3.223	2.999	2.008	2.3014	2.633
STLP	RMSE	0.014	0.019	0.015	0.023	0.018
	MAE	0.011	0.014	0.013	0.020	0.015
	MAPE	2.582	3.231	3.043	4.617	3.393
MTLP	RMSE	0.080	0.125	0.146	0.138	0.122
	MAE	0.076	0.122	0.143	0.132	0.118
	MAPE	17.487	28.071	32.907	30.374	27.210
LTLP	RMSE	0.283	0.347	0.269	0.325	0.306
	MAE	0.280	0.343	0.262	0.319	0.301
	MAPE	64.425	78.921	60.284	73.399	69.257
				SVM		
		Spring	**Summer**	**Autumn**	**Winter**	**Average**
USTLP	RMSE	0.015	0.018	0.012	0.020	0.016
	MAE	0.011	0.015	0.010	0.018	0.014
	MAPE	2.191	2.617	2.103	2.834	2.436
STLP	RMSE	0.039	0.058	0.020	0.035	0.038
	MAE	0.031	0.049	0.016	0.026	0.031
	MAPE	5.903	7.551	3.273	5.365	5.523
MTLP	RMSE	0.037	0.048	0.049	0.049	0.046
	MAE	0.028	0.040	0.042	0.042	0.038
	MAPE	6.444	7.735	9.987	9.158	8.331
LTLP	RMSE	0.031	0.062	0.078	0.071	0.061
	MAE	0.025	0.047	0.059	0.052	0.046
	MAPE	6.155	8.453	11.936	9.886	9.108

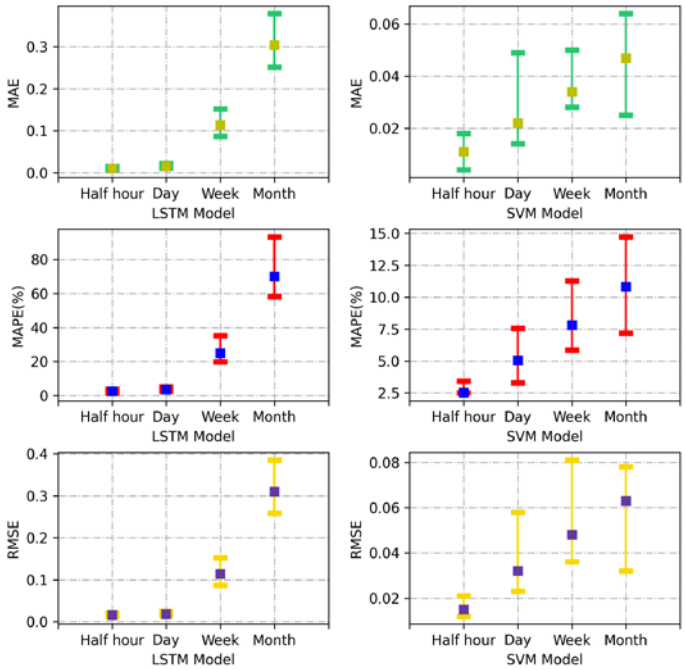

Figure 13. Error evaluation index values and error range for four scenarios of LSTM and SVM. Range of MAE values and values for four prediction scenarios for both algorithms among the test dataset (**first row**); Range of MAPE values and values for four prediction scenarios for both algorithms among the test dataset (**second row**); Range of RMSE values and values for four prediction scenarios for both algorithms among the test dataset (**third row**).

Table 5. Model metric for two algorithms in various scenarios.

				LSTM		
		Weekday	Weekend	Term time	Holiday	Average
USTLP	RMSE	0.015	0.014	0.012	0.013	0.014
	MAE	0.011	0.010	0.009	0.010	0.010
	MAPE	3.455	2.311	2.074	2.309	2.537
STLP	RMSE	0.015	0.021	0.021	0.018	0.019
	MAE	0.011	0.018	0.017	0.015	0.015
	MAPE	2.546	4.148	3.913	3.455	3.516
			Term time		Holiday	Average
MTLP	RMSE		0.111		0.144	0.128
	MAE		0.107		0.141	0.124
	MAPE		34.377		45.301	39.839
LTLP	RMSE		0.274		0.322	0.298
	MAE		0.271		0.319	0.295
	MAPE		62.354		73.402	67.878
				SVM		
		Weekday	Weekend	Term time	Holiday	Average
USTLP	RMSE	0.020	0.014	0.021	0.021	0.019
	MAE	0.016	0.010	0.017	0.017	0.015
	MAPE	2.727	2.544	3.108	3.362	2.935
STLP	RMSE	0.048	0.021	0.050	0.033	0.038
	MAE	0.040	0.016	0.036	0.026	0.030
	MAPE	6.757	3.992	6.081	5.129	5.490
			Term time		Holiday	Average
MTLP	RMSE		0.048		0.058	0.053

	MAE	0.040	0.050	0.045
	MAPE	7.735	10.409	9.072
	RMSE	0.062	0.059	0.061
LTLP	MAE	0.047	0.049	0.048
	MAPE	8.453	13.012	10.733

It can be concluded from Tables 4 and 5 that there is no significant fluctuation for model metrics across four seasons, indicating a relatively stable prediction performance for different seasons throughout the year. However, for MTLP and LTLP cases, the prediction accuracy of the two methods for weeks or months on holiday is lower than that in term time. This is because people's life patterns are more irregular during the holidays, increasing the complexity of load forecasting.

4.4. Blind Test

The blind test is often used to cross-validate the results of load prediction without waiting for events to occur [44]. A stable and good load prediction model should perform well on unseen data, showing the generalisation ability of this model [45]. Note that the blind data cannot be used for training or fitting the model and the model selection. Therefore, the October data in 2021 is determined as the blind date. The load in October is predicted through the data in September 2021. The blind test results of the two algorisms for the STLP, MTLP and LTLP prediction cases are presented in Figures 14 and 15. The model metric for the four scenarios is shown in Table 6.

Table 6. Model metric for two algorisms in a blind test.

		LSTM	**SVM**
	RMSE	0.004	0.002
USTLP	MAE	0.003	0.002
	MAPE (%)	1.142	0.656
	RMSE	0.22	0.041
STLP	MAE	0.17	0.035
	MAPE (%)	3.105	6.747
	RMSE	0.095	0.046
MTLP	MAE	0.071	0.037
	MAPE (%)	16.326	8.087
	RMSE	0.257	0.140
LTLP	MAE	0.253	0.108
	MAPE (%)	58.731	14.559

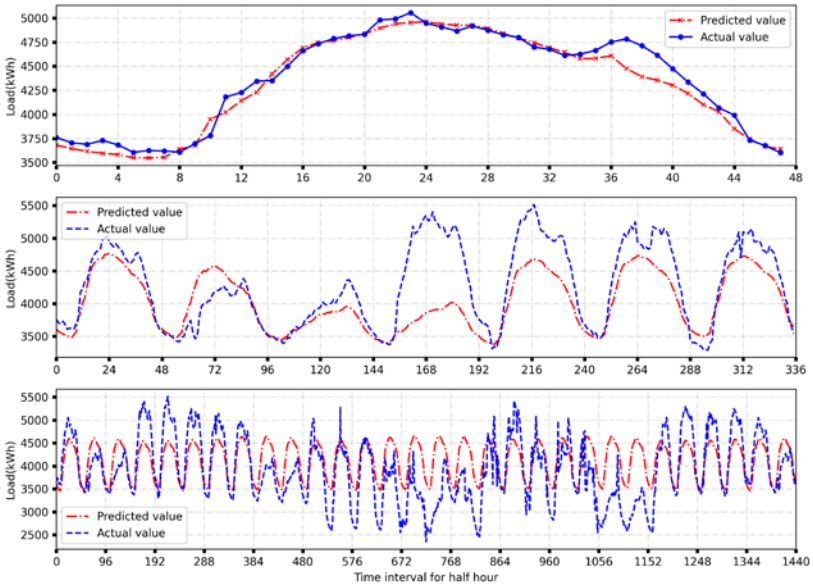

Figure 14. LSTM blind test.

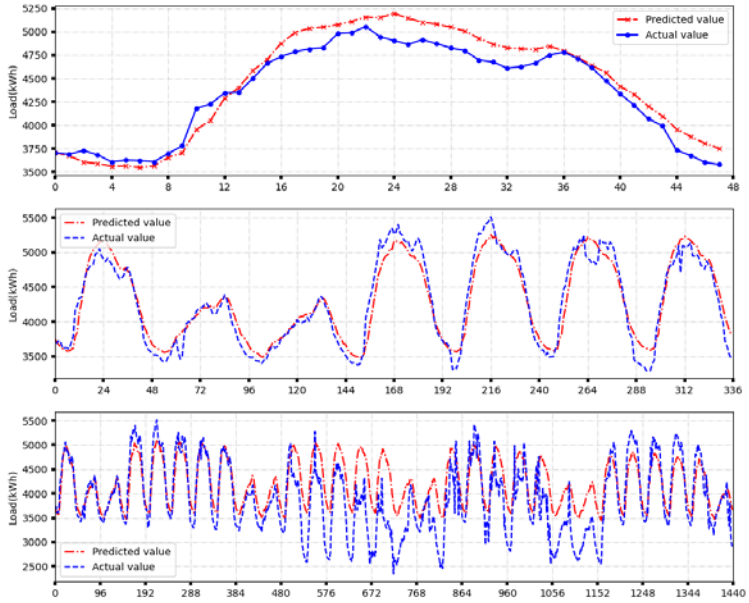

Figure 15. SVM blind test.

The blind test result also verifies that LSTM performs better in the STLP case and SVM achieves better prediction performance for MTLP and LTLP cases.

5. Conclusions

In this paper, two computational algorithms, LSTM and SVM, were introduced and developed in the establishment of the load prediction model. They were adapted to conduct the load prediction based on selected features and historical campus half-hour recorded load demand data in four different prediction time horizons. Prediction performance was evaluated through two aspects: training/epoch time and prediction accuracy for comparison. It was found that both methods had higher prediction accuracy for shorter time horizon prediction scenarios, indicating better applicability in shorter time horizon prediction. Comparison showed that the LSTM had a higher prediction accuracy according to ultra-short and short-term load prediction, while SVM predicted better for medium- and long-term scenarios; the prediction accuracy was also verified through the

blind test. Furthermore, the training time for SVM was shorter in medium- and long-term cases, confirming the high dependence of LSTM prediction accuracy on the computing power of the computer.

The existing platform limited the performance of LSTM in longer time horizon scenarios. So, a more complex LSTM model could be constructed in further stages with more nodes and hidden layers to improve its prediction accuracy when facing larger data sizes. The strategy of selecting hyperparameters can be improved with some algorithms, such as combining the model with the fish swarm algorithm. In addition, some more advanced artificial intelligence algorithms, which have been applied in dealing with time series data, can be compared with the existing two algorithms in more scenarios.

Author Contributions: Conceptualization, J.W.; methodology, S.Z. and J.L.; software, S.Z. and J.L.; validation, S.Z. and J.L.; investigation, S.Z. and J.W.; data curation, J.L.; writing—original draft preparation, S.Z. and J.L.; writing—review and editing, S.Z. and J.W.; visualization, S.Z. and J.L.; supervision, J.W.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by EPSRC Supergen Energy Storage Network Plus (EP/S032622/1) and the PhD studentship from the University of Warwick.

Data Availability Statement: Not applicable.

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

References

1. Tanneberger, F.; Abel, S.; Couwenberg, J.; Dahms, T.; Gaudig, G.; Günther, A.; Kreyling, J.; Peters, J.; Pongratz, J.; Joosten, H. Towards net zero CO_2 in 2050: An emission reduction pathway for organic soils in Germany. *Mires Peat.* **2021**, *27*, 5.
2. Liu, J.; Wang, J.; Cardinal, J. Evolution and reform of UK electricity market. *Renew. Sustain. Energy Rev.* **2022**, *161*, 112317.
3. Littlechild, S. Exploring customer satisfaction in Great Britain's retail energy sector part III: A proposed Overall Customer Satisfaction score. *Util. Policy* **2021**, *73*, 101–299.
4. Venayagamoorthy, G.K. Intelligent sense-making for smart grid stability. In Proceedings of the 2011 IEEE Power and Energy Society General Meeting, Detroit, MI, USA, 24–28 July 2011; pp. 1–3.
5. Wang, C.; Liu, J.; Cheng, H.; Zhuang, Y.; Zhao, Z. A modified one-cycle control for Vienna rectifiers with functionality of input power factor regulation and input current distortion mitigation. Energies 2019, 12, 3375.
6. UK Power System Balancing Costs Rise 48% in 2021 as BM Prices Rocket. Available online: https://www.spglobal.com/commodityinsights/en/market-insights/latest-news/electric-power/012122-uk-power-system-balancing-costs-rise-48-in-2021-as-bm-prices-rocket (accessed on 2 December 2022).
7. Liu, J.; Zhang, S.; Wang, J. Development and Comparison of Two Computational Intelligence Algorithms for Electrical Load Forecasts with Multiple Time Scales. In Proceedings of the 2022 Power System and Green Energy Conference (PSGEC), Shanghai, China, 25–27 August 2022; pp. 637–643.
8. Hajirahimi, Z.; Khashei, M. Hybrid structures in time series modeling and forecasting: A review. *Eng. Appl. Artif. Intell.* **2019**, *86*, 83–106.
9. Shah, I.; Jan, F.; Ali, S. Functional data approach for short-term electricity demand forecasting. *Math. Probl. Eng.* **2022**, *2022*, 6709779.
10. Jan, F.; Shah, I.; Ali, S. Short-Term Electricity Prices Forecasting Using Functional Time Series Analysis. *Energies* **2022**, *15*, 3423.
11. Divina, F.; Gilson, A.; Goméz-Vela, F.; García Torres, M.; Torres, J.F. Stacking Ensemble Learning for Short-Term Electricity Consumption Forecasting. *Energies* **2018**, *11*, 949.
12. Khan, S.; Alghulaiakh, H. ARIMA model for accurate time series stocks forecasting. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 7. https://doi.org/10.14569/IJACSA.2020.0110765.
13. Liu, H.; Yan, G.; Duan, Z.; Chen, C. Intelligent modeling strategies for forecasting air quality time series: A review. *Appl. Soft Comput.* **2021**, *102*, 106957.
14. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780.
15. Reddy, D.S.; Prasad, P.R.C. Prediction of vegetation dynamics using NDVI time series data and LSTM. *Model. Earth Syst. Environ.* **2018**, *4*, 409–419.
16. Dong, M.; Grumbach, L.S. A Hybrid Distribution Feeder Long-Term Load Forecasting Method Based On Sequence Prediction. *IEEE Trans. Smart Grid* **2020**, *11*, 470–482.
17. Ünlü, K.D. A Data-Driven Model to Forecast Multi-Step Ahead Time Series of Turkish Daily Electricity Load. *Electronics* **2022**, *11*, 1524.
18. Aragón, G.; Puri, H.; Grass, A.; Chala, S.; Beecks, C. Incremental Deep-Learning for Continuous Load Prediction in Energy Management Systems. In Proceedings of the 2019 IEEE Milan PowerTech, Milan, Italy, 23–27 June 2019; pp. 1–6.

19. Zhang, Y.; Li, D.; Yang, B. Application of Long-Short Term Memory Network and its Variants in Short-term Power Load Time Series Forecasting. In Proceedings of the 2020 International Conference on Smart Grids and Energy Systems (SGES), Perth, Australia, 23–26 November 2020; pp. 197–202.
20. Guo, X.; Zhao, Q.; Zheng, D.; Ning, Y.; Gao, Y. A short-term load forecasting model of multi-scale CNN-LSTM hybrid neural network considering the real-time electricity price. *Energy Rep.* **2020**, *6*, 1046–1053.
21. Jin, Y.; Guo, H.; Wang, J.; Song, A. A Hybrid System Based on LSTM for Short-Term Power Load Forecasting. *Energies* **2020**, *13*, 6241.
22. Zhou, Y.; Lin, Q.; Xiao, D. Application of LSTM-LightGBM Non-linear Combined Model to Power Load Forecasting. *J. Phys. Conf. Ser.* **2022**, *2294*, 012–035.
23. Hu, H.; Xia, X.; Luo, Y.; Zhang, C.; Nazir, M.S.; Peng, T. Development and application of an evolutionary deep learning framework of LSTM based on improved grasshopper soptimisation algorithm for short-term load forecasting. *J. Build. Eng.* **2022**, *57*, 104975.
24. Alsharekh, M.F.; Habib, S.; Dewi, D.A.A.; Albattah, W.; Islam, M.; Albahli, S. Improving the Efficiency of Multistep Short-Term Electricity Load Forecasting via R-CNN with ML-LSTM. *Sensors* **2022**, *22*, 6913.
25. Schölkopf, B.; Luo, Z.; Vovk, V. *Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik*; Springer Science & Business Media: Berlin/Heidelberg, Germany. 2013.
26. Hao, P.-Y.; Kung, C.-F.; Chang, C.-Y.; Ou, J.-B. Predicting stock price trends based on financial news articles and using a novel twin support vector machine with fuzzy hyperplane. *Appl. Soft Comput.* **2021**, *98*, 106806.
27. Shao, M.; Wang, X.; Bu, Z.; Chen, X.; Wang, Y. Prediction of energy consumption in hotel buildings via support vector machines. *Sustain. Cities Soc.* **2020**, *57*, 102128.
28. Tay, F.E.H.; Cao, L.J. ε-descending support vector machines for financial time series forecasting. *Neural Process. Lett.* **2002**, *15*, 179–195.
29. Zhang, Z.; Gao, G.; Tian, Y.-j.; Yue, J. Two-phase multi-kernel LP-SVR for feature sparsification and forecasting. *Neurocomputing* **2016**, *214*, 594–606.
30. Jie, Z.; Siyuan, W. Thermal Load Forecasting Based on PSO-SVR. In Proceedings of the 2018 IEEE 4th International Conference on Computer and Communications (ICCC), Chengdu, China, 7–10 December 2018; pp. 2676–2680.
31. Hua, Y.; Zhao, Z.; Li, R.; Chen, X.; Liu, Z.; Zhang, H. Deep learning with long short-term memory for time series prediction. *IEEE Commun. Mag.* **2019**, *57*, 114–119.
32. Zhang, J.; Wang, P.; Yan, R.; Gao, R.X. Long short-term memory for machine remaining life prediction. *J. Manuf. Syst.* **2018**, *48*, 78–86.
33. Jing, L.; Gulcehre, C.; Peurifoy, J.; Shen, Y.; Tegmark, M.; Soljačić, M.; Bengio, Y. Gated orthogonal recurrent units: On learning to forget. *Neural Comput.* **2019**, *31*, 765–783.
34. Greff, K.; Srivastava, R.K.; Koutník, J.; Steunebrink, B.R.; Schmidhuber, J. LSTM: A search space odyssey. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 2222–2232.
35. Brahma, D.K.R. *Prediction of Book Availability in the Library Management System Using Machine Learning*; Central institute of technology: Perth, Australia. 2019.
36. Wang, J.; Song, G. A deep spatial-temporal ensemble model for air quality prediction. *Neurocomputing* **2018**, *314*, 198–206.
37. Nakisa, B.; Rastgoo, M.N.; Rakotonirainy, A.; Maire, F.; Chandran, V. Long short term memory hyperparameter optimisation for a neural network based emotion recognition framework. *IEEE Access* **2018**, *6*, 49325–49338.
38. Yu, C.; Qi, X.; Ma, H.; He, X.; Wang, C.; Zhao, Y. LLR: Learning learning rates by LSTM for training neural networks. *Neurocomputing.* **2020**, *394*, 41–50.
39. Benkedjouh, T.; Medjaher, K.; Zerhouni, N.; Rechak, S. Remaining useful life estimation based on non-linear feature reduction and support vector regression. *Eng. Appl. Artif. Intell.* **2016**, *26*, 1751–1760.
40. Fallah, S.N.; Ganjkhani, M.; Shamshirband, S.; Chau, K.-w. Computational Intelligence on Short-Term Load Forecasting: A Methodological Overview. Energies 2019, 12, 393.
41. Afshin, M.; Sadeghian, A. PCA-based least squares support vector machines in week-ahead load forecasting. In Proceedings of the 2007 IEEE/IAS Industrial & Commercial Power Systems Technical Conference, Edmonton, AB, Canada, 6–11 May 2007; pp. 1–6.
42. Lisi, F.; Shah, I. Forecasting next-day electricity demand and prices based on functional models. *Energy Syst.* **2020**, *11*, 947–979.
43. Cherkassky, V.; Ma, Y. Selection of meta-parameters for support vector regression. In Proceedings of the International Conference on Artificial Neural Networks, Madrid, Spain, 28–30 August 2020; pp. 687–693.
44. Fabbri, A.G.; Chung, C.-J. On blind tests and spatial prediction models. *Nat. Resour. Res.* **2008**, *17*, 107–118.
45. Dahl, M.; Brun, A.; Kirsebom, O.S.; Andresen, G.B. Improving short-term heat load forecasts with calendar and holiday data. *Energies* **2018**, *11*, 1678.