sustainability

MDPI

Article

RAID: Robust and Interpretable Daily Peak Load Forecasting via Multiple Deep Neural Networks and Shapley Values

Joohyun Jang [1,†], Woonyoung Jeong [1,†], Sangmin Kim [1], Byeongcheon Lee [1], Miyoung Lee [2] and Jihoon Moon [1,*]

[1] Department of AI and Big Data, Soonchunhyang University, Asan 31538, Republic of Korea
[2] Department of Software, Sejong University, Seoul 05006, Republic of Korea
* Correspondence: jmoon22@sch.ac.kr; Tel.: +82-41-530-4956
† These authors contributed equally to this work.

Abstract: Accurate daily peak load forecasting (DPLF) is crucial for informed decision-making in energy management. Deep neural networks (DNNs) are particularly apt for DPLF because they can analyze multiple factors, such as timestamps, weather conditions, and historical electric loads. Interpretability of machine learning models is essential for ensuring stakeholders understand and trust the decision-making process. We proposed the RAID (robust and interpretable DPLF) model, which enhances DPLF accuracy by recognizing daily peak load patterns and building separate DNN models for each day of the week. This approach was accessible for energy providers with limited computational resources, as the DNN models could be configured without a graphics processing unit (GPU). We utilized scikit-learn's MLPRegressor for streamlined implementation, Optuna for hyperparameter optimization, and the Shapley additive explanations (SHAP) method to ensure interpretability. Applied to a dataset from two commercial office buildings in Richland, Washington, RAID outperformed existing methods like recurrent neural networks, Cubist, and HYTREM, achieving the lowest mean absolute percentage error values: 14.67% for Building 1 and 12.74% for Building 2. The kernel SHAP method revealed the influence of the previous day's peak load and temperature-related variables on the prediction. The RAID model substantially improved energy management through enhanced DPLF accuracy, outperforming competing methods, providing a GPU-free configuration, and ensuring interpretable decision-making, with the potential to influence energy providers' choices and promote overall energy system sustainability.

Keywords: daily peak load forecasting; deep neural network; hyperparameter optimization; Optuna; explainable artificial intelligence; Shapley additive explanations

Citation: Jang, J.; Jeong, W.; Kim, S.; Lee, B.; Lee, M.; Moon, J. RAID: Robust and Interpretable Daily Peak Load Forecasting via Multiple Deep Neural Networks and Shapley Values. *Sustainability* **2023**, *15*, 6951. https://doi.org/10.3390/su15086951

Academic Editor: Seung-Hoon Yoo

Received: 20 February 2023
Revised: 17 April 2023
Accepted: 19 April 2023
Published: 20 April 2023

1. Introduction

Energy management systems (EMSs) play a crucial role in promoting sustainability by reducing energy consumption and greenhouse gas emissions [1,2]. These systems monitor and control energy use in buildings and facilities, collecting data on energy usage to identify areas for improvement and make changes for energy savings [2]. For instance, EMSs can adjust lighting levels based on occupancy or natural light and control heating, ventilation, and air conditioning systems [3]. By reducing energy waste, EMSs help lower carbon emissions, contributing significantly to climate change mitigation [4]. Furthermore, EMSs help organizations save money by reducing energy costs and improving operational efficiency [5,6]. Through real-time energy usage data, organizations can make informed decisions to allocate resources for maximum energy savings and cost reduction [6]. Essentially, EMSs are crucial from a sustainability viewpoint, reducing energy consumption and carbon emissions, saving costs, and improving operational efficiency. Implementing an effective energy management strategy helps organizations promote sustainability and protect the environment for future generations [7].

Daily peak load forecasting (DPLF) is a vital component of an EMS, providing crucial information on future energy demand to ensure a steady energy supply [8]. The information DPLF provides plays a key role in the planning and management of energy resources, and also in the design and operation of energy systems [9]. For energy providers, DPLF helps in effective resource allocation, allowing increased energy production during periods of high demand and reduced during low demand. Accurate forecasting also enables providers to make informed decisions on the optimal mix of renewable and non-renewable energy sources for a reliable and sustainable energy supply [10]. For consumers, DPLF helps reduce energy costs by managing energy use during high-demand periods. By shifting energy-intensive activities to low-demand periods, consumers can lower energy costs and contribute to a more sustainable energy system [11]. Because accurate forecasting is critical for ensuring a reliable and sustainable energy system for the future, DPLF is crucial in the EMS, enabling effective energy demand and supply management, reducing costs, and promoting sustainability [8,9].

Despite the limited nature of daily peak load data, machine learning (ML) and deep learning (DL) techniques can still play a crucial role in achieving accurate DPLF [12]. These techniques offer immense value by analyzing and predicting energy demand based on multiple factors, not just the daily peak load data [13]. For instance, ML algorithms can be trained to recognize energy demand patterns influenced by weather conditions, energy usage habits, time of day, and other relevant data, leading to a more comprehensive understanding of energy demand and more accurate predictions. DL techniques, like artificial neural networks (ANNs), can analyze multiple inputs and underlying relationships, providing an in-depth understanding of energy demand patterns [14,15]. This results in even more accurate predictions and better decision-making by energy providers. Therefore, while daily peak load data may seem limited, combining it with other relevant data can provide valuable insights for improving DPLF accuracy. ML and DL techniques enable energy providers to analyze this combined data, identify patterns and trends, and make informed decisions for a more reliable and sustainable energy system [12,13].

The paper proposes a solution to enhance DPLF's accuracy through a robust and interpretable model called RAID as illustrated in Figure 1, which serves as the graphical abstract. This approach recognizes that the daily peak load pattern differs for each weekday and builds separate deep neural network (DNN) models for each, allowing for more accurate predictions. DNN models can be configured without a graphics processing unit (GPU), making the approach accessible for energy providers with limited computational resources. Moreover, the use of the Shapley additive explanations (SHAP) method provides a transparent understanding of the model's decision-making process, addressing the interpretability issue often referred to as the "black box" problem [16,17]. Essentially, the RAID approach offers a valuable contribution to the field of energy management by improving the accuracy of DPLF through separate DNN models, a GPU-free configuration, and interpretable decision-making.

This paper's key contributions are:

- We develop a DPLF model using two publicly available datasets, demonstrating improved performance compared to existing models in terms of the mean absolute percentage error (MAPE).
- We construct a DNN model with optimal hyperparameter values set using the Optuna optimization library, resulting in high prediction accuracy even in a central processing unit (CPU) environment.
- We utilize SHAP to provide interpretable insights into the model's behavior, helping EMS managers make informed decisions and highlighting the importance of using ML techniques in energy management.

The remainder of this study is as follows. Section 2 reviews related works in DPLF model construction. Sections 3 and 4 focus on the data preprocessing and construction steps involved in the RAID model, respectively. Section 5 presents our approach's experimental results. Section 6 discusses the limitations of the RAID model's performance and efficiency

under various conditions. Finally, Section 7 provides our conclusions and suggestions for future work in this area.

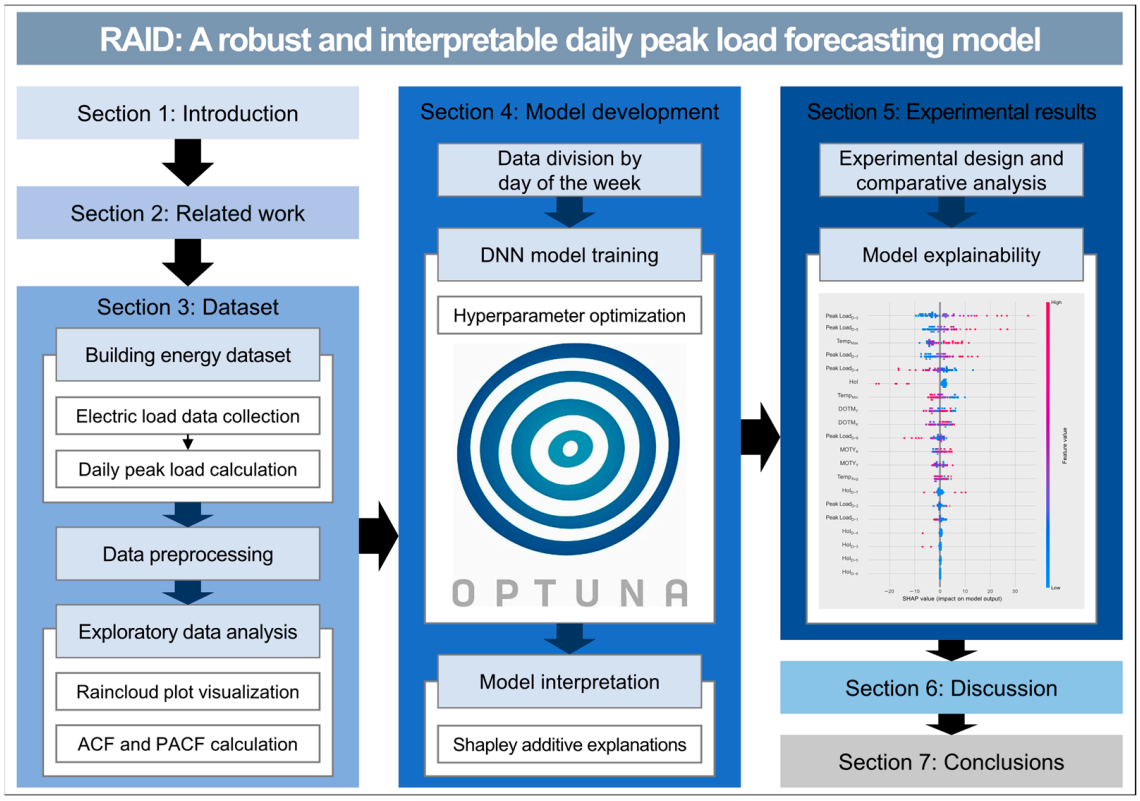


Figure 1. Graphical abstract for a visual overview of our study.

2. Related Work

Given the complex nature of energy consumption and its correlation with historical and current data, accurate DPLF remains a challenging issue. The significance of finding a solution has encouraged researchers to investigate different ways to enhance DPLF accuracy. As such, many studies have been conducted to achieve accurate DPLF based on ML and DL techniques, as discussed in Table 1 [18–26].

Several studies have been conducted targeting regions with large-scale power consumption in the order of megawatts or higher. Son et al. [18] developed a neuro-fuzzy-based DPLF model for power system management using Korea Power Exchange data. They divided the load data into seven subsets based on season and day type and selected optimal features through the non-overlap area distribution measurement method. The model achieved a mean absolute percentage error (MAPE) of 1.86% over one year. Yu et al. [19] introduced a forecasting algorithm combining shape-based dynamic time warping with a gated recurrent neural network for accurate DPLF using the EUNITE dataset. By incorporating some-hot encoding for calendar information, the algorithm significantly improved daily peak load forecasting accuracy compared to other algorithms on the same dataset. Ibrahim and Rabelo [20] evaluated deep learning methods, particularly convolutional neural networks (CNNs) and long short-term memory (LSTMs), for predicting peak demand in Panama's power system. They discovered that CNNs outperformed LSTMs, with the multivariate CNN achieving an accuracy of 0.92, a mean square error of 1271.65, a MAPE of 1.62%, and a mean absolute error of 27.86. Lastly, Lee [21] investigated the DPLF problem in South Korea, proposing multiple linear regression-based methods, including summer, winter, and all-season [22] models. These models, selected based on daily peak load characteristics, outperformed existing methods like extreme gradient boosting, with

mean absolute percentage errors of 1.95%. The proposed methods demonstrated seasonally adaptive characteristics for improved forecasting performance.

Table 1. Summary of related studies for machine learning-based DPLF.

Authors	Purpose	Target or Dataset	Applied Techniques	Performance Metrics
Son et al. [18]	• DPLF	Korea Power Exchange	• Non-overlap area distribution measurement • Neural network with weighted fuzzy membership functions	• MAPE: 1.86
Yu et al. [19]	• DPLF	EUNITE	• Dynamic time warping • GRU	• MAPE: 1.01
Ibrahim and Rabelo [20]	• DPLF	Panama's power system	• Multivariate CNN (the best performing model) • Multivariate CNN-LSTM • Multihead CNN • LSTM	• R^2: 0.92 (Multivariate CNN) • MSE: 1271.65 (Multivariate CNN) • MAPE: 1.62 (Multivariate CNN) • MAE: 27.86 (Multivariate CNN)
Lee et al. [21]	• DPLF	South Korea's daily peak loads	• Multiple linear regression	• MAPE: 1.93–1.96
Kim et al. [23]	• DPLF	Two small industrial facilities	• Bagging • RF • Extra trees • AdaBoost • Gradient boosting machine	• R^2: 0.73 (Facility 1), 0.57 (Facility 2) • RMSE: 0.10 (Facility 1), 0.09 (Facility 2) • MAPE: 23.49 (Facility 1), 23.24 (Facility 2)
Kim et al. [24]	• DPLF	Non-residential building	• Residual LSTM	• MAE: 6.86 • MAPE: 11.7 • RMSE: 10.5
Saxena et al. [25]	• Peak electric load days forecasting	RIT	• Autoregressive integrated moving average • Logistic regression • ANN	• Accuracy: 86% • Inaccuracy: 14% • False positive: 9%
Liu and Brown [26]	• Day-ahead daily peak time forecasting	IESO	• Naïve Bayes • Support vector machine • RF • AdaBoost • CNN • LSTM • Stacked autoencoder	• Precision: 0.18–0.68 • Recall: 0.69–0.97 • Accuracy: 0.82–0.98
Our study	• DPLF with inter-pretability	Two commercial office buildings	• ANN with two hidden layers • Optuna • Shapley additive explanations	• MAPE: 14.67 (Building 1), 12.74 (Building 2)

AdaBoost, adaptive boosting; ANN, artificial neural network; CNN, convolutional neural network; DPLF, daily peak load forecasting; EUNITE, European Network on Intelligent Technologies; GRU, gated recurrent unit; LSTM, long short-term memory; MAE, mean absolute error; MSE, mean squared error; MAPE, mean absolute percentage error; RF, random forest; RIT, Rochester Institute of Technology; IESO, Independent Electricity System Operator.

In addition to large-scale power consumption regions, researchers have also focused on DPLF studies targeting smaller areas, such as individual buildings with power usage in the kilowatt range. Kim et al. [23] proposed three ideas to enhance load prediction in industrial facilities: a new feature selection method, an aggregated model of ensemble models, and a modified isolation forest for predictive outlier detection and compensation. These implementations improved the system's security and reliability by increasing R2 scores and reducing peak load underestimations. Kim et al. [24] introduced a residual LSTM model to boost electricity demand forecasting accuracy for both peak and total demand. The model combined LSTM and residual blocks to minimize forecasting residuals. They evaluated the model using electricity-demand data from a non-residential building in Seoul. They found that it outperformed existing deep learning-based prediction methods like DNN, LSTM, CNN-LSTM, and recurrent inception CNN regarding peak demand and overall DPLF accuracy.

Not only the researchers mentioned earlier, but also our team has conducted numerous studies to achieve accurate DPLF at the individual building level. Moon et al. [2] developed a Cubist-based incremental learning model for precise and interpretable DPLF and total daily load forecasting. They identified critical factors affecting electrical load forecasting by employing time-series cross-validation and variable importance analysis. The model demonstrated high accuracy when tested with five building datasets with low MAPE and coefficient of variation of root mean square error (CVRMSE) values. The study also highlighted the importance of temperature, holiday information, and previous electrical loads in forecasting. Furthermore, Moon et al. [8] introduced a hybrid tree-based ensemble learning model called HYTREM for DPLF. They preprocessed two publicly available building energy datasets and employed tree-based ensemble learning methods, such as gradient boosting machine (GBM), extreme gradient boosting (XGBoost), Cubist, and random forest (RF), for daily peak load prediction. The Boruta algorithm was used for feature selection, and a time series cross-validation (TSCV)-based online RF model was built for robust day-ahead DPLF. The HYTREM model outperformed RNN and other ensemble learning-based forecasting models regarding MAPE and CVRMSE.

In addition to DPLF, studies focus on predicting specific peak load days, such as peak electric load days (PELDs) forecasting and day-ahead daily peak time forecasting, contributing to better preparedness for peak demand. Saxena et al. [25] developed a hybrid model for forecasting PELDs by combining auto-regressive integrated moving averages (ARIMA), logistic regression, and ANN models. This model was tested on energy data from the Rochester Institute of Technology and demonstrated higher accuracy, lower inaccuracy, and fewer false positives than individual base models. The authors estimated that if the model had been used for demand response, it could have saved up to $80,000 in one year. Liu and Brown [26] aimed to predict the time of daily load peaks 24 h beforehand in Ontario, Canada. Using historical power demand and weather data, they constructed classification models for winter and summer seasons. Several algorithms, including Naïve Bayes, support vector machine (SVM), RF, adaptive boosting (AdaBoost), CNN, LSTM, and stacked autoencoder, were applied and compared in terms of accuracy, precision, and recall. The LSTM technique performed best, and the models showed promising results in effectively locating the daily load peak hour.

In conclusion, tackling the challenges of DPLF at the building level (kW scale) has proven to be considerably more complex than at the regional level (MW scale). Despite the impressive strides made by previous research, there remains ample room for enhancing prediction accuracy and model interpretability. Our innovative RAID model rises to the occasion, providing an accurate, interpretable, and accessible solution designed explicitly for low-performance computing environments and publicly available datasets. By thoroughly comparing Lee's all-season model and our previous Cubist and HYTREM models, we establish the undeniable robustness of the RAID model for DPLF. This groundbreaking research propels prediction performance to new heights and offers energy managers a comprehensive and transparent tool for understanding and managing peak loads at the

building level. We are one step closer to a more sustainable and efficient energy future by fostering better decision-making and resource allocation in the energy sector, ultimately benefiting society.

3. Dataset

Section 3 describes the dataset used for building the RAID model, which involves preprocessing the data to configure the input variables using timestamps, weather, and historical load information. Section 3.1 presents an overview of the descriptive statistics of the daily peak load data of Buildings 1 and 2. Meanwhile, Section 3.2 discusses the data preprocessing, including the input variables used in the RAID model. Finally, Section 3.3 presents exploratory data analysis, including monthly and daily peak load distribution by day of the week and autocorrelation and partial autocorrelation functions.

3.1. Building Energy Dataset

In this section, we describe the preprocessing of data for building a forecasting model which involves configuring the input variables using timestamps, weather, and historical load information. In addition, we provide an overview of the descriptive statistics of the daily peak load data for Building 1 and Building 2, which enables us to gain insights into the energy consumption patterns of the commercial office buildings. The dataset includes publicly available data, including timestamp, meteorological, and historical information, from two commercial office buildings located in Richland, Washington and covers a three-year period from 2 January 2009, to 31 December 2011 [27]. While the hourly electrical load data for each day comprises 24 rows, the daily peak electrical load data are represented by a single row for each day. For each day in the dataset, containing 24 h of electrical load data, we calculated the highest electrical load value for use in the DPLF model. Table 2 presents a comprehensive overview of the descriptive statistics of the daily peak load data of Building 1 and Building 2.

The descriptive statistics were calculated using methods such as the mean, median, standard deviation, and range. The statistics reveal that Building 2 has a marginally higher mean value than Building 1. Moreover, the standard deviation values for both buildings are similar, implying that the variable's values are dispersed around the mean value almost equally in both buildings. The median value for Building 2 is slightly higher than Building 1. The range, the difference between the maximum and minimum values, is more substantial for Building 1 than for Building 2, indicating more significant variability of the variable values in Building 1. Interestingly, the skewness values for both buildings are close to zero, implying that the distribution of the values of the variable is relatively symmetrical. The kurtosis values are also near zero, indicating that the values of the variable are similar to those of a normal distribution. Finally, the small standard error values suggest that the sample mean values are highly representative of the actual population mean values.

Table 2. Statistical analysis of daily peak load data (units: kW).

Statistics	Building 1	Building 2
Number of valid cases	1094	1094
Mean	49.16	54.46
Standard deviation	21.69	21.15
Trimmed mean	50.40	56.23
Median	48.59	54.52
Median absolute deviation	19.87	18.52
Minimum	8.86	10.97
Maximum	141.11	135.00
Range	132.25	124.03
Skew	0.34	0.05
Kurtosis	0.43	0.09
Standard error	0.66	0.64

3.2. Data Preprocessing

This information was important for our goal of improving the DPLF's accuracy compared to previous studies [2,8], achieved by using the same input variables as previous studies as described in Table 3.

Table 3. List of input variables for daily peak load forecasting.

IV	Input Variable	Variable Type	Data Type
01	$MOTY_X$	Continuous [–1, 1]	Timestamp
02	$MOTY_Y$	Continuous [–1, 1]	Timestamp
03	$DOTM_X$	Continuous [−1, 1]	Timestamp
04	$DOTM_Y$	Continuous [−1, 1]	Timestamp
05	$DOTW_X$	Continuous [−1, 1]	Timestamp
06	$DOTW_Y$	Continuous [−1, 1]	Timestamp
07	Hol	Binary	Timestamp
08	$Temp_{Min}$	Continuous	Meteorological
09	$Temp_{Avg}$	Continuous	Meteorological
10	$Temp_{Max}$	Continuous	Meteorological
11	Hol_{D-7}	Binary	Historical
12	Peak $Load_{D-7}$	Continuous	Historical
13	Hol_{D-6}	Binary	Historical
14	Peak $Load_{D-6}$	Continuous	Historical
15	Hol_{D-5}	Binary	Historical
16	Peak $Load_{D-5}$	Continuous	Historical
17	Hol_{D-4}	Binary	Historical
18	Peak $Load_{D-4}$	Continuous	Historical
19	Hol_{D-3}	Binary	Historical
20	Peak $Load_{D-3}$	Continuous	Historical
21	Hol_{D-2}	Binary	Historical
22	Peak $Load_{D-2}$	Continuous	Historical
23	Hol_{D-1}	Binary	Historical
24	Peak $Load_{D-1}$	Continuous	Historical

MOTY, month of the year; DOTM, day of the month; DOTW, day of the week; Hol, holiday; Temp, temperature; Min, minimum; AVG, average; Max, maximum.

Electricity consumption is closely connected to the time it is being used. So, when we were trying to understand how electricity was used, we considered all the time-related factors, such as the month, day, and day of the week. However, using the timestamp data to predict daily peak load were needed to give us more information because these data are in a sequence format and does not consider any repeating patterns. To overcome this problem, we transformed the timestamp data into a two-dimensional (2D) format using Equations (1)–(6), allowing us to capture the patterns in the data more accurately [2,27].

$$MOTY_X = \sin(360^\circ/12 \times Month), \quad (1)$$

$$MOTY_Y = \cos(360^\circ/12 \times Month), \quad (2)$$

$$DOTM_X = \sin(360^\circ/LDM_{Month} \times Day), \quad (3)$$

$$DOTM_Y = \cos(360^\circ/LDM_{Month} \times Day), \quad (4)$$

$$DOTW_X = \sin(360^\circ/7 \times Day\ of\ the\ Week), \quad (5)$$

$$DOTW_Y = \cos(360^\circ/7 \times Day\ of\ the\ Week), \quad (6)$$

where *MOTY*, *DOTM*, and *DOTW* refer to the month of the year, day of the month, and day of the week that the prediction time point belongs to, respectively. LDM_{Month} refers to the last day of the month that a particular day belongs to. For example, if it is February, the LDM_{Month} is 28 or 29. If it is March, the LDM_{Month} is 31. This information is helpful in determining which month a particular day belongs to and how many days that month has.

To verify the validity and applicability of the 2D representation, we measured several regression statistics on the electrical loads in both 1D space (months, days, and days of the week) and 2D space. In Table 4, the comparison of the regression statistics between the 1D and 2D spaces provides evidence that the 2D representation improves our understanding of the relationships between time factors and electrical load trends.

Table 4. Regression statistics comparison between 1D and 2D time representations.

Statistics	Building 1		Building 2	
	1D Space	2D Space	1D Space	2D Space
Residual standard error	17.7	17.1	17.4	16.5
Multiple R-squared	0.339	0.386	0.326	0.397
Adjusted R-squared	0.337	0.382	0.324	0.394

For Building 1, the residual standard error decreased from 17.7 in the 1D space to 17.1 in the 2D space. The multiple R-squared value increased from 0.339 to 0.386, and the adjusted R-squared value also increased from 0.337 to 0.382. These improvements indicate that the 2D representation better fits the data and better accounts for the variability in electrical loads. Similarly, for Building 2, the residual standard error decreased from 17.4 in the 1D space to 16.5 in the 2D space. The multiple R-squared value increased from 0.326 to 0.397, and the adjusted R-squared value increased from 0.324 to 0.394. Again, these improvements demonstrate the effectiveness of the 2D representation in capturing the relationships between time factors and electrical load trends. Hence, the two-dimensional representation, as demonstrated by the improved regression statistics for both buildings, enhances our ability to predict electrical loads by more accurately accounting for the periodic patterns present in the data. Based on these findings, we used six input variables provided by Equations (1)–(6) to describe the date and time of the prediction time points.

The daily peak load patterns in different types of buildings can change on weekdays and holidays. To consider this, we included a "holiday indicator" in our model, using information about public holidays from the Time and Date database [28]. Holidays also included weekends (Saturday and Sunday). We used a one-hot encoding method to make the data more accessible, meaning if a day was a holiday, it was labeled as "1", otherwise it was labeled as "0". With this information, we used seven different time-related variables as inputs for making predictions about daily peak load.

The use of electrical heating and air conditioners increases the daily peak load during the winter and summer, respectively [29]. The most crucial factor affecting daily peak load is temperature, so we focused on temperature-related information. The collected data included information about hourly temperature, in Fahrenheit, and daily peak load, so we calculated both buildings' daily highest, average, and lowest temperatures. As such, we could consider temperature's influence on daily peak load.

The history of daily peak loads is crucial for forecasting future daily peak loads because it shows how consumption has changed. We used the daily peak load of the last seven days to make accurate predictions for the prediction model's input variables, allowing them to reflect current trends in consumption effectively. In addition, daily peak load on holidays and weekdays can be different, so we included a holiday indicator in the daily peak load of the last seven days to show whether the day being predicted was a holiday [2,8]. In total, the prediction model was built using 24 input variables.

3.3. Exploratory Data Analysis

Figure 2 shows raincloud plots of the monthly distribution of daily peak load data for Buildings 1 and 2, revealing that each building has a different distribution of usage for each month. Summer and winter months, such as July, August, and December in WA, USA, have a relatively high daily peak load compared to other months. Time series data can have various characteristics, including seasonal patterns, and incorporating these into a time series prediction model can lead to improved performance [30].

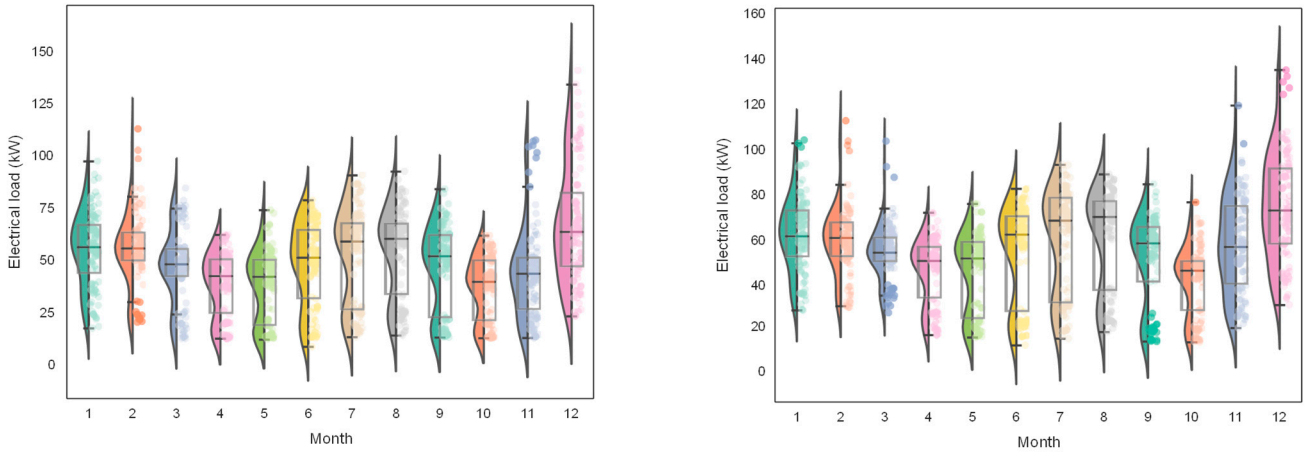


Figure 2. Distribution of the daily peak load by month. (**a**) Building 1; (**b**) Building 2.

We found that time series forecasts have different characteristics based on temporal factors, as shown by the monthly data distribution raincloud plots in Figure 2. Figure 3 is a raincloud displaying the daily peak load data distribution by day of the week for Buildings 1 and 2. It is typical for commercial establishments to not use electricity on weekends, resulting in lower daily peak loads. This is reflected in the box plot, showing significantly lower daily peak loads on weekends compared to weekdays. The data's characteristics were better reflected when the data was divided by the day of the week instead of the month. Therefore, it would be more appropriate to approach this data by classifying it by day of the week and building a predictive model accordingly.

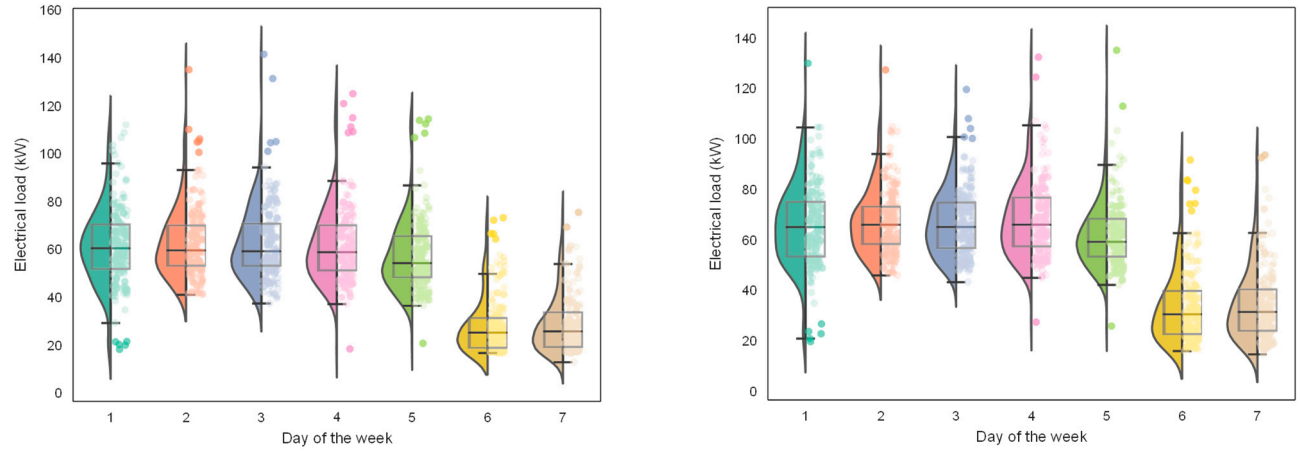


Figure 3. Distribution of the daily peak load by day of the week. (**a**) Building 1; (**b**) Building 2.

To substantiate the validity of our proposed method, we employed the autocorrelation function (ACF) and partial autocorrelation function (PACF). Autocorrelation, to begin with,

refers to the correlation between different time points of a given series, representing a lag-based correlation. For instance, it signifies the correlation between data at time t and $t-1$. Naturally, as the lag increases, the correlation is bound to diminish. However, when evaluating stationarity, a high correlation is observed even as the lag expands, suggesting a lack of stationarity.

Intuitively, autocorrelation indicates the extent to which the next time point can be predicted based on the previous one. The ACF measures the correlation between a time series and its lagged values up to a specified lag order. The ACF at lag k is computed as the correlation coefficient between the original time series and the same time series delayed by k time units. Before calculating the ACF, it is essential to preprocess the time series to ensure its stationarity. The process typically involves differencing, trend removal, or seasonal adjustment, following Equation (7).

$$ACF_k = (\sum(Y_t - \mu) \times (Y_{t-k} - \mu))/(\sum(Y_t - \mu))^2, \quad (7)$$

where Y_t represents the time series data at time t, μ is the mean of the time series, and the summations run over all possible values of t for which both Y_t and Y_{t-k} are defined.

We utilized the *acf*() method from Python's statsmodel library [31], which takes the time series as input and the number of lags to be computed. The output displays a plot of the ACF values at each lag. Figure 4 shows the autocorrelation of the daily peak load of Buildings 1 and 2. The ACF reveals a strong relationship between the two data sets. The degree of autocorrelation changes over time, suggesting that the data has different patterns for each day of the week. To make more accurate predictions, it would be better to categorize the data by day of the week rather than using a general approach to the whole dataset.

Similar to the ACF, the PACF measures the correlation between Y_t and Y_{t+k} as shown in Equation (8) for the PACF at lag k:

$$PACF_k = Corr(e_t, e_{t-k}) = Corr(Y_t - \beta_1 \times Y_{t-1} - \ldots - \beta_{k-1} \times Y_{t-k+1}, Y_{t-k}), \quad (8)$$

here Y_t represents the time series data at time t, and β_i represents the coefficients for the lagged values Y_{t-i}. The PACF at lag k measures the correlation between Y_t and Y_{t-k} after accounting for the influence of other lagged values, Y_{t-1} through Y_{t-k+1}.

The PACF calculation excludes the influence of other Y values between t and $t+k$. In this context, e_t represents the residual after accounting for the influence of Y_{t-1} through Y_{t-k-1} and removing the effect of Y_{t-k}, and the same applies to e_{t-k}. Therefore, calculating $Corr(e_t, e_{t-k})$ exclusively examines the relationship between Y_t and Y_{t+k}. Before computing the PACF, it is necessary to perform preprocessing steps, such as differencing and removing trends or seasonality from the data, to ensure the time series is stationary. To calculate the PACF, one can utilize the *pacf*() method from Python's statsmodels library [31], which offers a convenient and accurate approach for obtaining partial autocorrelations at different lag orders.

Figure 5 shows the partial autocorrelation of the daily peak load for Buildings 1 and 2. From Figure 5, there is no autocorrelation as the time lag increases. However, there are significant autocorrelations between some time points. Although the correlation between each time point is not uniform, we confirmed some autocorrelation concerning time point t exists, meaning that the time points from $t-1$ to $t-7$ affect the time point t. Based on this, it would be appropriate to consider data from the seven days prior to this dataset's time point of prediction.

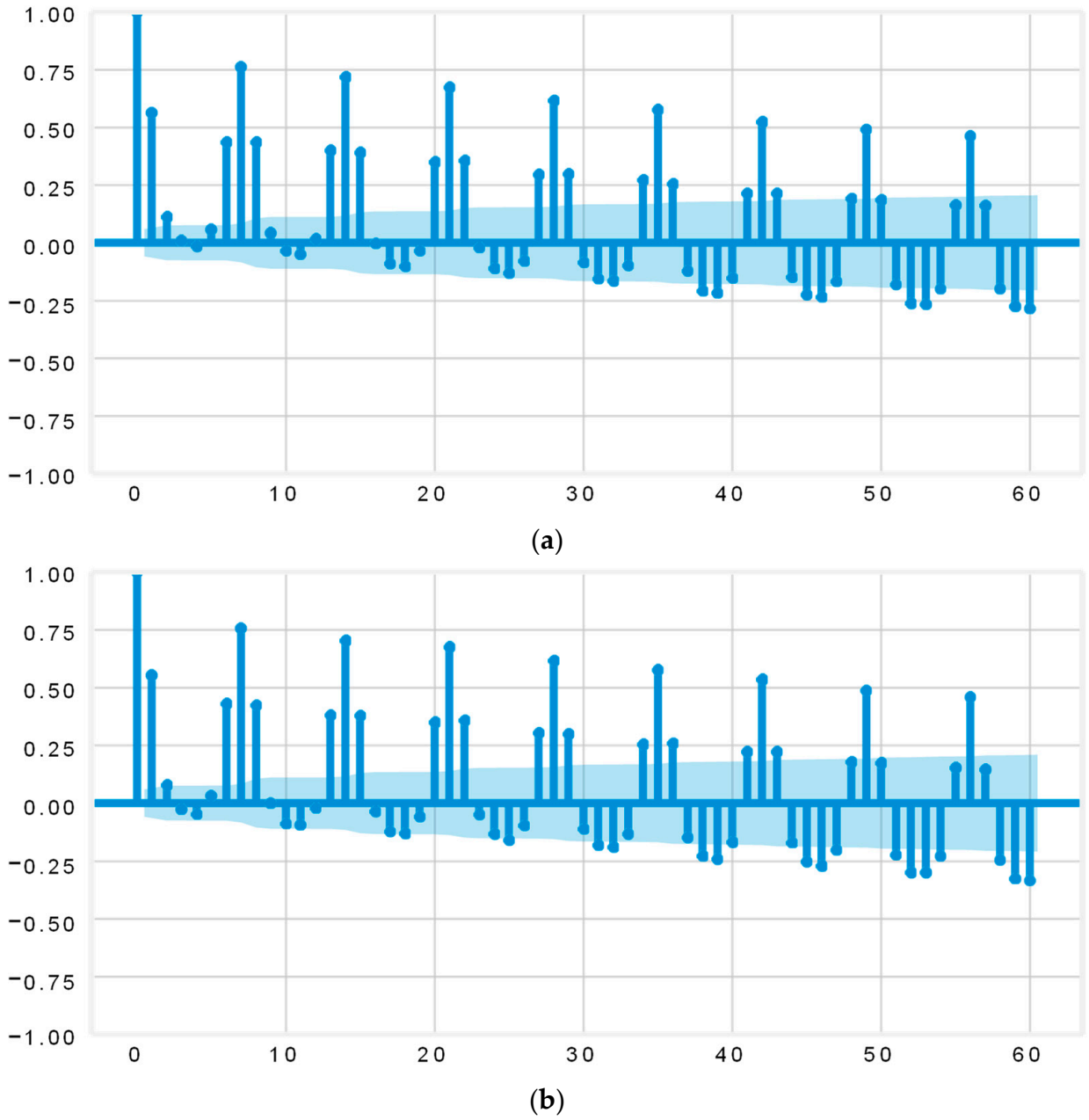


Figure 4. Autocorrelation plot. (**a**) Building 1; (**b**) Building 2.

To make sure that the data used to train the DL-based DPLF model are consistent, we had to normalize the different input variables. Normalization helps to make all input variables comparable by transforming them into a scale between 0 and 1. This process is called min-max normalization or feature scaling. We used Equation (9) by finding the minimum and maximum values in the data and then dividing each value by the range to get a value between 0 and 1. This helps the model train more accurately and overcome any imperfections in the data.

$$z = (x - \text{Min}(x))/(\text{Max}(x) - \text{Min}(x)), \quad (9)$$

here z is the new value of a number in the set of observed values x after normalization. Min(x) and Max(x) are the minimum and maximum values of x.

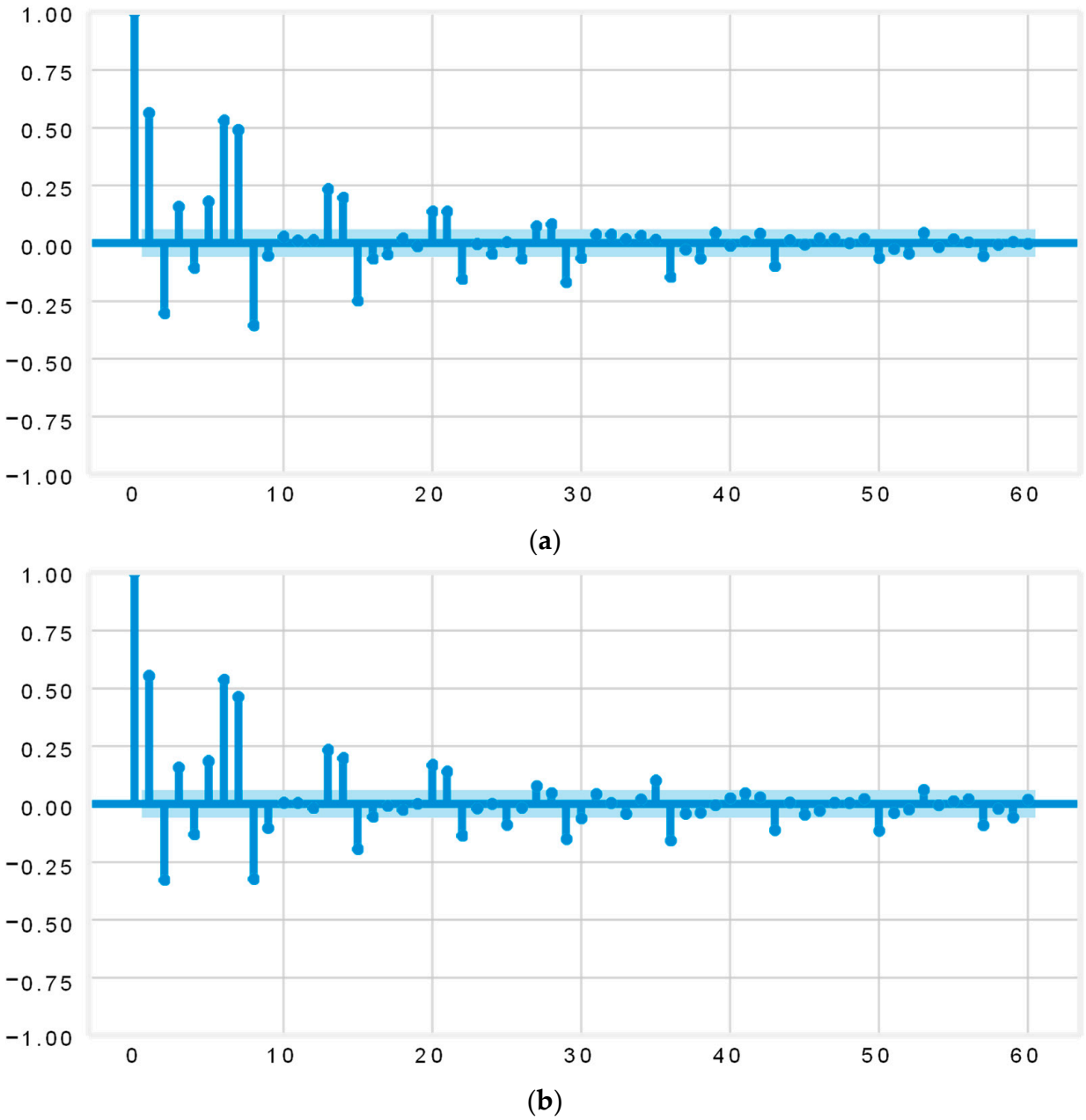


Figure 5. Partial autocorrelation plot. (**a**) Building 1; (**b**) Building 2.

4. Model Development

We configured a RAID model following a process illustrated in Figure 6. We divided the data into groups based on the day of the week to make the analysis more manageable. For our analysis, we used a few different techniques to improve the model's performance. First, we used a multilayer perceptron (MLP) for feature learning. We also used Optuna to optimize the hyperparameters. Finally, we used a tool called SHAP to interpret the model's results, providing the variables that determine how the model operates or insight into how the model arrived at its predictions. Further details on these techniques are explained later in the study.

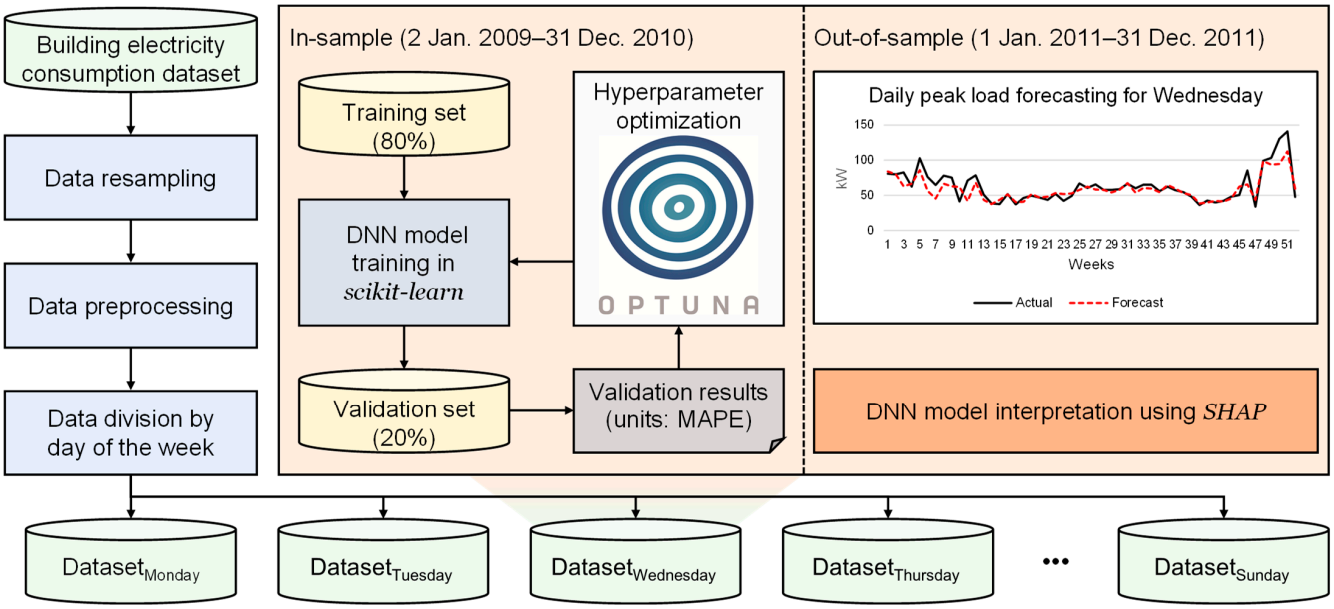


Figure 6. Architecture of the RAID model (MAPE, mean absolute percentage error; SHAP, Shapley additive explanations).

4.1. Feature Learning

An MLP, also known as a typical ANN architecture, is a way for computers to learn by example [32,33]. It is like a map that helps the computer understand how to solve a problem. The MLP comprises individual nodes, called perceptrons, that are organized in different layers. There are three layers in an MLP: the input layer that receives information that the computer uses to make a prediction, the hidden layer where the computer extracts essential information from the input layer to help it make a better prediction, and the output layer which provides the final answer based on the information processed in the hidden layer. We determined the output values of the neurons in the hidden layer of an MLP using Equation (10), which considers the input data, connections between neurons (weights), and an activation function.

$$y_j = \varphi(\sum w_{ij} \times x_i + w_{i0}) \tag{10}$$

An MLP's input layer contains neurons, where each neuron is represented as x_i. Synaptic weights, represented as w_{ij}, model the connections between the neurons in the input and hidden layers. The activation function is represented by φ and determines the output of the hidden layer's neurons. The connections between the hidden and output layers are referred to as "weights" and play a crucial role in the processing and modeling of information. One advantage of MLPs is their ability to process information in parallel, with each neuron operating independently. This makes the system fault-tolerant, as it can still learn and process information even if some neurons malfunction [34].

Scikit-learn's MLPRegressor, part of a popular machine-learning library in Python called scikit-learn, is a DL model for regression tasks, precisely predicting numerical outputs based on input features [35]. MLPRegressor is limited to CPUs and cannot be run on GPUs because it is implemented as a feedforward neural network with a single hidden layer, which is unsuitable for parallel processing on GPUs [36]. Despite this, MLPRegressor can still provide a good performance on tabular datasets [37]. MLPRegressor's performance can be improved by adjusting the number of hidden units, the activation function in the hidden layer, the batch size, the learning rate schedule for weight updates, and the maximum number of iterations.

Hence, we used scikit-learn's MLPRegressor to build models for regression tasks that work well with tabular data and are limited to CPUs. The model's architecture is based on

an MLP. The hidden layer can impact the network's performance, so considering factors such as the number of layers and nodes and the activation function when setting it up is essential. In our case, we used an MLP with two hidden layers as we did not need many input variables for our prediction model. To improve its performance, we can customize different aspects of the MLP, such as the number of hidden units, the activation function, the solver algorithm, and the number of iterations.

4.2. Hyperparameter Optimization

Optuna is an open-source Python library that provides a unified interface for hyperparameter optimization, tuning hyperparameters in machine-learning models to improve their performance [38]. Hyperparameters are settings for an algorithm that must be set before training, as opposed to parameters that are learned during training. Examples of hyperparameters in a neural network include the learning rate, number of hidden layers, and regularization strength [39]. Optuna is designed to automate the search process for optimal hyperparameters. It provides a simple and flexible API that can be used to define a machine-learning model and a search space for hyperparameters. The library then uses various algorithms to search the hyperparameter space, such as random search, grid search, Bayesian optimization, and gradient-based optimization, to find the best set of hyperparameters that optimize a given metric, such as accuracy or F1 score.

One of Optuna's main features is its ability to handle the parallel execution of trials, which are individual evaluations of a model with different hyperparameter settings. This enables Optuna to search the hyperparameter space more efficiently and quickly. Another essential feature is its support for early stopping, allowing the optimization process to be terminated early if it determines that further trials are unlikely to improve the results, saving time and resources and also helping prevent overfitting. Optuna also provides a logging API to track the progress of the optimization process and the results of individual trials. This can be useful for analyzing the results and debugging the optimization process. In addition, Optuna has built-in integration with popular ML frameworks such as TensorFlow, PyTorch, and XGBoost, making it easy to use in existing workflows.

Hence, we used Optuna because it is a flexible, efficient, and an easy-to-use library for hyperparameter optimization that helps automate the search process for optimal hyperparameters so that we could focus on the ML model and not on the hyperparameter search process.

4.3. Model Interpretation

In the field of energy management, it can be challenging for EMS managers to understand complex AI models, mainly when they are considered "black boxes" [40,41]. The SHAP technique helps explain the impact of input variables on the output of an AI model [42,43]. It does so by calculating the Shapley value for each input variable, showing its influence on the dependent variable [43], helping to address the problem of AI models being seen as "black boxes" where it is difficult to understand why the model made a specific prediction. The calculation of the Shapley value is based on the concept of cooperative game theory, where the prediction made by the model is considered a cooperative game among the independent variables [44]. The Shapley value provides a fair way to allocate the contribution of each independent variable to the prediction made by the model.

The Shapley value provides a way to quantify the influence of each input variable in the model and can be expressed as either negative or positive values. This information helps us understand the relationship between the input and dependent variables. For example, let's say we have an ML model that predicts daily peak load as a dependent variable, and temperature is one of the input variables. If the Shapley value for temperature is positive, it means that as the temperature increases, the daily peak load also increases [40]. We

calculated the weight of the input variable and its contribution to the dependent variable using Equation (11) for the Shapley value.

$$\Phi_i = \sum_{S \subseteq F \setminus \{i\}} (|S|! \times (|F| - |S| - 1)!)/(|F|!) \times [f_{S \cup i}(x_{S \cup i}) - f_S(x_S)], \quad (11)$$

here F represents the set of all independent variables the model uses to make its prediction, f is the ML model itself, and i is a specific independent variable that we want to examine. The Φ_i value represents the contribution that the independent variable i makes to the model's overall prediction. We calculated the contribution as the difference between the prediction made by the model when variable i is present and when i is absent. We calculated the contribution as a weighted average that considers the number of cases in which the independent variables are present or absent. The weighting factors are based on the number of arrangements of the variables, with larger values for more combinations in which variable i is present or absent.

We aimed to understand how a DNN model, an MLP with more than two hidden layers [34,37], made its predictions for DPLF. To achieve this, we used a method called Kernel SHAP, a specific type of SHAP that can be used to interpret the predictions of any ML model [42]. It provides a straightforward way to understand how a model's inputs affect its outputs. We first divided the training set into different days of the week and found the optimal hyperparameters for each day via Optuna. After finding the optimal hyperparameters, we applied Kernel SHAP to the DNN model to explain its predictions, allowing them to see how input variables influenced the model's predictions. The results provided transparency into the model's decision-making process, which can be valuable for EMS managers who may not be experts in ML or DL.

5. Experimental Results

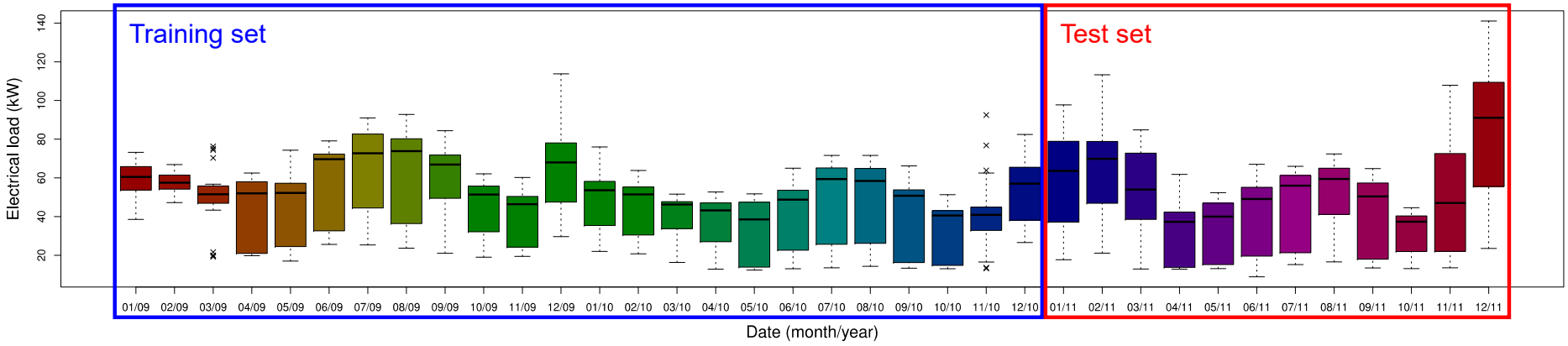
5.1. Experimental Design

To compare the DPLF models' performances, the MAPE, defined in Equation (12), was used. We calculated the MAPE by comparing the actual and predicted daily peak loads at a specific time (t) and the number of observations (n). In the formula, A_t represents the actual daily peak load, and P_t represents the predicted daily peak load.

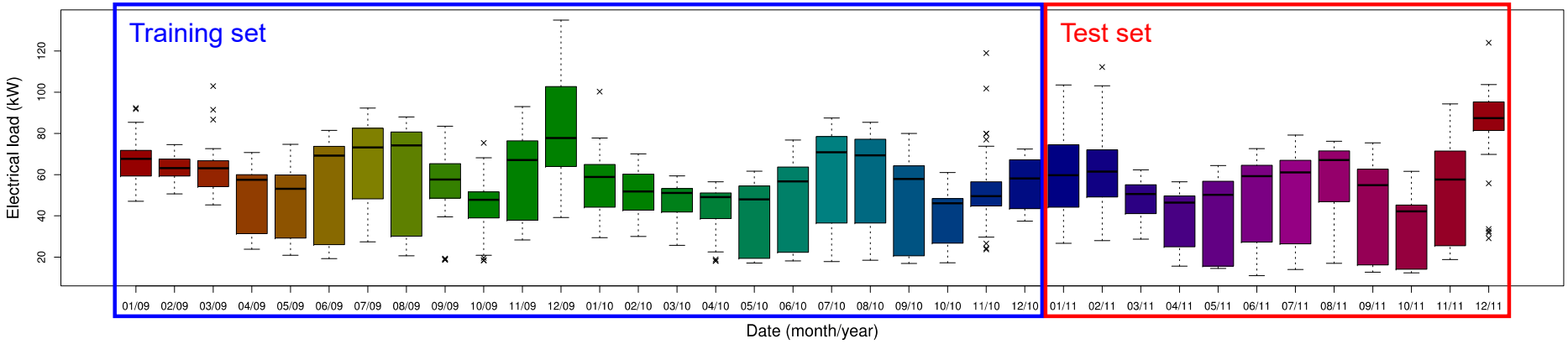
$$MAPE = (\sum |P_t - A_t| / A_t)/n \times 100 \quad (12)$$

As mentioned, we analyzed three years of daily peak load data for Buildings 1 and 2, from 2009 to 2011. We divided the data into two parts: two-thirds for training and one-third for testing. Figure 7 shows a graph displaying each month's daily peak load and how the data were split into training and testing sets. Additionally, we divided the data by day of the week. Table 5 shows the number of days belonging to each day of the week. It is important to note that Buildings 1 and 2 have the same values because the data were collected during the same period. We removed certain variables from the input variables for our model, depending on whether it was a weekday or a weekend. On weekdays, we removed $DOTW_X$ and $DOTW_Y$, and on weekends, we removed $DOTW_X$, $DOTW_Y$, Hol, and Hol_{D-7} since all variables had the same values.

The settings of DNN models, called hyperparameters, can impact their performance. To get the best results, adjusting these hyperparameters is essential. We used Optuna to determine the optimal hyperparameters for our DNN models based on the day of the week. We split the training data into 80% (January 2009–July 2010) for training the model and 20% (August 2010–December 2010) for evaluating its performance. We used the MAPE as the evaluation metric for the validation set, and we set the number of iterations in Optuna to 5000. Table 6 lists the hyperparameters for the DNN model and the range of values that were considered for each one. In order to ensure consistency in our results, we adopted the widely recognized "Adam" solver for weight optimization and set the random state to the commonly used value of 42 [37,45].



(**a**)



(**b**)

Figure 7. Box plots of the daily peak load and separation of training and testing sets. (**a**) Building 1; (**b**) Building 2.

Table 5. Number of days that belong to each day of the week.

Day of the Week	Training	Testing
Monday	104	52
Tuesday	104	52
Wednesday	104	52
Thursday	104	52
Friday	105	52
Saturday	104	53
Sunday	104	52
Total	729	365

Table 6. List of hyperparameters used to build optimal DNN models.

MLPRegressor's Hyperparameter	Definition	Range
hidden_layer_sizes (1)	Number of neurons in the first hidden layer	1–40 (Step: 1)
hidden_layer_sizes (2)	Number of neurons in the second hidden layer	1–40 (Step: 1)
activation	Activation function applied to the hidden layer	identity, tanh, relu
alpha	Intensity of the L2 regularization	0.0001–0.001 (Step: 0.0001)
batch_size	Batch size for stochastic optimization algorithms	5–100 (Step: 1)
learning_rate	Plan for updating the weights using a learning rate	constant, adaptive
learning_rate_init	Starting learning rate	0.0001–0.1 (Step: 0.0001)
max_iter	Highest number of iterations allowed	100–2000 (Step: 10)

5.2. Comparative Analysis

To demonstrate the effectiveness of the proposed method for efficient DPLF, we conducted comparative experiments using four different cases:

1. Training a DNN using the entire training set with default hyperparameter values;
2. Training a DNN using the entire training set and optimizing the hyperparameters using Optuna;
3. Training a DNN with default hyperparameter values using the training set for each day of the week; and
4. Our proposed method: training a DNN using the training set for each day of the week and optimizing the hyperparameters using Optuna.

Table 7 compares the MAPE of four different cases of DPLF for Buildings 1 and 2. The results show that the lowest MAPE values were achieved in Cases 2 and 4, with values of 17.03 and 14.67 for Building 1 and 13.33 and 12.74 for Building 2, respectively. Cases 1 and 3 had higher MAPE values of 23.22 and 52.29 for Building 1 and 21.97 and 54.92 for Building 2, respectively. These results demonstrate the effectiveness of optimizing the hyperparameters for DPLF. When dividing the daily peak load data by day of the week, the results showed that optimizing the hyperparameters of the DNN model was vital to achieving the best performance.

Table 7. Mean absolute percentage error (MAPE) comparison for different cases.

Case	Building 1	Building 2
1	23.22	21.97
2	17.03	13.33
3	52.29	54.92
4	14.67	12.74

The MAPE values of 14.67 and 12.74 for Buildings 1 and 2 were the lowest among the four cases considered. This indicates that considering the day of the week is essential when constructing the prediction model and adjusting its hyperparameters. The model's

performance was poor with default hyperparameters because of the limited data available for each day of the week. Nevertheless, with the optimization of hyperparameters, the model effectively learned the daily patterns and produced satisfactory results despite the small data set.

To compare the performance of our proposed model, RAID, with other well-performing RNN models in time series prediction, we developed eight RNN-based DPLF models using a Keras deep learning environment. These models included vanilla RNN, GRU, LSTM, bidirectional LSTM (BiLSTM), attention-based RNN (Att-RNN), attention-based GRU (Att-GRU), attention-based LSTM (Att-LSTM), and attention-based BiLSTM (Att-BiLSTM). We selected the hyperparameters of these models through various trials, with a window size of 28, one hidden layer, 256 hidden units, mean square error as the loss function, Adam as the optimizer, batch size of 100, 500 epochs (with early stop), and a learning rate of 0.01. The attention-based RNN models included an attention layer and different RNN architectures.

Table 8 compares the MAPE of various RNN-based DPLF models for Buildings 1 and 2. The results show that almost all RNN models had inaccurate predictions due to insufficient data. However, the RAID model, a hybrid model, outperformed every model with the lowest MAPE values of 14.67 and 12.74 for Buildings 1 and 2, respectively, making it the best option for both accuracy and robustness. An additional advantage of the RAID model is that it provides excellent performance even when run on CPUs, outperforming deep learning models trained on GPUs.

Table 8. MAPE comparison between recurrent neural network (RNN) models and the RAID model.

Models	Building 1	Building 2
RNN	27.43	19.86
GRU	25.39	21.44
LSTM	25.03	20.01
BiLSTM	28.01	21.23
Att-RNN	29.06	21.52
Att-GRU	24.94	20.42
Att-LSTM	26.70	22.14
Att-BiLSTM	25.90	22.38
RAID (ours)	14.67	12.74

To evaluate the performance of the RAID model in DPLF, we compared it to other existing models such as the Persistence model, Lee's All-Season model, and our previous DPLF models (Cubist and HYTREM). The Persistence model is a simple statistical approach that uses historical data from the day before the prediction time point. As a baseline model, we used the All-Season model based on MLR proposed by Lee [22] because it can be easily implemented using the variable information we have. The exact equation of the All-Season model is as follows:

$$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-7} + \beta_3 y_{t-14} + \beta_4 w_t + \beta_5 x_t + \beta_6 i_t + \beta_7 m_t + \beta_8 s_t + \beta_9 y_{t-1} x_t + \beta_{10} m_t w_t + \beta_{11} s_t x_t + \beta_{12} w_t x_t + \beta_0 + \varepsilon \quad (13)$$

The model uses the following variables:

- y_{t-k} represents the peak load of day $_{t-k}$.
- w_t is the mean temperature in Celsius of day t (we converted Fahrenheit to Celsius).
- x_t is the day of the week of day t, where Monday, Tuesday, . . . , Sunday are the elements of the categorical variable.
- i_t is the index number of day t.
- m_t is the month to which day t belongs and is another categorical variable with twelve elements: January, February, . . . , December.
- s_t is a binary variable that indicates whether or not day t is a special day (1 if it is, 0 otherwise).
- $\beta_0, \ldots, \beta_{12}$ are parameters and ε is the model's error term.

We consider the autocorrelation effect by including terms y_{t-1}, y_{t-7}, and y_{t-14}. We also consider the mean temperature of the forecast day (w_t). x_t as well as y_{t-7} and y_{t-14} cover the day of the week effect. The index variable i_t, which takes values of 1, 2, . . . , N, where N is the number of samples used to estimate the parameters, captures the trend effect.

The model also includes four types of interaction effect:

- $y_{t-1} \cdot x_t$ represents the variation of autocorrelation with lag 1 depending on the day of the week.
- $m_t \cdot w_t$ shows that the temperature effect can vary across different months.
- $s_t \cdot x_t$ considers the differential impacts of special days based on the day of the week.
- $w_t \cdot x_t$ shows that the temperature effect can vary according to the day of the week.

Our previous model, Cubist, had a better prediction accuracy compared to other machine learning models such as MLR, partial least squares, multivariate adaptive regression splines, k-nearest neighbors, support vector regression, decision tree, and bagging. However, our HYTREM model had better prediction accuracy compared to other ensemble learning models such as RF, GBM, and XGBoost. Both Cubist and HYTREM demonstrated high levels of predictive accuracy.

Table 9 shows a comparison of the MAPE between existing DPLF models and the RAID model for two different buildings. In comparing the performance of these two models, we found that the RAID model proposed in Table 9 can deliver even better results. And these results show that the RAID model (our proposed model) had the lowest MAPE for both buildings.

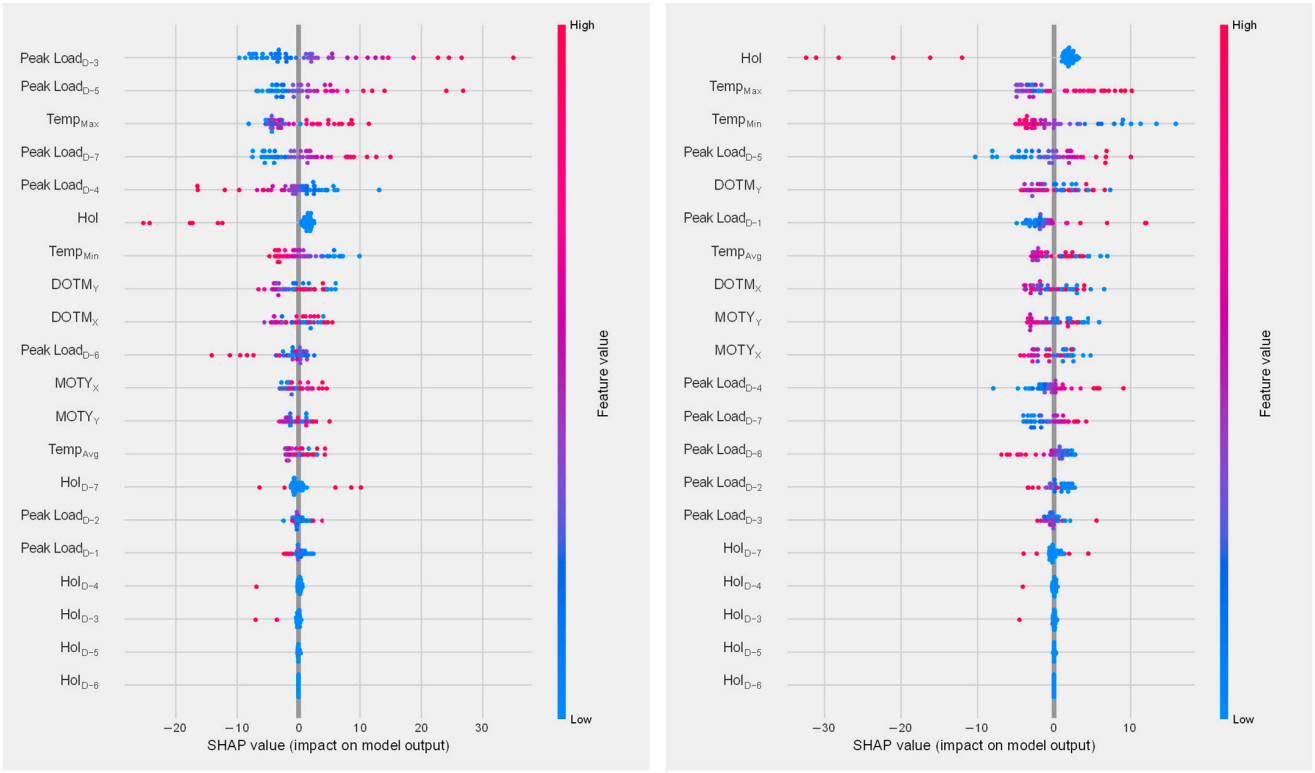
Table 9. MAPE comparison between existing DPLF models and the RAID model.

Models	Building 1	Building 2
Persistence	46.80	41.59
MLR (Lee et al.) [22]	54.91	52.64
Cubist (Moon et al.) [2]	16.98	13.51
HYTREM (Moon et al.) [8]	16.03	13.30
RAID (ours)	14.67	12.74

5.3. Model Explainability

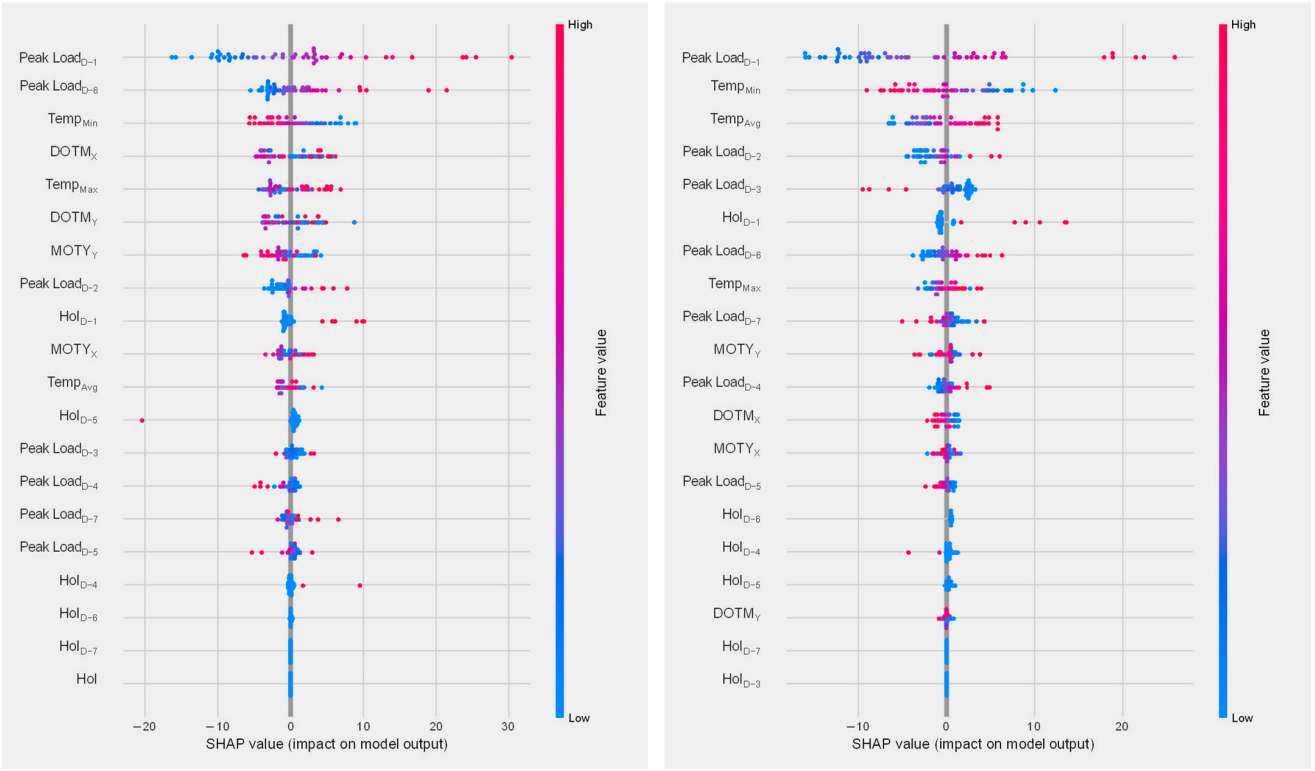
Ensuring the reliability of a model is a growing concern, and the use of explainability methods has gained significant attention as a solution [46]. To improve the explainability of the RAID model, we used the kernel SHAP method, which calculates the contribution of each input variable to the prediction. We visualized the results using a summary plot, displaying the contribution of each attribute (input variable) in the RAID model for each day of the week in Figures 8–14 for both datasets. We arranged the summary plot in order of the greatest influence, where a larger feature value has a greater impact on the forecasting model. The characteristics of the figures can be categorized into four groups: Monday (Figure 8), Tuesday to Friday (Figures 9–12), Saturday (Figure 13), and Sunday (Figure 14).

For Monday, we determined that the influence of Peak Load_{D-1} was less significant compared to other weekdays (Tuesday to Friday) because the day before was a public holiday (Sunday). In contrast, the day before the other weekdays was a weekday. In the case of Building 1, we found the closest weekday (Friday) represented by the variable Peak Load_{D-3} to be necessary. For Building 2, the presence or absence of a public holiday had the most significant impact on the prediction, and we observed a similar result in Building 1. We discovered that when Monday is a public holiday, most power devices are turned off at the beginning of the holiday, resulting in a low peak load, which significantly negatively impacts the prediction due to consecutive holidays.



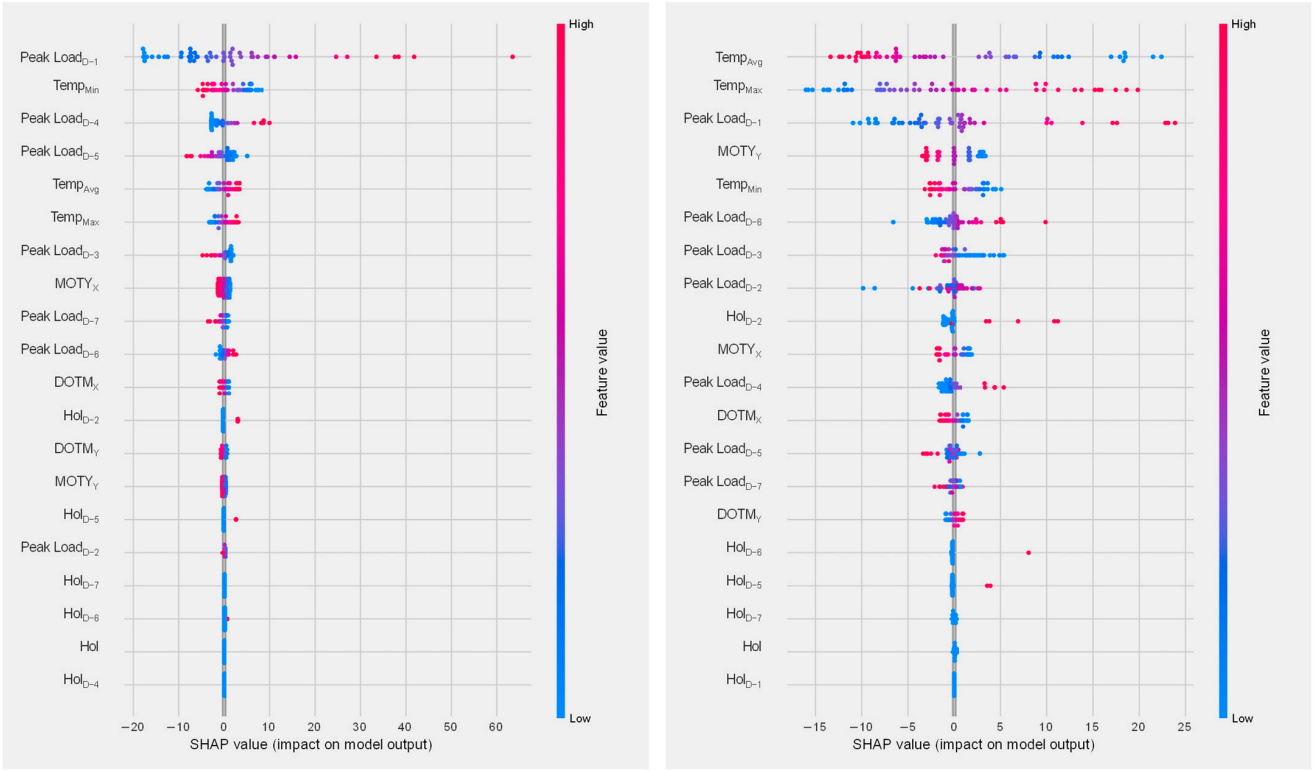
(**a**) (**b**)

Figure 8. Summary plot for the Monday's RAID model. (**a**) Building 1; (**b**) Building 2.



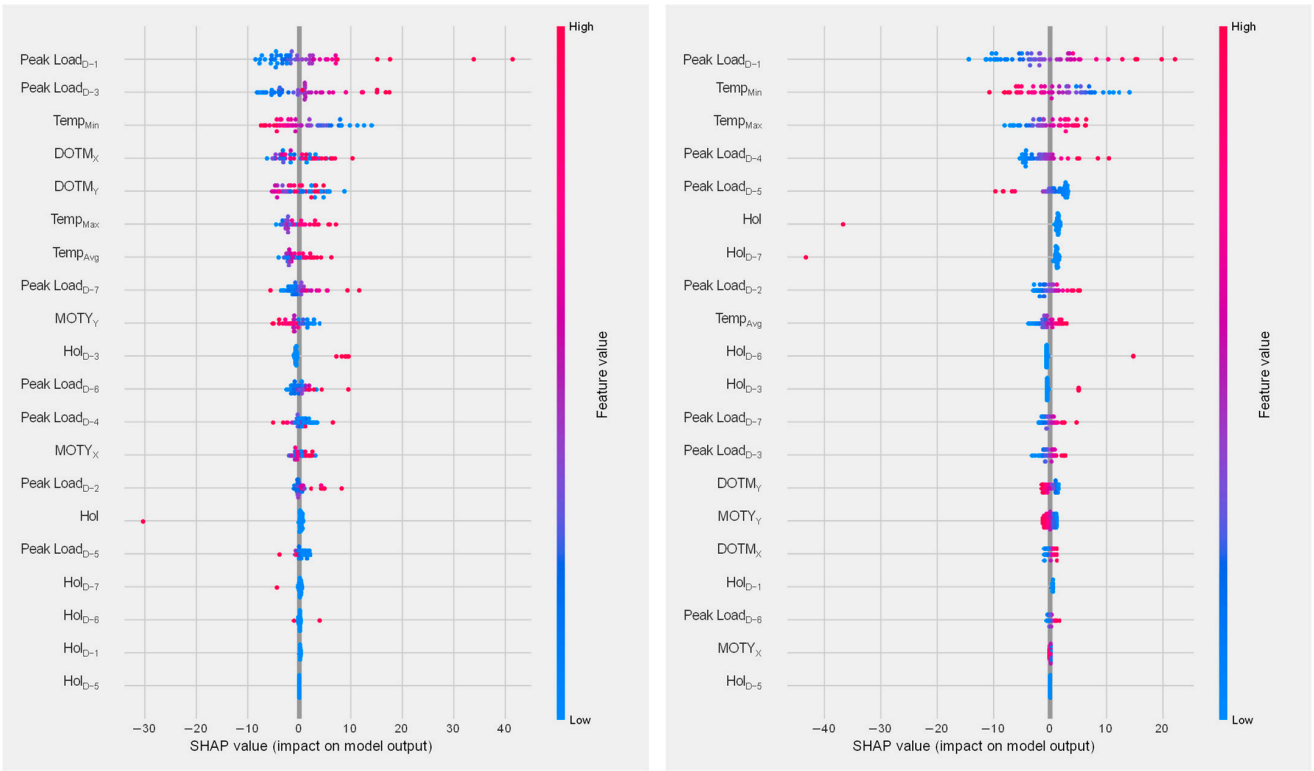
(**a**) (**b**)

Figure 9. Summary plot for the Tuesday's RAID model. (**a**) Building 1; (**b**) Building 2.



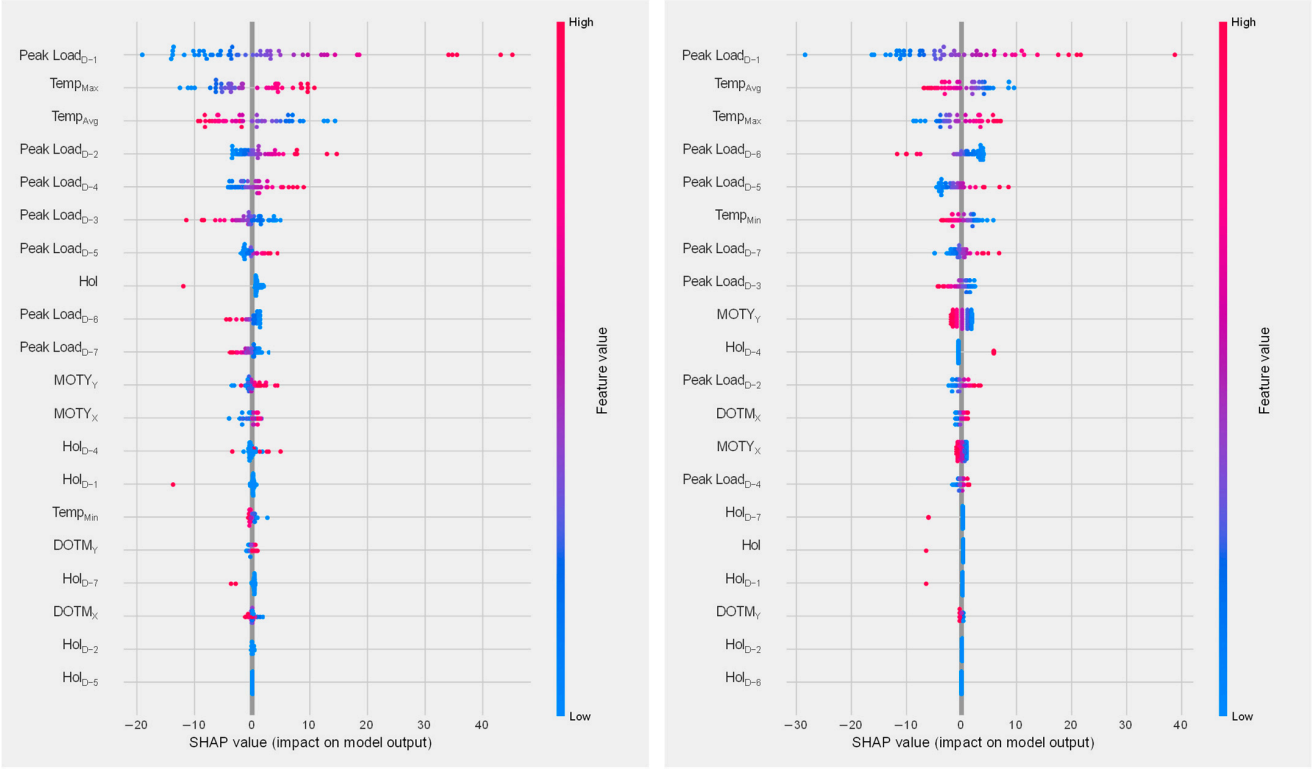
(**a**) (**b**)

Figure 10. Summary plot for the Wednesday's RAID model. (**a**) Building 1; (**b**) Building 2.



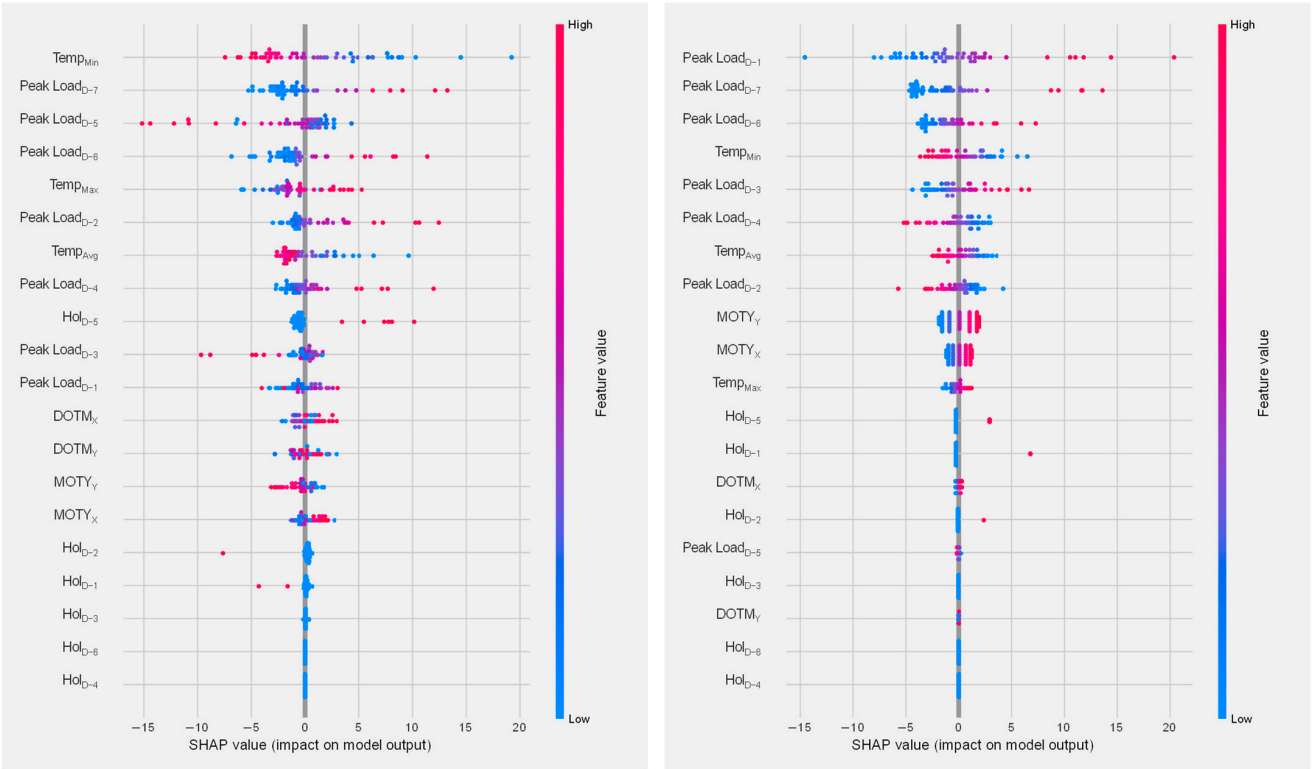
(**a**) (**b**)

Figure 11. Summary plot for the Thursday's RAID model. (**a**) Building 1; (**b**) Building 2.



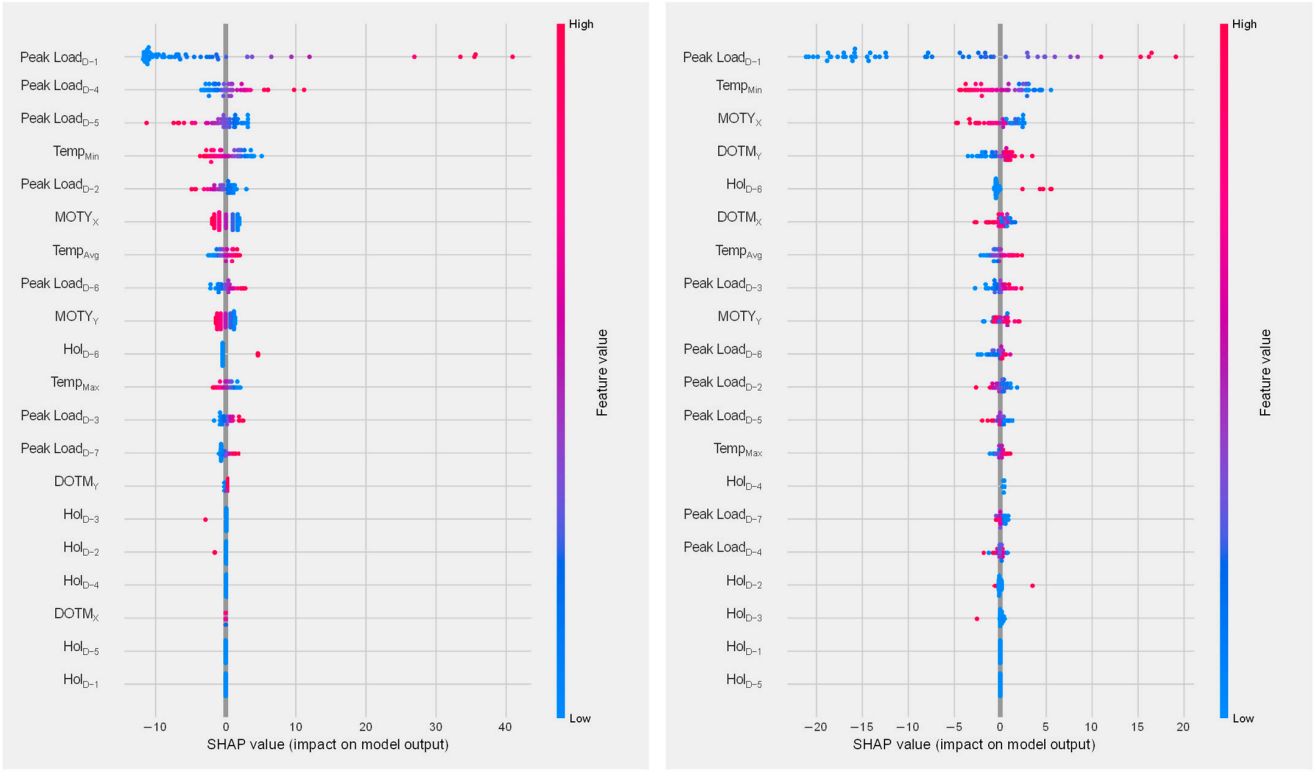
(**a**) (**b**)

Figure 12. Summary plot for the Friday's RAID model. (**a**) Building 1; (**b**) Building 2.



(**a**) (**b**)

Figure 13. Summary plot for the Saturday's RAID model. (**a**) Building 1; (**b**) Building 2.



(**a**) (**b**)

Figure 14. Summary plot for the Sunday's RAID model. (**a**) Building 1; (**b**) Building 2.

For Tuesday to Friday, we observed that the impact of the previous day's peak load (Peak Load$_{D-1}$) had a lasting effect on the current day's prediction for Tuesday to Friday, as these days are weekdays. This implies that if the previous day's peak load was low, it was likely that the current day's peak load would also be low, and vice versa. For example, if the daily peak load on Tuesday was 100 kW, the daily peak load on Wednesday would likely be low, say 80 kW. If the daily peak load on Thursday was 150 kW, the daily peak load on Friday would likely be high, say 180 kW. In addition to the impact of the previous day's peak load, we found that variables related to temperature have a strong correlation with the prediction. This means that the temperature of a day can significantly influence the daily peak load forecast.

For Saturday, although the previous day's peak load (Peak Load$_{D-1}$) had a significant impact on Building 2, we observed that the daily peak load of the previous week (Peak Load$_{D-7}$) had a higher correlation with the prediction because the daily peak load pattern on weekdays was different from the pattern on the weekend. Additionally, we determined that the minimum temperature had a high correlation with the prediction, but we found this correlation to be relatively insignificant because in the summer, when the temperature was at its highest, people were not present on Saturdays and the air conditioner was not turned on.

For Sunday, since the day before was a public holiday (Saturday), we observed that the previous day's peak load (Peak Load$_{D-1}$) was of high importance, unlike Saturday. Similar to Saturday, we also confirmed that the lowest temperature had a high correlation with the prediction. In general, past daily peak loads had a high level of influence, as reflected in the autocorrelation coefficient information, and variables related to temperature also had a significant impact. However, information on past holidays had a relatively low impact on the prediction, and it was difficult to determine if the timestamp had a significant impact because past daily load information can somewhat account for the timestamp information.

6. Discussion

In this study, we conducted additional experiments employing time series cross-validation (TSCV) techniques to address the issue of data scarcity and provide a more accurate representation of the expected model performance. TSCV is a widely used method that can better capture the temporal structure of the data. However, our results show that models without TSCV performed better on average, as demonstrated in Table 10, which displays the performance of the RAID model with and without TSCV for two different buildings across all days of the week.

Table 10. Comparison of RAID model performance with and without time series cross-validation.

Day of the Week	Building 1		Building 2	
	RAID w/o TSCV	RAID w/ TSCV	RAID w/o TSCV	RAID w/ TSCV
Monday	15.06	35.19	12.42	30.38
Tuesday	9.36	16.42	7.03	23.40
Wednesday	11.69	21.43	7.78	20.62
Thursday	10.66	23.79	8.94	24.86
Friday	11.13	28.58	8.86	18.57
Saturday	27.39	44.40	24.13	49.27
Sunday	17.18	66.14	19.78	61.84
Avg.	14.67	33.74	12.74	32.75

One crucial factor to consider is that during the TSCV process, the pre-determined optimal hyperparameter values obtained from the initial training set were used instead of optimizing the model's hyperparameters and parameters using Optuna each time step. This process led to the optimized values being tailored to the initial training data. When new, unseen data are introduced in the test set, the model's parameters must be adjusted accordingly. With limited data, the model's performance depends heavily on the chosen hyperparameter settings, which can lead to significant differences in performance, especially in DL models. In contrast, models like RFs can achieve superior predictive performance with less fine-tuning of hyperparameters, even with small data samples [47,48]. Therefore, for future research, we plan to consider constructing a TSCV model using RFs when dividing the data according to time series characteristics and applying TSCV. This approach addresses the limitations of using TSCV for evaluating model performance when dealing with limited data and fixed hyperparameter values.

Hyperparameter tuning is a critical step in improving a model's performance, but it can be time-consuming, particularly when there are numerous hyperparameters and the model is complex. In this study, we employed Optuna for hyperparameter tuning, which offers the advantage of efficiently and quickly exploring the hyperparameter space. Although crucial for enhancing the model's performance, it is difficult to consider it an essential step from an economic perspective due to the approximately 2 h required to perform hyperparameter tuning when constructing the RAID model. Future research should focus on finding ways to guarantee the model's economic efficiency while improving its performance.

7. Conclusions

DPLF is critical for ensuring a reliable and sustainable energy system for the future and enabling effective energy demand and supply management, reducing costs, and promoting sustainability. In this study, we developed a RAID model for DPLF using an MLP for feature learning, Optuna for hyperparameter optimization, and SHAP for model interpretation. We evaluated the performance of the DPLF models using the MAPE metric and compared it with other RNN-based DPLF models and existing DPLF models such as Persistence, Lee's All-Season model, Cubist, and HYTREM. The results showed that the proposed RAID model outperformed other models and we achieved the lowest MAPE values when we optimized the hyperparameters of the DNN models and we divided the daily peak load

data by day of the week. The study also used the kernel SHAP method to improve the explainability of the RAID model, which showed that the previous day's peak load and temperature had a significant impact on the prediction. Using Kernel SHAP, we can build trust in the AI model and help it be more easily adopted into EMSs. This method allowed us to understand how the model uses the input variables to make its predictions, which can be helpful for those who want to use the model in the future.

In conclusion, this study has made significant strides in advancing the DPLF model; however, some limitations must be addressed to improve its accuracy and generalizability further. The current model primarily focuses on external environmental factors, such as temperature, which could limit its applicability to other buildings. To overcome this, future research should incorporate internal building variables, such as occupancy levels, appliance usage, and lighting, to provide a more comprehensive understanding of daily peak load fluctuations. Moreover, expanding the dataset used for training and testing and considering additional environmental factors like humidity, wind speed, and atmospheric pressure will contribute to the model's robustness. Investigating different building types, including residential and commercial buildings, will enhance the model's accuracy and generalizability across various contexts.

The potential integration of generative adversarial networks (GANs) or a diffusion model for data augmentation offers promising avenues for addressing data scarcity issues in the DPLF model, as daily peak electrical load data are often represented by a single row for each day. GANs can generate realistic synthetic data, thereby enhancing the model's performance. While this approach could be particularly beneficial for DPLF, it also holds promise for other applications, including photovoltaic (PV) systems, where accurate forecasting is crucial. Future studies should explore the applicability of the proposed methodology to a diverse range of domains, including EMS and PV system, while harnessing the power of GPUs or distributed processing environments to maximize the efficiency of the proposed methodology.

By addressing these limitations and recommendations, we are confident that our collective efforts will lead to the development of a more accurate and reliable DPLF model, ultimately contributing to a sustainable and efficient energy system for the future. This endeavor benefits the scientific community and has profound implications for society as we strive to meet the increasing energy demands in an environmentally responsible manner.

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

Funding: This study was supported by the MSIT (Ministry of Science, ICT), Korea, under the National Program for Excellence in SW, supervised by the IITP (Institute of Information & communications Technology Planning & Evaluation) in 2021 (2021-0-01399) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019M3F2A1073179). This study was also supported by the Soonchunhyang University Research Fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are openly available in Mendeley Data at http://doi.org/10.17632/pfrpx3dn39.1 (accessed on 10 February 2023), reference number [27].

Acknowledgments: We would like to acknowledge valuable comments from Seungmin Rho, which led to significant improvements in the presentation quality relative to our original submission, and also express our sincere gratitude to the five reviewers for their insightful and valuable feedback, which has helped us improve our work.

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

References

1. Jamal, S.; Tan, N.M.L.; Pasupuleti, J. A Review of Energy Management and Power Management Systems for Microgrid and Nanogrid Applications. *Sustainability* **2021**, *13*, 10331. [CrossRef]
2. Moon, J.; Park, S.; Rho, S.; Hwang, E. Interpretable Short-Term Electrical Load Forecasting Scheme Using Cubist. *Comput. Intell. Neurosci.* **2022**, *2022*, 6892995. [CrossRef] [PubMed]
3. Mariano-Hernández, D.; Hernández-Callejo, L.; Zorita-Lamadrid, A.; Duque-Pérez, O.; Santos García, F. A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis. *J. Build. Eng.* **2021**, *33*, 101692.
4. Alzahrani, A.; Ramu, S.K.; Devarajan, G.; Vairavasundaram, I.; Vairavasundaram, S. A Review on Hydrogen-Based Hybrid Microgrid System: Topologies for Hydrogen Energy Storage, Integration, and Energy Management with Solar and Wind Energy. *Energies* **2022**, *15*, 7979. [CrossRef]
5. Mahmood, N.S.; Ajmi, A.A.; Sarip, S.B.; Kaidi, H.M.; Jamaludin, K.R.; Talib, H.H.A. Modeling the Sustainable Integration of Quality and Energy Management in Power Plants. *Sustainability* **2022**, *14*, 2460. [CrossRef]
6. Alhasnawi, B.N.; Jasim, B.H.; Siano, P.; Guerrero, J.M. A Novel Real-Time Electricity Scheduling for Home Energy Management System Using the Internet of Energy. *Energies* **2021**, *14*, 3191. [CrossRef]
7. Li, B. Effective energy utilization through economic development for sustainable management in smart cities. *Energy Rep.* **2022**, *8*, 4975–4987. [CrossRef]
8. Moon, J.; Park, S.; Rho, S.; Hwang, E. A Hybrid Tree-Based Ensemble Learning Model for Day-Ahead Peak Load Forecasting. In Proceedings of the 2022 15th International Conference on Human System Interaction (HSI), Melbourne, Australia, 28–31 July 2022; pp. 1–6.
9. Mughees, N.; Mohsin, S.A.; Mughees, A.; Mughees, A. Deep sequence to sequence Bi-LSTM neural networks for day-ahead peak load forecasting. *Expert Syst. Appl.* **2021**, *175*, 114844. [CrossRef]
10. Mohseni, S.; Brent, A.C.; Kelly, S.; Browne, W.N. Demand response-integrated investment and operational planning of renewable and sustainable energy systems considering forecast uncertainties: A systematic review. *Renew. Sustain. Energy Rev.* **2022**, *158*, 112095. [CrossRef]
11. Vargas-Salgado, C.; Berna-Escriche, C.; Escrivá-Castells, A.; Díaz-Bello, D. Optimization of All-Renewable Generation Mix According to Different Demand Response Scenarios to Cover All the Electricity Demand Forecast by 2040: The Case of the Grand Canary Island. *Sustainability* **2022**, *14*, 1738. [CrossRef]
12. Lee, J.; Cho, Y. National-scale electricity peak load forecasting: Traditional, machine learning, or hybrid model? *Energy* **2022**, *239*, 122366. [CrossRef]
13. Heidari, A.; Navimipour, N.J.; Unal, M. Applications of ML/DL in the management of smart cities and societies based on new trends in information technologies: A systematic literature review. *Sustain. Cities Soc.* **2022**, *85*, 104089. [CrossRef]
14. Hsu, Y.Y.; Tung, T.T.; Yeh, H.C.; Lu, C.N. Two-Stage Artificial Neural Network Model for Short-Term Load Forecasting. *IFAC-PapersOnLine* **2018**, *51*, 678–683. [CrossRef]
15. Sakurai, D.; Fukuyama, Y.; Iizaka, T.; Matsui, T. Daily Peak Load Forecasting by Artificial Neural Network using Differential Evolutionary Particle Swarm Optimization Considering Outliers. *IFAC-PapersOnLine* **2019**, *52*, 389–394. [CrossRef]
16. Nwakanma, C.I.; Ahakonye, L.A.C.; Njoku, J.N.; Odirichukwu, J.C.; Okolie, S.A.; Uzondu, C.; Ndubuisi Nweke, C.C.; Kim, D.-S. Explainable Artificial Intelligence (XAI) for Intrusion Detection and Mitigation in Intelligent Connected Vehicles: A Review. *Appl. Sci.* **2023**, *13*, 1252. [CrossRef]
17. Lee, J.; Jeong, J.; Jung, S.; Moon, J.; Rho, S. Verification of De-Identification Techniques for Personal Information Using Tree-Based Methods with Shapley Values. *J. Pers. Med.* **2022**, *12*, 190. [CrossRef]
18. Son, S.-Y.; Lee, S.-H.; Chung, K.; Lim, J.S. Feature selection for daily peak load forecasting using a neuro-fuzzy system. *Multimed. Tools Appl.* **2015**, *74*, 2321–2336. [CrossRef]
19. Yu, Z.; Niu, Z.; Tang, W.; Wu, Q. Deep Learning for Daily Peak Load Forecasting–A Novel Gated Recurrent Neural Network Combining Dynamic Time Warping. *IEEE Access* **2019**, *7*, 17184–17194. [CrossRef]
20. Ibrahim, B.; Rabelo, L. A Deep Learning Approach for Peak Load Forecasting: A Case Study on Panama. *Energies* **2021**, *14*, 3039. [CrossRef]
21. Lee, G.-C. Regression-Based Methods for Daily Peak Load Forecasting in South Korea. *Sustainability* **2022**, *14*, 3984. [CrossRef]
22. Lee, G.-C. Regression Based Methods with Interaction Effects for Daily Peak Load Forecasting. *J. Manag. Econ.* **2020**, *42*, 77–97.
23. Kim, D.-H.; Lee, E.-K.; Qureshi, N.B.S. Peak-Load Forecasting for Small Industries: A Machine Learning Approach. *Sustainability* **2020**, *12*, 6539. [CrossRef]
24. Kim, H.; Jeong, J.; Kim, C. Daily Peak-Electricity-Demand Forecasting Based on Residual Long Short-Term Network. *Mathematics* **2022**, *10*, 4486. [CrossRef]
25. Saxena, H.; Aponte, O.; McConky, K.T. A Hybrid Machine Learning Model for Forecasting a Billing Period's Peak Electric Load Days. *Int. J. Forecast.* **2019**, *35*, 1288–1303. [CrossRef]
26. Liu, J.; Brown, L.E. Prediction of Hour of Coincident Daily Peak Load. In Proceedings of the 2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), Washington, DC, USA, 18–21 February 2019; pp. 1–5.
27. Moon, J.; Park, S.; Rho, S.; Hwang, E. Robust building energy consumption forecasting using an online learning approach with R ranger. *J. Build. Eng.* **2021**, *47*, 103851. [CrossRef]

28. Time and Date. Available online: https://www.timeanddate.com/holidays/ (accessed on 10 February 2023).
29. Colelli, F.P.; Wing, I.S.; Cian, E.D. Air-conditioning adoption and electricity demand highlight climate change mitigation–adaptation tradeoffs. *Sci. Rep.* **2023**, *13*, 4413. [CrossRef]
30. Stefenon, S.F.; Seman, L.O.; Mariani, V.C.; Coelho, L.d.S. Aggregating prophet and seasonal trend decomposition for time series forecasting of Italian electricity spot prices. *Energies* **2023**, *16*, 1371. [CrossRef]
31. Seabold, S.; Perktold, J. Statsmodels: Econometric and statistical modeling with Python. In Proceedings of the 9th Python in Science Conference. Scipy, Austin, TX, USA, 28 June–3 July 2010; Volume 57, p. 61.
32. Abualsauod, E.H. Machine learning based fault detection approach to enhance quality control in smart manufacturing. *Prod. Plan. Control.* **2023**, 1–9. [CrossRef]
33. Richetti, J.; Diakogianis, F.I.; Bender, A.; Colaço, A.F.; Lawes, R.A. A methods guideline for deep learning for tabular data in agriculture with a case study to forecast cereal yield. *Comput. Electron. Agric.* **2023**, *205*, 107642. [CrossRef]
34. Ryu, S.; Noh, J.; Kim, H. Deep Neural Network Based Demand Side Short Term Load Forecasting. *Energies* **2017**, *10*, 3. [CrossRef]
35. Hackeling, G. *Mastering Machine Learning with Scikit-Learn*; Packt Publishing Ltd.: Birmingham, UK, 2017.
36. Scikit-Learn: Machine Learning in Python. Available online: https://scikit-learn.org/stable/faq.html#will-you-add-gpu-support (accessed on 10 February 2023).
37. Moon, J.; Jung, S.; Rew, J.; Rho, S.; Hwang, E. Combination of short-term load forecasting models based on a stacking ensemble approach. *Energy Build.* **2020**, *216*, 109921. [CrossRef]
38. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A Next-Generation Hyperparameter Optimization Framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; ACM: Anchorage, AK, USA, 2019; pp. 2623–2631.
39. Morteza, A.; Yahyaeian, A.A.; Mirzaeibonehkhater, M.; Sadeghi, S.; Mohaimeni, A.; Taheri, S. Deep Learning Hyperparameter Optimization: Application to Electricity and Heat Demand Prediction for Buildings. *Energy Build.* **2023**, *289*, 113036. [CrossRef]
40. Moon, J.; Rho, S.; Baik, S.W. Toward explainable electrical load forecasting of buildings: A comparative study of tree-based ensemble methods with Shapley values. *Sustain. Energy Technol. Assess.* **2022**, *54*, 102888. [CrossRef]
41. Chen, Z.; Xiao, F.; Guo, F.; Yan, J. Interpretable machine learning for building energy management: A state-of-the-art review. *Adv. Appl. Energy* **2023**, *9*, 100123. [CrossRef]
42. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 4765–4774.
43. Lundberg, S.M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J.M.; Nair, B.; Katz, R.; Himmelfarb, J.; Bansal, N.; Lee, S.I. From Local Explanations to Global Understanding with Explanable AI for Trees. *Nat. Mach. Intell.* **2020**, *2*, 56–67. [CrossRef]
44. Sundararajan, M.; Najmi, A. The many Shapley values for model explanation. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 9269–9278.
45. Sahagian, G. What is Random State? And Why is it Always 42? Available online: https://grsahagian.medium.com/what-is-random-state-42-d803402ee76b (accessed on 10 February 2023).
46. Maqsood, M.; Yasmin, S.; Gillani, S.; Aadil, F.; Mehmood, I.; Rho, S.; Yeo, S.-S. An autonomous decision-making framework for gait recognition systems against adversarial attack using reinforcement learning. *ISA Trans.* **2023**, *132*, 80–93. [CrossRef]
47. Zeini, H.A.; Al-Jeznawi, D.; Imran, H.; Bernardo, L.F.A.; Al-Khafaji, Z.; Ostrowski, K.A. Random Forest Algorithm for the Strength Prediction of Geopolymer Stabilized Clayey Soil. *Sustainability* **2023**, *15*, 1408. [CrossRef]
48. Borup, D.; Christensen, B.J.; Mühlbach, N.N.; Nielsen, M.S. Targeting predictors in random forest regression. *Int. J. Forecast.* **2023**, *39*, 841–868.