


## Article

# Hyperparameter Optimization of Bayesian Neural Network Using Bayesian Optimization and Intelligent Feature Engineering for Load Forecasting

M. Zulfiqar <sup>1</sup>, Kelum A. A. Gamage <sup>2,\*</sup> , M. Kamran <sup>3</sup> and M. B. Rasheed <sup>4</sup>

- <sup>1</sup> Department of Telecommunication Systems, Bahauddin Zakariya University, Multan 60000, Pakistan; zulfiqarchishti@gmail.com
- <sup>2</sup> James Watt School of Engineering, James Watt South Building, University of Glasgow, Glasgow G12 8QQ, UK
- <sup>3</sup> Department of Electrical Engineering, University of Engineering and Technology, Lahore 54000, Pakistan; mkamran@uet.edu.pk
- <sup>4</sup> Escuela Politécnica Superior, Universidad de Alcalá, ISG, 28805 Alcalá de Henares, Spain; muhammad.rasheed@uah.es
- \* Correspondence: kelum.gamage@glasgow.ac.uk

**Abstract:** This paper proposes a new hybrid framework for short-term load forecasting (STLF) by combining the Feature Engineering (FE) and Bayesian Optimization (BO) algorithms with a Bayesian Neural Network (BNN). The FE module comprises feature selection and extraction phases. Firstly, by merging the Random Forest (RaF) and Relief-F (ReF) algorithms, we developed a hybrid feature selector based on grey correlation analysis (GCA) to eliminate feature redundancy. Secondly, a radial basis Kernel function and principal component analysis (KPCA) are integrated into the feature-extraction module for dimensional reduction. Thirdly, the Bayesian Optimization (BO) algorithm is used to fine-tune the control parameters of a BNN and provides more accurate results by avoiding the optimal local trapping. The proposed FE-BNN-BO framework works in such a way to ensure stability, convergence, and accuracy. The proposed FE-BNN-BO model is tested on the hourly load data obtained from the PJM, USA, electricity market. In addition, the simulation results are also compared with other benchmark models such as Bi-Level, long short-term memory (LSTM), an accurate and fast convergence-based ANN (ANN-AFC), and a mutual-information-based ANN (ANN-MI). The results show that the proposed model has significantly improved the accuracy with a fast convergence rate and reduced the mean absolute percent error (MAPE).

**Keywords:** Bayesian Neural Networks; Bayesian Optimization; convergence rate; Hamilton dynamic; electric load forecasting



**Citation:** Zulfiqar, M.; Gamage, K.A.A.; Kamran, M.; Rasheed, M.B. Hyperparameter Optimization of Bayesian Neural Network Using Bayesian Optimization and Intelligent Feature Engineering for Load Forecasting. *Sensors* **2022**, *22*, 4446. <https://doi.org/10.3390/s22124446>

Academic Editors: Antonio Fernández-Caballero and Juan M. Corchado

Received: 5 May 2022

Accepted: 9 June 2022

Published: 12 June 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

An accurate ELF is essential for smart grids (SGs) to make strategic decisions such as operational and planning management [1], load switching [2], energy generation expansion, maintenance scheduling, security, demand monitoring inspections, and providing a reliable energy supply [3], since inaccurate forecast results may pose serious challenges in making short- and long-term decisions and planning for SGs. Overestimation in a forecast may lead to excessive spinning reserves, production capacity, and limited energy distribution, resulting in higher operational costs. In contrast, underestimation may create consistency, power quality, safety, and monitoring issues. Therefore, the distribution system operators (DSOs) need an acceptable accuracy to guarantee endurance and stable grid operation [4]. For this purpose, much devotion is given to provide instant, accurate, and stable load forecasts to ensure the safe and reliable operation of the grid [5]. However, the accuracy of ELF often cannot cope with societal requirements. It is affected by probabilistic and uncertain factors

such as economic development, human social activity, irradiance meteorological parameters, environmental parameters, temperature, climate change, and government policies [6]. Therefore, improving the accuracy of the predictive network is a challenge. Considering all influencing factors is unrealistic or cumbersome [7]. Consequently, we can improve prediction accuracy by developing a model that considers the key parameters. On the other hand, different forecasting algorithms have been developed to accurately predict the load. These include the time-series models such as Kalman filter-based approaches [8], the box–Jenkins model [9], state-space models [10], exponential smoothing [11], regression-based methods [12], and the autoregressive integrated moving average (ARIMA) [13]. Traditional statistical strategies identify and capture performance patterns and apply a time-series approach for better results. However, these techniques are useful for linear analysis and are unable to deal with nonlinear time-series problems, which is where artificial intelligence (AI) methods have also been commonly used to improve performance [14]. As these models are not involved in complex mathematical modeling, therefore, they can easily and properly process the historical load-demand data, effectively and efficiently [15]. The most commonly used AI-based approaches include expert systems [16], support vector machines (SVMs) [17], machine learning (ML) models [18], deep learning (DL) models [19], and artificial neural networks (ANNs) [20] that can be used for ELF. Among other forecasting techniques, an ANN is one of the most widely used algorithms due to its improved accuracy. However, an ANN has problems with network design, overfitting, history dependencies, and connection-weight estimation. However, in contrast, a BNN has recently received attention due to its excellent performance and capability to handle uncertain tetrameters [21]. This is due to its learning capability from the network architecture and the selection of weight vector, while network topology is comprised of three feed-forward layers, with an arbitrary number of hidden units. Then, by taking into account the network design, learning the weight vector entails the selecting of a weight vector with the largest posterior probability distribution [22–24]. Load forecasting time series typically consists of prefiltering, sequence modeling, independent forecasting, and aggregation. Therefore, a large number of hyperparameters need tuning for higher forecast accuracy. A grid search is mostly considered to change the configurations, due to its simplicity. However, repetitive searches without correlation can be extraordinarily time-consuming and biased. When faced with the tuning task, evolutionary and genetic algorithms become the choices for interpretability. The motivation for considering the BO algorithm [25] in hyperparameter tuning lies with two aspects. Firstly, the time cost is controllable and can be fixed according to our necessities. Secondly, the balance of exploitation and exploration is well-sustained compared to other existing models, making the search more effective. Ref. [26] implements the BO algorithm for the attention-based LSTM model for stream-flow prediction. Ref. [27] optimizes the network structure and the time lag for prediction. Similarly, [28] builds a hybrid model combining multi-ahead attention, LSTM, and CNN outputs, applying the BO algorithm to search for optimal model hyperparameters for COVID-19 prediction. All the works mentioned above validate the improvement of using the BO algorithm compared with baseline models.

The combined model of predictions is still inconsistent due to some open issues. The first drawback occurs when the model candidates change a little. For example, [29] only evaluates candidates' deep neural networks (DNN) such as multilayer perceptron (MLP), a convolutional neural network (CNN), and long short-term memory (LSTM). Ref. [30] combines only MLP, dynamic architecture for an ANN, a recurrent neural network (RNN), and an echo state network (ESN). Authors in [31] conclude that pure data-driven ML and DNN models are not well-suited for large-scale predictive competition. Combining the statistical and DNN predictive models provides more robust and effective predictions. An adaptive combinational method that assigns time-varying weights to the model to solve the underlying pattern shifts provides promising forecasts. However, some vital signs are delicate and prevent practitioners from using them efficiently. They must be adjusted manually. Last but not least, the performance of many combinational strategies has not

been fully confirmed due to the small amount of experimental data from related studies and the seasonal diversity.

Hence, a new feature engineering (FE) and optimization concept is introduced. The proposed BO algorithm has been selected to adjust the control parameters because of its fast convergence and robustness in finding the optimal solution [32–34]. The BO algorithm optimizes the threshold weights for the filters and finds the optimized thresholds to be used in the FE module for feature selection. The complete forecasting framework consists of an integrated framework of feature engineering (FE), a stochastic BNN model, and the BO algorithm (FE-BNN-BO). The performance of the proposed FE-BNN-BO model is validated by comparing the results with the existing models in terms of mean absolute percentage error (MAPE).

### 1.1. Contributions

The real contributions are presented as follows:

- An ingenious and robust framework, FE-BNN-BO has been proposed that integrates the FE and BO algorithm with BNN. The FE module solves the concern associated with redundancy and irrelevance (dimension reduction). In the meantime, the BO algorithm optimizes the hyper-parameters of the BNN predictor to enhance accuracy while securing fast convergence. The combination of the FE module and the BO algorithm significantly improves the performance and effectiveness of the BNN model.
- BNN models are complex in estimating computational efficiency and cannot handle uncertainty. Therefore, the iterative and irrelevant features may enhance the complexity, slow down the BNN training process and affect the prediction accuracy. The proposed FE addresses this problem by combining the Random Forest and Relief-F-based feature selector and radial-based kernel principal component analysis (RBKPCA)-based feature extractor. The feature selector converges the Random Forest with Relief-F, calculates the importance of the feature, selects the relevant feature, and discards the irrelevant feature. This further enhances the computational performance and efficacy of the BNN model.
- Moreover, the BO algorithm is automatically applied to search for the best ensemble configuration. The devised BO algorithm is more controllable and efficient in time and complexity than the widely used grid search methods.
- The proposed model is validated against the latest hourly load data obtained from the USA electricity grid. The proposed frameworks outperformed the benchmark frameworks, such as LSTM, ANN-MI, ANN-AFC, and Bi-Level, when considering the accuracy and convergence speed.

### 1.2. Paper Organization

The rest of the sections of this work are organized as follows: the recent and relevant work is demonstrated in Section 2. Section 3 illustrates the proposed system model. Section 4 discusses the simulation results and discussion. The paper is finalized in Section 5.

## 2. Literature Survey

This is because static and ML models are widely used in the literature. These can be divided into two major categories to understand better how they perform and the impediments that come with them. A detailed description is as follows.

### 2.1. Individual ELF Models

The individual models are used for ELF without fusing any other algorithm. Therefore, only the algorithmic efficacy is estimated using various performance parameters. The authors [35] have proposed a distribution practice for meteorological data predicting the prospective load. The energy system is further divided into two sub-systems depending on the climate. In addition, the two distinct forecasting models, Grey and ARIMA, are used in both sub-systems. The fitted models are assessed by approximating them to the

definitive models using MAPE as a performance metric. In [36], an individual approach based on a deep recurrent neural network (DRNN) is introduced to forecast the household load. This approach ensures the overfitting issue more efficiently than the classical DNN systems. Furthermore, the results show the improved performance of the proposed strategy by comparing it to the other single methods, such as ARIMA, SVM, Grey, and traditional RNNs. Authors in [37] proposed an RNN based on the long short-term memory (RNN-LSTM) framework to forecast the household loads. The forecast accuracy has been enhanced by utilizing the embedded appliance-usage series strategy of training data. However, the author ignores the convergence rate and computational complexity, and they focus only on accuracy. The demand response scheme has considered the ANN for the price forecasting in [38]. The proposed price-forecasting model uses mixed-integer linear programming (MILP) to lessen energy costs. Simulation results depict that hourly demand response is more optimistic than a day ahead, with an enhanced ability to encounter the industrial market by diminishing cost. The authors presented a probabilistic prediction model for predicting PV, electrical energy consumption, and scalability [39]. Quantile regression (QR) models and dynamic Gaussian processes (GP) are applied to the Sydney metropolitan area data for probabilistic prediction. Simulation results demonstrate that the proposed model excels in all three predictive scenarios. In [40], a long-term predictive model was proposed to improve the relative prediction accuracy of the integrated power resource plan. The authors [41] have investigated new aspects using loads and temperatures over the past few hours. The primary purpose is to determine the hourly moving average temperature with a time lag to improve the accuracy of the prediction. The impact of timeliness is examined in three scenarios: the aggregated geographic hierarchy level, the lowest geographic hierarchy level, and each time. However, it improves accuracy at the cost of model complexity. Though individual models are robust and fast converging, they are still inaccurate and do not reach the required level.

The above discussion finalizes that single strategies are not helpful in all facets (rate of convergence, accuracy, stability) due to each technique's unique flaws, imperfections, defects, and intrinsic limitations [41]. For example, non-linear and seasonal behavior cannot be learned from a linear regression-based model [12]. The gray model is distinctive for the exponential growth trend [11]. Expert systems rely on a solid knowledge base [16], and intelligent methods rely on thresholds, weights, biases, and hyperparameter adjustments [14]. These annoyances affect ELF and result in inconsistent performance. Due to these shortcomings, individual methods cannot achieve all goals (accuracy, rate of convergence, stability) simultaneously. Multiple optimization algorithms, such as meta-heuristic [42], bio-inspired [43], and heuristics [44], are integrated into a single model to devise hybrid models to overcome the problems and limitations of the single methods [33]. The goal is to attain increased precision and overwhelm the flux of final forecasts by optimizing thresholds, hyper-parameter adjustments, random weight initialization, and biases for individual models.

## 2.2. Hybrid ELF Models

The new integrated and hybrid predictive model is an intelligent solution that maximizes the desired characteristics of individual models to ensure superior performance [45,46]. The hybrid model integrates the FE engine and the optimization engine with a combination of prediction algorithms to improve accuracy by optimizing the control parameters of the prediction engine. For instance, a hybrid wavelet neural network (WNN) based on a stepped forward differential evolution (SFDE) framework is devised. The optimizing algorithm SFDE efficiently tunes the hyperparameters of the proposed WNN. Experimental results show that the proposed framework is efficient, when compared with different frameworks such as ANN-based particle swarm optimization (ANN-PSO), ANN-based genetic algorithm (GA-ANN), and ANN-based evolutionary programming (ANN-EP) in terms of accuracy, efficiency, effectiveness, and hyperparameters tuning for ELF [47]. A hybrid model of nonlinear AR with GA and an extrinsic NN is proposed

in [48] for STLF. Use statistical and pattern-recognition-based schemes to fine-tune the input parameters of the proposed model. GA is used for the weight and bias of the NN training selection. The proposed model is validated by comparing it to existing models such as means and regression tree models with extrinsic inputs. The author proposed a robust STLF framework with an automated data cleaning method for load prediction of distribution feeders [49]. The previous day's building level LF model was proposed based on DL [50]. The proposed DL model is validated by comparing the accuracy with the traditional models. An integrated framework for VMD, LSTM, and BO algorithms has been proposed [51]. This model aims to be superior to existing models regarding accuracy and stability. A modified hybrid model of the multipurpose cuckoo search algorithm (MCSA) and GNRR has been proposed [52]. The proposed model will be tested against existing models for predictive accuracy using real-time load data from the Australian energy market operator (AEMO). The author [53] proposed a prediction engine based on the Neural Elman network to predict the future load of SG. Intelligent algorithms optimally adjust the biases and weights of this network to acquire accurate predictions. The author [54] proposed an STLF model based on SSVR. The main goal is to enhance the accuracy and efficiency of comparative predictions. The output of the prediction engine passes through the optimization engine and fine-tuning the parameters to improve the accuracy and efficiency of exact predictions. However, the prediction accuracy is enhanced at the expense of computational complexity. A fused framework is presented in [55] based on SVR and DE to enhance the forecasting performance by adapting SVR parameters. The developed framework surpasses backpropagation ANN, regression frameworks, and typical SVR. While in [56], a hybrid of SVR and the fruit-fly (Ff) algorithm framework is designed to address hyperparameter selection and improve forecasting accuracy. In addition, a new approach has been developed to achieve accurate ELF by merging the firefly optimization algorithm (FFO) with the SVR model and fitting optimal hyperparameters in [57,58]. A hybrid prediction strategy has been proposed to solve the above problems. However, the hybrid prediction strategy has improved modeling capabilities compared to the non-hybrid methods. Still, slow convergence and long execution times have a problem due to the many adjustable parameters. In [59], the author used a Bi-Level strategy based on ANN and the DE algorithm for ELF. Methods based on AFC-ANN and the modified extended DE algorithm (MEDEA) [60] have been proposed to predict future loads [61].

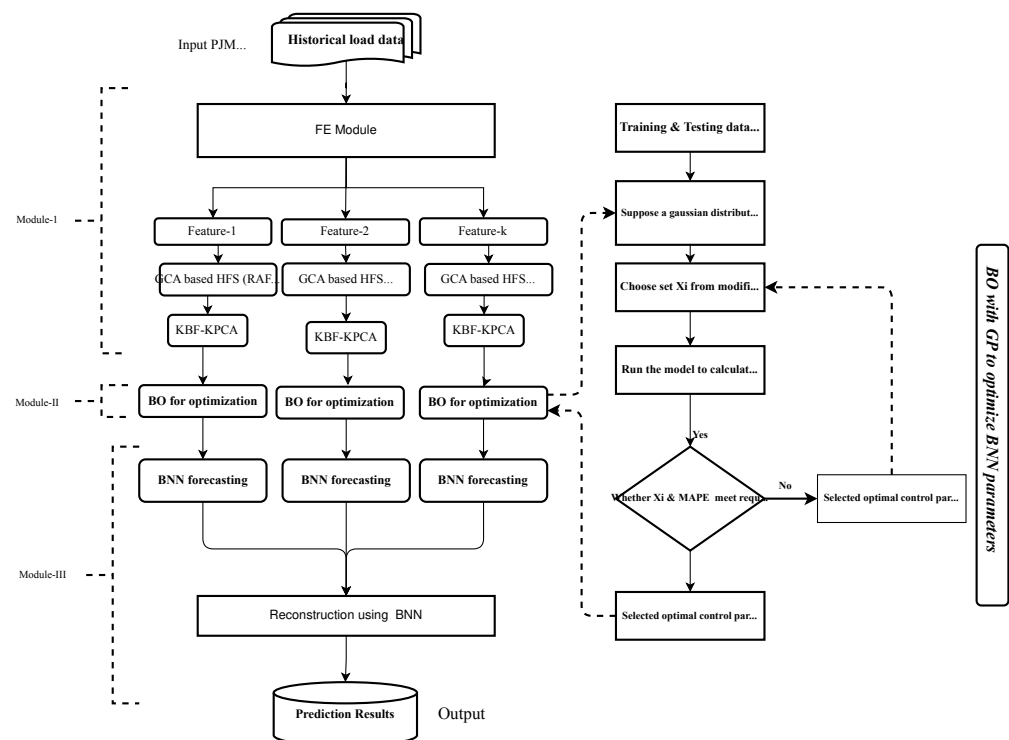
The combination of static optimal predictions calculates the weights of each model through pairwise performance fitting of training verifications that have been studied empirically and theoretically unpredictably. The authors have developed a hybrid model consisting of the navel switching delayed PSO (NSDPSO) algorithm and ELM for STLF [62]. Weights and biases are optimized using the proposed NSDPSO algorithm. The tanh function is considered an activation function because it has more generalization issues and avoids unwanted hidden nodes and overfitting issues. Experimental results show that the proposed framework is superior to RBFNN. The devised model successfully applies to the STLF of the energy system. A hybrid prediction framework has been developed that combines a feature extraction method with a two-step prediction engine. The two-stage prediction engine uses Ridgelet NN (RNN) and Elman NN (ENN) to provide accurate predictions. Optimization algorithms are applied to determine the control parameters of the prediction engine [63]. The hybrid models above can be considered optimistic and valuable in improving prediction accuracy by adequately modifying the hyperparameters. However, the authors of these articles focus on bias initialization and random weighting optimization or proper adjustment and selection of hyperparameters. Moreover, none of these models considered accuracy, rate of convergence, and stability simultaneously. From numerous analyses and investigations, one aspect (bias initialization and random weight optimization, or proper setting and selection of hyperparameters) and one measurement (convergence, accuracy, or stability) are not enough. We have concluded that it is sufficient. Therefore, a robust hybrid model is needed to overcome the problems of current models while improving prediction accuracy and stability with fast convergence rates.



From the literature, we can safely conclude that ELF has made great strides in energy management. However, existing approaches are not practical when dealing with large amounts of data. Adjusting control parameters is complex, and redundancy, irrelevance, and dimensionality reduction are unavoidable, which makes the calculation very difficult as it cannot quickly converge. Furthermore, the above literature does not consider forecast accuracy and rate of convergence at the same time. To address these problems, we require a fast and accurate model. Therefore, an SVM and gradient descent (GD) algorithm-based model is proposed [64]. However, this model introduces computational complexity and fails to converge. Some authors have focused on feature selection algorithms, traditional classifier decision trees (DTs), and ANNs [65]. However, the DT faces the problem of overfitting. So, while DT works well in training, it does not work well in prediction. ANNs have limited generalizability and are challenging to control convergence. The authors [41] proposed a hybrid feature selection (HFS), extraction, and classification model for STLF. However, this method is too complex to converge.

### 3. Proposed Model

This study proposes a novel hybrid framework based on the FE method, neural network model (BNN), and BO algorithm for ELF, as shown in Figure 1. This work targets only daily load forecasting using a new concept of scalability and robustness evaluation. The proposed model is an integrated framework of three modules: (i) an FE module comprised of hybrid feature selector (HFS) and the feature extractor (FX), (ii) a forecasting module based on the BNN model, and (iii) an optimizing module based on the BO algorithm.

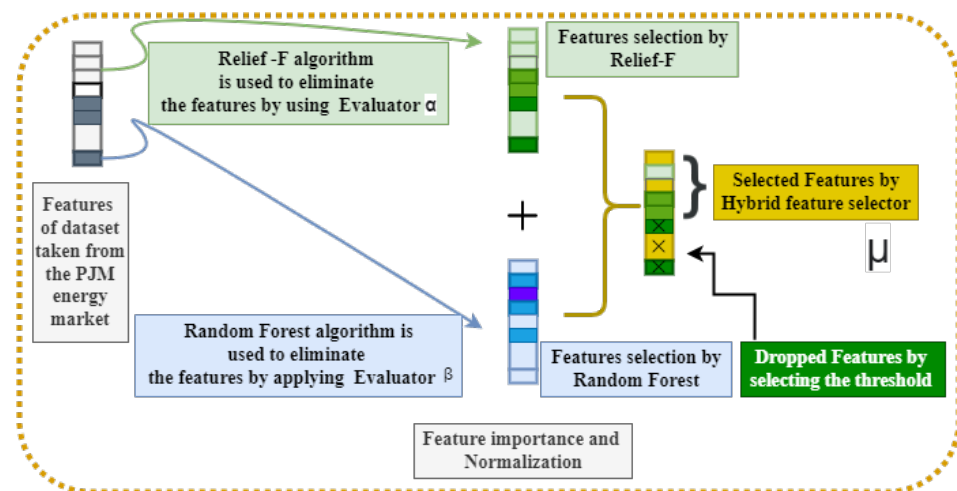


**Figure 1.** Single-line diagram of a new hybrid proposed model FE-BO-BNN for ELF.

#### 3.1. FE Module

The first module is FE. In this phase, abstract and critical features are picked and removed from the preprocessed data, while repetitious and irrelevant elements are discarded. The desired features are picked from the dataset by GCA and extracted by radial basis KPCA (RB-KPCA). FS relies on GCA to drive feature selection. It includes Relief-F (ReF) and Random Forest (RaF) algorithms for estimating the importance of features as depicted in Figure 2. In addition, the FS decides whether to reserve or abandon a feature based on the extent. RBKPCA-based FX uses kernel functions to process high-dimensional nonlinear

data. Feature extraction aims to reduce redundant features. Below is a brief demonstration of the FE module.



**Figure 2.** Illustration the details of fused feature selector (FFS) if a feature is reserved by an index, which is given by Relief-F and Random Forest.

### 3.1.1.1. FS

The feature selection system is primarily based totally on GCA, which has evolved with the aid of using RaF and ReF and is managed using a combined controlling threshold ( $\varphi$ ). The GCA more or less selects a feature space in which the maximum applicable and preferred features are kept, and inappropriate features are discarded primarily based totally on the feature significance and feature selection  $\varphi$ . Let  $\mathcal{L}$  be the electric load data matrix, which is defined as follows:

$$\mathcal{L} = \begin{bmatrix} l_{11} & l_{12} & l_{13} & \cdots & l_{1n} \\ l_{21} & l_{22} & l_{23} & \cdots & l_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{m1} & l_{m2} & l_{m3} & \cdots & l_{mn} \end{bmatrix} \quad (1)$$

The columns show the feature index, while the rows present the timestamps. Moreover,  $l_{mn}$  is the  $m$ th component of data, which is  $n$ th hour ahead of the electrical energy consumption pattern that is to be forecasted. Equation (1) can also be expressed in the form of a time sequence:

$$\mathcal{L} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{bmatrix} \quad (2)$$

where,

$$t_k = [l_{k1} \ l_{k2} \ l_{k3} \ \cdots \ l_{kn}] \quad k \in [1, m]. \quad (3)$$

Many factors affect the ELF pattern in different ways. GCA estimates the importance of each component and its impact on ELF. GCA effectively controls the feature selection process by determining the correlation between each feature and the final ELF pattern. GCA resolves the accessibility of data signals by correlation. Correlation is directly correlated to the proximity of the data signal. As a result, GCA determines the proximity of the two data signals. Each framework feature has its physical meaning and dimensions, so dimensionless data is standardized by the mean or maximum when GCA is executed. Since the original sequence has the characteristic of “the larger the better” [66], it can be normalized as:

$$Z_j^*(k) = \frac{Z_j(k) - Z_{\min_j}(k)}{Z_{\max_j}(k) - Z_{\min_j}(k)} \tag{4}$$

where  $Z_j(k)$  is the original sequence,  $Z_j^*(k)$  is the sequence after the data preprocessing,  $Z_{\max_j}(k)$  is the largest value of  $Z_j(k)$ , and  $Z_{\min_j}(k)$  is the smallest value of  $Z_j(k)$ . The grey relational coefficient ( $\eta$ ) after normalization [66] is calculated in Equation (5) as follows:

$$\eta(Z_0^*(k), Z_j^*(k)) = \frac{\Sigma_{\min} + \nu \Sigma_{\max}}{\Sigma_{0j}(k) + \nu \Sigma_{\max}}, \quad \nu \in (0, 1) \tag{5}$$

where  $\Sigma_{0j}(k)$  is the deviation sequence of the reference sequence  $Z_0^*(k)$  and the comparability sequence  $Z_j^*(k)$ , and  $\nu$  is a distinguishing coefficient, fixed to 0.50 [67] and represented in Equation (6):

$$\begin{aligned} \Sigma_{oi}(k) &= |Z_0^*(k) - Z_j^*(k)|, \\ \Sigma_{\max} &= \max_{j,k} |Z_0^*(k) - Z_j^*(k)|, \\ \Sigma_{\min} &= \min_{j,k} |Z_0^*(k) - Z_j^*(k)| \end{aligned} \tag{6}$$

The grey relational grade ( $\mathcal{G}_j$ ) [68,69] is a weighting sum of the  $\eta$ . It is defined in Equation (7) as follows:

$$\mathcal{G}_j(Z_0^*, Z_j^*) = \frac{\sum_{k=1}^m \eta(Z_0^*(k), Z_j^*(k))}{m} \tag{7}$$

Grey relational analysis is a measurement of the absolute value of the data difference between sequences, and it can be used to calculate approximation correlation. The low-correlated features are deleted, and the remainder of the selected items  $l_{kn}$  are arranged from least to most significant, providing the time sequence  $t_j$  as illustrated in Equation (8):

$$t_j = [l_{k1} \ l_{k2} \ l_{k3} \ \dots \ l_{kn-\delta}], \tag{8}$$

where  $\delta$  illustrates the dropped function and  $t_j$  is the time series.

The RaF evaluator  $\beta$  processes the boot-strap-bagging (BSB) samples [70]. BSB samples are split into out-of-bat (OoB) samples and training samples. In the first evaluator  $\beta$ , all weights are initialized to zero, and RaF training begins. Then, the feature's importance is determined by the OoB data with noise. For the second evaluator  $\alpha$ , the weights are updated with the concept of distance among hits and misses. Both  $\alpha$  and  $\beta$  evaluators forward the determined feature importance to the FS to perform feature selection based on the  $\varphi$ .

The  $\mathfrak{F}_i^f[\tau_k]$  and  $\mathfrak{F}_i^r[\tau_k]$  represents the feature importance calculated by ReF and RaF, respectively. The parameters are updated in Equations (9) and (10), respectively:

$$\mathfrak{F}_i^r[\tau_k] = \mathfrak{F}_i^r[\tau_k] - \frac{\sum_{j=1}^k F(D, b^*)}{n * k} \tag{9}$$

$$\begin{aligned} \mathfrak{F}_i^f[\tau_k] &= \mathfrak{F}_i^f[\tau_k] - \frac{\sum_{j=1}^k F(D, b^*, L_j)}{m * k} \\ &+ \frac{\sum_{C \neq class(x)} F(D, b^*, N_j(C))}{m * k}, \end{aligned} \tag{10}$$

where  $C$  is the class and  $b^*$  is the randomly selected item in the  $C$ , and function  $F(D, b^*, N_j(C))$  computes the attribute difference  $D$  between  $r_1$  and  $r_2$ . The function  $F$  is mathematically modeled as in Equation (11):



$$F = \begin{cases} 0 & \text{values are distinct} \\ 1 & \text{values are same} \\ \text{Difference among attributes is} \\ \text{normalized to be within [01]} \end{cases} \quad (11)$$

The ReF-based feature importance  $\mathfrak{F}_i^f$  and RaF-based feature importance  $\mathfrak{F}_i^r$  are normalized for feature selection, as depicted in Equations (12) and (13):

$$\mathfrak{F}_i^r = \frac{\tilde{\mathfrak{F}}_i^r}{\max(\tilde{\mathfrak{F}}_i^r)} \quad (12)$$

$$\mathfrak{F}_i^f = \frac{\tilde{\mathfrak{F}}_i^f}{\max(\tilde{\mathfrak{F}}_i^f)} \quad (13)$$

A combined feature importance value greater than  $\varphi$  is considered as a key feature, while those with value less than  $\varphi$  are rendered irrelevant. The core features are restored, while the irrelevant features are eliminated. This procedure is mathematically represented in Equation (14):

$$\tau_k = \begin{cases} \text{retain } \mathfrak{F}_i^r[\tau_k] + \mathfrak{F}_i^f[\tau_k] > \sigma \\ \text{remove } \mathfrak{F}_i^r[\tau_k] + \mathfrak{F}_i^f[\tau_k] \leq \sigma \end{cases} \quad (14)$$

The selected features are passed to a feature extraction phase that uses RBKPCA to reduce redundancy between features.

### 3.1.2. FX

The feature extraction procedure based on RB-KPCA is committed in the second stage. This operation aims to remove redundant data to solve the dimensionality-lessening problem. The output of the FS is sent to the RB-KPCA-based FX. This produces a dimensionally diminished matrix presented in Equation (15), including the most relevant features of interest that can be modeled as in [71]:

$$\mathcal{R} = (r_1, r_2, r_3, \dots, r_j)^T \quad (15)$$

where  $r_j$  is the  $j$ th variable associated with the EL. The correlation between eigenvalues and features is calculated as follows:

$$\lambda_{ev} = \mathcal{V}^{f^*} ev, \quad \lambda \geq 0 \ \& \ ev \in f^* \quad (16)$$

where the covariance matrix of  $\mathcal{R}$  is denoted by  $\mathcal{V}$ , and  $f^*$  represents the feature space, while the eigenvector is represented by  $ev$  and  $\lambda$  is the eigenvalue. Furthermore,  $\mathcal{V}^{f^*} ev$  is determined using Equation (17):

$$\mathcal{V}^{f^*} ev = \frac{1}{M} \sum_{j=1}^M \langle \phi(r_j), ev \rangle \phi(r_j), \quad (17)$$

while

$$\sum_{k=1}^M \phi(r_k) = 0, \quad (18)$$

where  $\phi$  shows the feature space and input data mapping, and  $\langle r, z \rangle$  expresses the product of  $r$  and  $z$ . Equation (16) becomes Equation (19) by proposing the above-mentioned modifications:

$$\lambda \langle \phi(r_k), ev \rangle = \langle \phi(r_k), \mathfrak{V}^{f^*} ev \rangle, \quad (19)$$

where  $ev$  for  $\lambda = 0$  can be determined as in Equation (20):

$$ev = \sum_{j=1}^M \gamma_j \phi(r_j), \quad (20)$$

where  $\gamma_j$  denotes indices corresponding to  $r_j$ . The kernel function mentioned in [72] is now utilized as follows in Equation (21):

$$\mathcal{K}_{jk} = \langle \phi(r_j), \phi(r_k) \rangle \quad \forall j, k \in [1, M] \quad (21)$$

after combining Equations (19) and (20), the combined form is defined as:

$$\lambda \sum_{j=1}^M \gamma_j \mathcal{K}_j = \frac{1}{M} \sum_{j=1}^M \gamma_j \sum_{k=1}^M \mathcal{K}_{kj} \mathcal{K}_{jk}, \quad (22)$$

where

$$\gamma = [\gamma_1, \gamma_2, \dots, \gamma_M]^T \quad (23)$$

Now, Equation (19) may be rewritten as:

$$\lambda M \mathcal{K} \gamma = \mathcal{K}^2 \gamma \quad (24)$$

To conduct dimensionality reduction by normalization, the eigenvectors  $\gamma$  and  $\lambda$  are chosen. Therefore, we have:

$$\langle ev_j, ev_k \rangle = 1 \quad \forall j, k \in [1, M]. \quad (25)$$

The consequential Equation (26) can be achieved by substituting Equation (20) for Equation (25), which is as follows:

$$\left\langle \sum_{j=1}^M \gamma_j^n \phi(r_j), \sum_{k=1}^M \gamma_k^n \phi(r_k) \right\rangle = 1 \quad (26)$$

The LHS of Equation (26) is further solved in Equations (27)–(30) to prove the Equation (25):

$$= \sum_{j=1}^M \sum_{k=1}^M \gamma_j^n \gamma_k^n \langle \phi(r_j), \phi(r_k) \rangle \quad (27)$$

$$= \sum_{j=1}^M \sum_{k=1}^M \gamma_j^n \gamma_k^n \mathcal{K}_{jk} \quad (28)$$

$$= \langle \gamma_n, \mathcal{K} \gamma_n \rangle \quad (29)$$

$$= \lambda_n \langle \gamma_n, \gamma_n \rangle \quad (30)$$

The principal component extraction can be calculated in Equation (31):

$$\mathcal{P}_n = \langle ev_n, \phi(r) \rangle = \sum_{j=1}^M \gamma_j^n \langle \phi(r_j), \phi(r) \rangle, \quad (31)$$

where  $\mathcal{P}$  signifies the principal element and the generalized versions of the kernel function are:

- Linear kernel function: the linear kernel is used when the data is linearly separable. It can be separated by one line. This is one of the most commonly used kernels.

This is primarily used when a particular dataset contains a large number of features. Mathematically, it may be formulated in Equation (32):

$$\mathcal{K}(r, z) = \langle r, z \rangle \quad (32)$$

- Kernel function based on logistic sigmoidal: this function is equivalent to a two-layer, perceptron model of the neural network, which is used as an activation function for artificial neurons. Equation (33) show the mathematical representation of kernel-based sigmoid function:

$$\mathcal{K}(r, z) = \tanh\left(a_0 \langle r, z \rangle^d + a_1\right) \quad (33)$$

- Kernel function based on radial basis: radial basis function kernels or RBF kernels are common kernel functions used in various kernel-learning algorithms. In particular, they are often used to classify SVMs. Mathematically, an RBF kernel is represented in Equation (34):

$$\mathcal{K}(r, z) = \exp\left(-\theta \|r - z\|^2\right) \quad (34)$$

After the FE step, the selected and extracted feature matrix is provided as input to the BNN-based prediction engine for predicting performance patterns.

### 3.2. BNN-Based Forecasting Module

The fundamental purpose of NN training is to obtain an appropriate network architecture  $\mathcal{A}$  and weight vector  $\mathbf{w}$ . NN offers an implicit function  $f(\mathbf{x}, \mathbf{w})$  design that connects the input variable  $\mathbf{x}$  to the output variable  $\eta$  provided as  $\mathcal{A}$  and  $\mathbf{w}$ . For the dataset  $\mathcal{D} = (\mathbf{x}_1, \tau_1), (\mathbf{x}_2, \tau_2), \dots, (\mathbf{x}_n, \tau_n)$ , with assumed  $\mathcal{A}$ , to achieve  $\mathbf{w}$  with reference to the weight vector by training it, the mapping function  $f(\mathbf{x}, \mathbf{w})$  has the lowest error  $E_{\mathcal{D}}(\mathbf{w})$  [73] and is presented in Equation (35):

$$E_{\mathcal{D}}(\mathbf{w}) = \frac{1}{2} \sum_{j=1}^n (f(\mathbf{x}_j, \mathbf{w}) - \tau_j)^2 \quad (35)$$

$$= \frac{1}{2} \sum_{j=1}^n \text{Exp}_j^2 \quad (36)$$

Overfitting concerns and generalizing performance reduction are always present in the NN training process. Therefore, a regularization approach in the NN training process used, and a new mathematical error called the generic error is upgraded by replacing a  $E_{\mathcal{D}}(\mathbf{x})$  mapping error [73] depicted in Equation (37):

$$\mathfrak{E}(\mathbf{w}) = \beta E_{\mathcal{D}}(\mathbf{w}) + \alpha E_{\mathbf{w}}(\mathbf{w}) \quad (37)$$

where

$$E_{\mathbf{w}}(\mathbf{w}) = \frac{1}{2} \sum_{j=1}^n \mathbf{w}_j^2 \quad (38)$$

The factor  $E_{\mathbf{w}}(\mathbf{w})$  in Equation (38) is known as the weight decay term, and the driving parameters are  $\alpha$  and  $\beta$ . They have the ability to impact the complexity and adaptability of NN. The prediction function of the ANN family is generally determined using the root mean squared error (RMSE) [74], which is defined in Equation (39):

$$\mathcal{RMSE} = \left( \frac{1}{n} \sum_{j=1}^n |f(\mathbf{x}_j, \mathbf{w}) - \tau_j|^2 \right)^{1/2} \quad (39)$$

The mean absolute error (MAPE) [74] is commonly used in the electrical market, and may be defined in Equation (40):

$$\mathcal{MAPE} = \frac{1}{n} \frac{\sum_{j=1}^n |f(\mathbf{x}_j, \mathbf{w}) - \tau_j|}{f(\mathbf{x}_j, \mathbf{w})} \times 100 \quad (40)$$

Using the Bayesian approach, Equation (41) depicts the posterior probability distribution of  $\mathbf{w}$  [75,76] as under:

$$P(\mathbf{w} | \mathcal{D}, \beta, \alpha) = \frac{P(\mathcal{D} | \mathbf{w}, \beta)P(\mathbf{w} | \alpha)}{P(\mathcal{D} | \beta, \alpha)} \quad (41)$$

where  $P(\mathcal{D} | \mathbf{w}, \beta)$  is the probability function, and  $P(\mathbf{w} | \alpha)$  is the prior of  $\mathbf{w}$ . If each error item presented in Equation (42):

$$\mathcal{E}_j = f(\mathbf{x}_j, \mathbf{w}) - \tau_j, j = 1, 2, \dots, n \quad (42)$$

Each error has a normal probability with zero mean and variance  $1/\beta$ , the weights  $w_k, k = 1, \dots, m$  would also be a normal with zero mean and variance  $1/\alpha$  [76], then

$$P(\mathcal{D} | \mathbf{w}, \beta) = \frac{1}{\mathfrak{Z}_{\mathcal{D}}(\beta)} \exp(-\beta E_{\mathcal{D}}(\mathbf{w})) \quad (43)$$

$$P(\mathbf{w} | \alpha) = \frac{1}{\mathfrak{Z}_{\mathbf{w}}(\alpha)} \exp(-\alpha E_{\mathbf{w}}) \quad (44)$$

Substituting (43) and (44) into (45), the posterior becomes as:

$$P(\mathbf{w} | \mathcal{D}, \alpha, \beta) = \frac{1}{\mathfrak{Z}_{\mathfrak{E}}(\alpha, \beta)} \exp(-\beta E_{\mathcal{D}} - \alpha E_{\mathbf{w}}) \quad (45)$$

$$= \frac{1}{\mathfrak{Z}_{\mathfrak{E}}(\alpha, \beta)} \exp(-\mathfrak{E}(\mathbf{w})) \quad (46)$$

and the evidence function has the accompanying structure [77]:

$$P(\mathcal{D} | \alpha, \beta) = \frac{1}{Z_{\mathcal{D}}(\beta)} \frac{1}{Z_{\mathbf{w}}(\alpha)} \int \exp(-M(\mathbf{w})) d\mathbf{w} \quad (47)$$

$$= \frac{\mathfrak{Z}_{\mathcal{S}}(\alpha, \beta)}{\mathfrak{Z}_{\mathcal{D}}(\beta) \mathfrak{Z}_{\mathbf{w}}(\alpha)} \quad (48)$$

Let  $\mathbf{w}^*$  be the maximum value point of  $P(\mathbf{w} | \mathcal{D}, \alpha, \beta)$ , i.e.,  $\mathbf{w}^*$  be the minimum value point of  $\mathfrak{E}(\mathbf{w})$  [24]. Using the Taylor expansion of  $\mathfrak{E}(\mathbf{w})$  around  $\mathbf{w}^*$  and retaining terms up to the second order, then

$$\mathfrak{E}(\mathbf{w}) = \mathfrak{E}(\mathbf{w}^*) + \frac{1}{2} \Delta \mathbf{w}^T \mathcal{H} \Delta \mathbf{w} \quad (49)$$

where

$$\Delta \mathbf{w} = \mathbf{w} - \mathbf{w}^* \quad (50)$$

$\mathcal{H}$  is the Hessian Matrix of  $\mathfrak{E}(\mathbf{w})$  at  $\mathbf{w}^*$ . Thus, the posterior distribution can be written as in Equation(51):

$$P(\mathbf{w} | \mathcal{D}, \alpha, \beta) = \frac{1}{\mathfrak{Z}_{\mathfrak{E}}^*(\alpha, \beta)} \exp(-\mathfrak{E}(\mathbf{w})) \quad (51)$$

where  $\mathfrak{Z}_{\mathfrak{E}}^*(\alpha, \beta)$  is the normalization factor. While:

$$\mathfrak{Z}_{\mathfrak{E}}^*(\alpha, \beta) = (2\pi)^{\frac{k}{2}} |\mathcal{H}|^{-\frac{1}{2}} \exp(-\mathfrak{E}(\mathbf{w}^*)) \quad (52)$$

Picking a legitimacy premise of the weight space, with the end goal that the  $\mathcal{H}$  is the identity  $\mathfrak{J}$  [78]:

$$\nabla \nabla E_{\mathbf{w}}(\mathbf{w}^*) = \mathfrak{J} \quad (53)$$

Setting

$$\nabla \nabla E_{\mathcal{D}}(\mathbf{w}^*) = \mathcal{A} \quad (54)$$

then

$$\mathcal{H} = \beta \mathcal{A} + \alpha \mathcal{I} \quad (55)$$

Let  $\lambda_1, \lambda_2, \dots, \lambda_p$  be the eigenvalues of the matrix  $\mathcal{A}$ , then the  $\mathcal{H}$  has eigenvalues  $\lambda_1 + \alpha, \lambda_2 + \alpha, \dots, \lambda_p + \alpha$  [79], hence

$$\frac{\partial}{\partial \alpha} \ln(|\mathcal{H}|) = \frac{\partial}{\partial \alpha} \ln\left(\prod_{j=1}^p \lambda_j + \alpha\right) \quad (56)$$

$$= \sum_{j=1}^p \frac{1}{(\lambda_j + \alpha)} \quad (57)$$

If the logarithm evidence for Equation (47) acquires maximization at point  $\alpha$ , then

$$\begin{aligned} \frac{\partial}{\partial \alpha} \ln(P(\mathcal{D} | \alpha, \beta)) &= -E_{\mathbf{w}}(\mathbf{w}^*) - \frac{1}{2} \frac{\partial}{\partial \alpha} \ln(|\mathcal{H}|) + \\ \frac{\mathcal{S}}{2\alpha} \frac{d\partial}{d\alpha} \ln(P(\mathcal{D} | \alpha, \beta)) &= 0 \end{aligned} \quad (58)$$

So,

$$2\alpha E_{\mathbf{w}}(\mathbf{w}^*) = p - \sum_{j=1}^p \frac{\alpha}{(\lambda_j + \alpha)} \quad (59)$$

$$= \sum_{j=1}^p \frac{\lambda_j}{(\lambda_j + \alpha)} \quad (60)$$

$$= \gamma \quad (61)$$

The most probable values of hyper-parameters  $\alpha, \beta$  are

$$\alpha^* = \frac{1}{2} \frac{\gamma}{E_{\mathbf{w}}(\mathbf{w}^*)}, \quad \beta^* = \frac{1}{2} \frac{n - \gamma}{E_{\mathcal{D}}(\mathbf{w}^*)} \quad (62)$$

The predicted power consumption pattern is dispatched to the optimization module to further minimize errors and enhance accuracy.

### 3.3. Optimization Module Based on BO

The optimizing module is used to provide accurate, dependable, and robust predicting outputs by interacting with BNN to optimize hyper-parameters to generate effective and consistent predictive results. A probabilistic model is used to construct many evolutionary algorithms for this purpose. The BO algorithm has gained a lot of attention among these optimization systems.

#### BO-Based Optimizer

Significant hyper-parameters in traditional models, such as BNN, heavily depend on the datasets. Assuring a correct fit for these hyper-parameters is an art. The DL framework employs a variety of hyper-parameters' tuning algorithms, such as grid search and random search, among others [32–34]. BO is a viable strategy for locating the extrema of a given objective function ( $\mathcal{O}\mathcal{F}$ ). The  $\mathcal{O}\mathcal{F}$  is estimated as a Gaussian process (GP) and perceived as a proxy function (pf). BO performs well when the closed-form expression of the provided  $\mathcal{O}\mathcal{F}$  is unknown, but specific observations may be derived from it. In our devised model, BO is employed to find the best hyper-parameters for discovering the test or validation loss minima. The hyper-parameter search space is denoted by  $\mathcal{S}$ , and model parameters



such as the number of hidden layers are represented by  $\mathcal{N}_h$ , dropout rate as  $\mathcal{T}_d$ , batch size as  $\mathcal{B}_s$ , etc. Thus, the  $\mathcal{O}\mathcal{F}$  can be expressed as:

$$\mathfrak{F} : \mathcal{S}(\mathcal{N}_h, \mathcal{N}_h, \mathcal{N}_h, \dots, \mathcal{N}_n) \subset \mathbb{R}^n \rightarrow \mathbb{R} \quad (63)$$

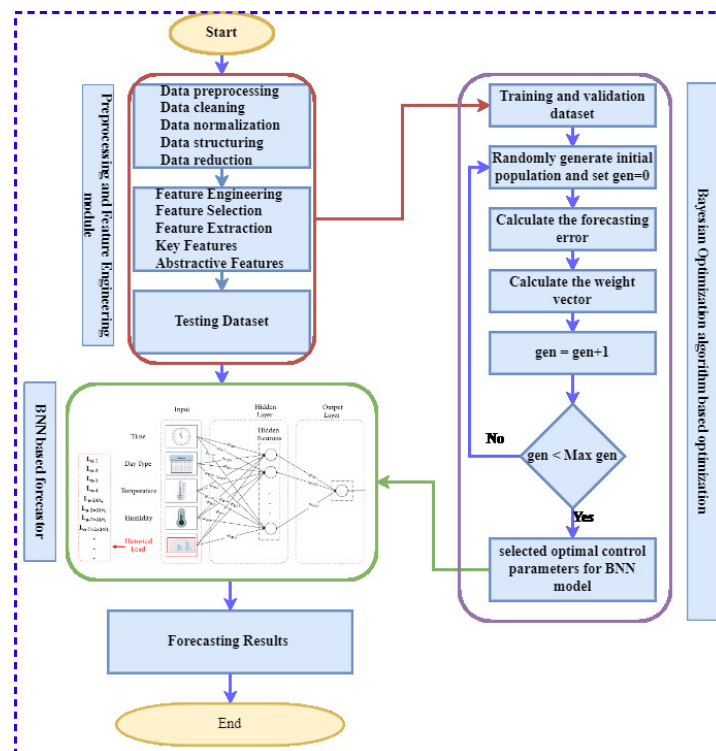
The  $\mathcal{S}$  for determining the optimal model hyper-parameter arrangement can be characterized as  $\mathfrak{s}^* \in \mathcal{S}$  such that:

$$\mathfrak{s}^* = \arg \min_{\mathfrak{s} \in \mathbb{R}} \mathfrak{F} \quad (64)$$

Here, the observations of the  $\mathcal{O}\mathcal{F}$  can be expressed as:

$$\mathcal{D}_{1:m} = (\mathfrak{s}_{1:m}, \mathfrak{F}_{\mathfrak{s}_{1:m}}) \quad (65)$$

This allows BO to develop a probabilistic model within  $\mathfrak{F}(\mathfrak{s})$ , so the model can be used to find the next position in  $\mathcal{S}$  in a sample search, which can be found using Bayesian theory, as BO builds a posterior distribution of  $\mathcal{O}\mathcal{F}$ , and subsequent hyperparameter configurations are selected from this distribution. Use the initial sampling point information to calculate the shape of  $\mathcal{O}\mathcal{F}$  and the hyperparameters that optimize the expected results. The framework hyper-parameters in our article generate that  $\mathcal{O}\mathcal{F}$ , and the goal is to optimize the negative of the validation loss. BO is used to calculate critical BNN hyper-parameter values. Since these variables affect prediction accuracy and robustness, BO designed a system to select and optimize the hyper-parameters of the BNN framework. Train the BNN network with these optimum circumstances after fine-tuning it with BO (best features plus fine-tuned hyper-parameters). This is the final forecasting model that will be put to the test. The overall step-by-step procedure of the proposed framework is depicted in Figure 3.



**Figure 3.** Step-by-step working flow chart of the proposed schematic framework for ELF. Red box shows data pre-processing and feature engineering module, green box shows BNN-based forecaster module, and purple box represents BO-algorithm-based optimization module.

### 3.4. BO Algorithm for Hyperparameters Tuning

As a model-based hyperparameter-tuning technique, the BO algorithm models the conditional probabilities of the validation set performance when hyperparameters are selected using surrogate functions. In contrast to the grid or random searches, the BO algorithm tracks all historical evaluations. Therefore, avoid wasting calculations to evaluate bad hyperparameters. In addition, the acquisition function finds the most promising hyperparameter to assess in the next iteration. The proposed model applies the BO algorithm strategies to find the optimal hyperparameters in the dynamic ensemble module. The BO algorithm achieves better tuning efficiency in a much shorter evaluation time. BO algorithm consists primarily of five parts: the hyperparameter space,  $\mathcal{O}\mathcal{F}$ , the acquisition function, the history of evaluations, and the surrogate function. In this article, we define the hyperparameters domain in Table 1. The  $\mathcal{O}\mathcal{F}$  is the forecasting error on the augmented validation data. We implement the tree-based Parzen window estimation (TPE) practice to accomplish the probabilistic modeling of the surrogate function and adopt the expected improvement to be the acquisition function  $A$ , defined in Equation (66).

$$A_{g^*}(\nu) = \int_{-\infty}^{g^*} (g^* - g)Q(g | \nu)dg, \quad (66)$$

where  $g$  is the  $\mathcal{O}\mathcal{F}$  and  $g^*$  is the threshold of  $\mathcal{O}\mathcal{F}$ , given the hyperparameter choice  $\nu$ . The simplified algorithmic description of the TPE-based BO algorithm is shown below in Algorithm 1.

---

#### Algorithm 1 TPE-based BOA for HP tuning.

---

**Require:**  $\mathcal{O}\mathcal{F}$   $g$ , TPE method  $\mathcal{M}$ , hyperparameter domain  $\mathbb{H}_\nu$ , acquisition function validation  $A$ , initialized memory  $\mathcal{U}$

**Ensure:**  $\mathcal{O}\mathcal{F}$   $f$ , TPE method  $\mathcal{M}$ , HP domain  $\mathbb{H}_\nu$ , acquisition function validation  $A$ , initialized memory  $\mathcal{U}$

- 1: for  $j = 1 : N$  do
  - 2:  $Q(g | \nu) \leftarrow$  Fit memory  $\mathcal{U}$  using  $\mathcal{M}$
  - 3:  $\nu_{j+1} \leftarrow$  Maximize acquisition function  $A$  in Equation (66) in search for the next hyperparameter choice
  - 4:  $g(\nu_{j+1}) \leftarrow$  Evaluate the  $\mathcal{O}\mathcal{F}$
  - 5:  $\mathcal{U} \leftarrow \mathcal{U} \cup (\nu_{j+1}, g(\nu_{j+1}))$
  - 6: End
- 

**Table 1.** Hyperparameter domain of the BO-algorithm-based tuning for the dynamic ensemble configurations.

Hyperparameters	Notation	Hyperparameter Space
Cold-start model index	$\iota_{cs}$	$\eta(1, 10)$
Maximum number of chosen models	$\mathcal{N}_{max}$	$\eta(1, 12)$
Size of past observations to evaluate	$\chi$	$\eta(1, 10)$
Weight-calculation function	$\omega$	Choice of $\left( \frac{ \cdot ^{-1}}{\sum  \cdot ^{-1}}, \frac{ \cdot ^{-2}}{\sum  \cdot ^{-2}}, \frac{\exp(- \cdot )}{\sum \exp(- \cdot )} \right)$
Discount factor of past observations	$\kappa$	$\nu(1, 1.5)$
Updated parameter	$\zeta$	$\nu(0, 1)$

$\eta$  denotes discrete uniform distribution;  $\nu$  denotes uniform distribution.

## 4. Simulation Results and Discussion

### 4.1. Simulation Setup

The CPUs and GPUs used in this task are an Intel Core i710701 K @ 3.82 GHz and an NVIDIA GeForce RTX 2070 SUPER. Modeling, training, tuning, and testing are programmed in Python 3.7. The libraries used for this task are: statsmodels 0.12.0 (for Relief-F and Random Forest), sklearn 0.23.1 (for LR, SVR, EWTFMMSVR, BNN, and ANNMI),

xgboost 1.2.1 (for XGB), Torch 1.6.0 (for ESN, MLP, LSTM, DeepAR, TCN, Nbeats, MOP-SOCD, ITVPNNW, ANNMI, and AFCANN), hyperopt 0.2.3 (for BO), and pymoo 0.4.2.1 (for EDNNDCE).

#### 4.2. Compared Models

The BO-algorithm-based optimization module directly correlates to between the convergence rate and accuracy. However, the devised model is better than the existing models such as ANN-MI [80], LSTM, Bi-Level [59], and AFC-ANN [81]. The above models have been determined as benchmark models due to structural resemblances with the evolved model. Convergence rate and accuracy are two quantifiable metrics to evaluate performance.

- Time consumed during execution by the forecasting approach is called convergence rate, and the execution time is calculated in seconds (s).
- While Accuracy ( $\mathfrak{A}$ ) is defined as:

$$\mathfrak{A} = 100 - MAPE \quad (67)$$

and is measured in (%).

The simulation parameters are enumerated in Table 2 and are retained as the same for the presented and benchmark schemes. The explicit depiction of the simulation results is presented as follows:

**Table 2.** Simulation parameters.

Control Parameters	Value
Hidden layer	3
Neurons in hidden	15
Output layer	1
Number of output neurons	1
Number of epochs	200
Number of iterations	200
Learning rate	0.0017
Momentum	0.55
Initial weight	0.1
Initial bias	0
Max	0.8
Min	0.2
Decision variables	3
Population size	23
Delay of weight	0.0003
Historical load data	4 yrs
Exogenous parameters	4 yrs

#### 4.3. Description of Dataset

Historical daily EL data from the publicly available PJM electricity market is used to evaluate the effectiveness of the proposed scheme. The training forecasting model is characterized by various factors (temperature, humidity, dew point, and time of day). The historic and hourly load data for the USA electrical system for the last four years (2017–2020) is used. The data includes humidity, temperature, and load parameters. The electric grid (FE) has the highest load profile and covers the most densely inhabited area. The dataset goes through the feature engineering module, where the abstracted features are extracted from the specified dataset. The subset (abstracted features) of the dataset is divided into training samples and test samples. We considered three years of data for network training and one year for network testing. The input vector, the above mentioned variables, and the main target load profile are included in the training data samples from 2017 to 2019. Test data samples are collected and used for testing in 2020. Data sample

validation is created from the training sequence data to improve the parameter selection for validation errors.

#### 4.4. Learning Curve Evaluation

A learning curve is a pictorial illustration that approximates the efficiency of a framework when training and testing data samples across different numbers of epochs. We can use the learning curve to notice if the selected model is training or storing data. If the bias and variance are high, the learning curve is poor, so the model does not memorize or learn. The high bias results in higher training and testing error as well as faster convergence rates. In contrast, significant variances occur when there is a substantial gap between training and test errors. In either case, the model is inappropriate, and the generalization is inadequate. Overfitting occurs when test errors begin to increase and training errors decrease. This shows that the model memorizes but does not learn. Therefore, such a model is under generalized. The dropout method and early stopping prevent overfitting problems [82]. However, for BNN, it is observed that the test errors gradually decrease, similar to the training errors in the USA power grid (FE). Therefore, the BNN model solved the problem of overfitting. In addition, the gap between training and testing errors is small, with no bias or variance, as shown in Figure 4.

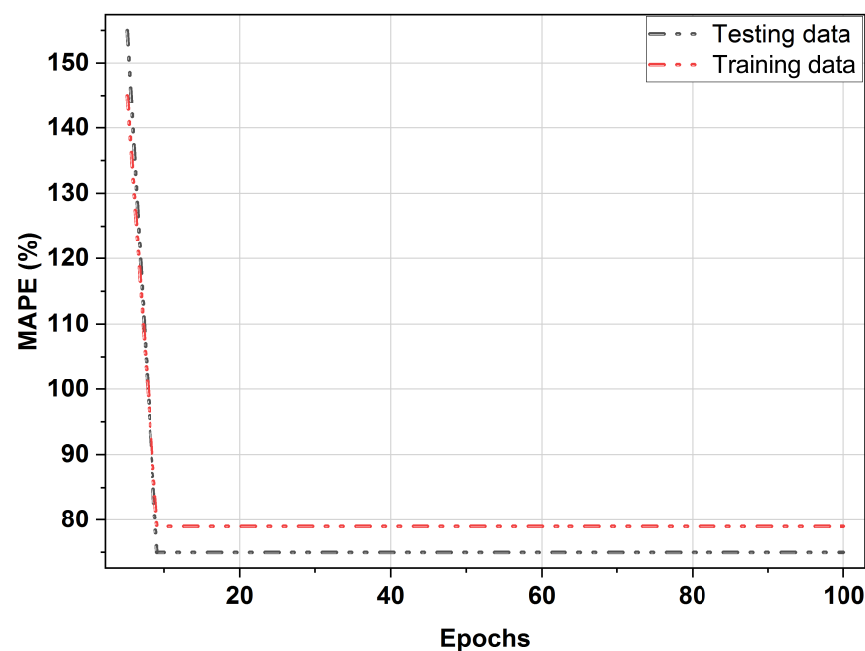


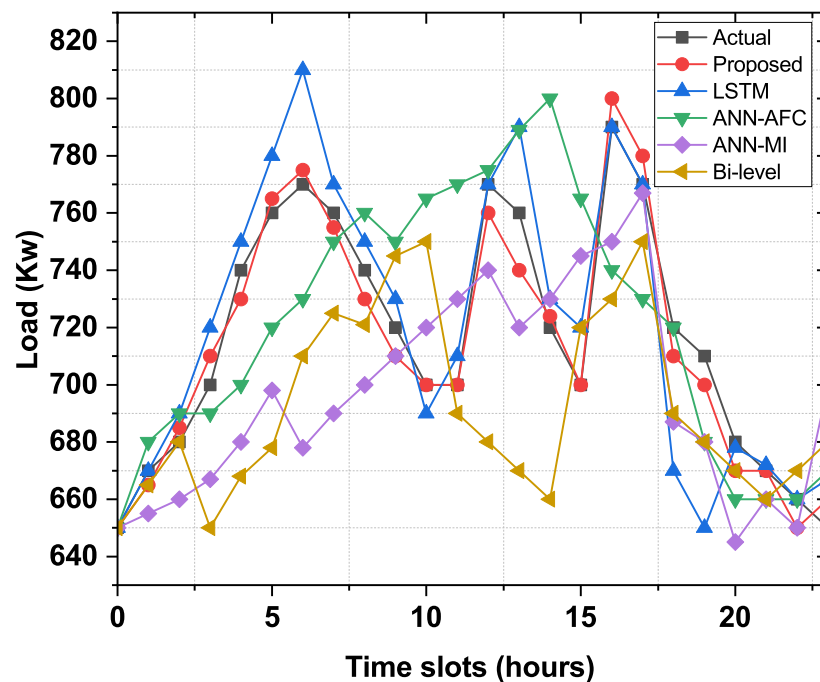
Figure 4. The BNN-EMC learning curve evaluation on FE.

#### 4.5. Day-Ahead Analysis

Figure 5 shows the day-ahead ELF profile with an hourly resolution using the developed and other benchmark frameworks, such as LSTM, Bi-Level, ANN-MI, and AFC-ANN for the USA power grid (FE). The graphical representation shows that all predictive models, including the proposed model, can capture nonlinear load behavior from historical data and predict future electrical loads based on the captured behavior. It is also clear that models such as Bi-Level, MI-ANN, AFC-ANN, and LSTM use the Levenberg–Marquardt, sigmoid activation function, and multivariate AR algorithms for network training. In contrast, customized BNNs are trained with the Tangent Hyperbolic (Tanh) function due to their short execution time. This can be seen in Figure 5. The predictive load for the day ahead, based on a BNN-based model and other benchmark models, is shown in Table 3. The MAPE of the devised BNN-based framework is 0.4920%, the MAPEs of the ANN-AFC, ANN-MI, and Bi-Level models are 2.9186%, 4.3371%, and 2.4741% respectively. The developed model has lowered MAPE compared to the benchmark models, resulting in superior accuracy.

**Table 3.** Targeted and forecasted load (F.load) of the devised and benchmark frameworks evaluated by using MAPE.

Hours	Targeted (kW)	Proposed and Benchmark ELF Frameworks							
		Proposed		ANN-AFC		ANN-MI		Bi-Level	
		F.load	MAPE	F.load	MAPE	F.load	MAPE	F.load	MAPE
		(kW)	(%)	(kW)	(%)	(kW)	(%)	(kW)	(%)
00.00	670.3223	667.1829	0.4525	673.4829	1.5400	645.5829	3.9157	650.8923	3.1255
01.00	677.7923	674.4538	0.4926	660.4538	2.5581	665.4538	1.8204	630.7923	6.9343
02.00	700.3192	703.7687	0.4926	715.7687	4.6806	635.7687	9.2173	725.3192	3.5698
03.00	734.5654	738.1835	0.4926	725.1835	2.0696	745.1835	1.4455	759.5654	3.4034
04.00	760.9115	757.1637	0.4924	743.1637	1.5923	777.1637	2.1359	740.9115	2.6284
05.00	767.4346	771.2146	0.4925	755.2146	4.4182	795.2146	3.6199	797.4346	3.9091
06.00	754.7077	758.4250	0.4915	730.4250	4.0749	745.4250	0.8851	750.7077	2.3554
07.00	744.3962	748.0627	0.4923	714.0627	4.3205	755.0627	1.2300	740.3962	0.5300
08.00	731.4692	727.8664	0.4925	699.8664	1.7461	718.8664	1.4329	735.4692	0.5373
09.00	717.9577	714.4214	0.4926	705.4214	0.7822	700.4214	1.7229	720.9577	0.5468
10.00	706.0231	709.5006	0.4926	700.5006	1.4930	730.5006	2.4425	760.0231	6.4179
11.00	699.6500	703.0961	0.4925	710.0961	3.9058	699.0961	0.0792	685.6500	2.0010
12.00	703.1462	706.6095	0.4925	730.6095	2.8131	725.6095	3.1947	707.6213	0.9955
13.00	726.0346	729.6107	0.4926	705.6107	2.2272	750.6107	3.3850	710.1462	1.9283
14.00	753.6077	757.3196	0.4925	755.3196	1.1835	700.3196	7.0711	740.0346	1.5923
15.00	768.8000	772.5867	0.4925	785.5867	3.6695	785.5867	2.1835	765.6077	1.2435
16.00	768.8538	772.6408	0.4925	740.6408	4.5999	778.6408	1.9500	720.8000	6.6503
17.00	754.7423	751.0248	0.4925	720.0248	1.1768	740.0248	0.1325	763.8538	2.1325
18.00	730.7462	734.3454	0.4926	739.3454	1.5246	690.9239	1.3136	755.7423	3.7790
19.00	703.3885	699.9239	0.4926	715.9239	2.1544	670.9967	1.7721	743.7462	8.8145
20.00	682.3577	678.9967	0.4925	760.9967	1.2358	655.1795	1.6650	765.3885	0.1153
21.00	661.9192	665.1795	0.4926	676.1795	2.1540	630.0057	1.0182	682.3577	2.1151
22.00	672.6923	676.0057	0.4926	681.0057	1.2822	630.1417	6.3456	675.9192	0.4797
23.00	676.6923	680.1750	0.4926	686.5057	1.4502	633.1530	5.8778	680.6923	1.1893
Avg.			0.4920		2.4741		2.9186		4.3371



**Figure 5.** Actual and predicted load day-ahead assessment in terms of prediction accuracy.



#### 4.6. Convergence Rate Evaluation

Figure 6 presents a performance rating of the devised and benchmark models in relation to the rate of convergence of the USA power grid (FE). There is an inverse relation between the rate of convergence and forecast accuracy. The ANN-AFC framework is more accurate than the ANN-MI framework. This gain in accuracy reaches at the expense of more prolonged execution times. As shown in Figure 6, the execution time rises from 20 s to 110 s. The execution time of the proposed framework has been reduced for two reasons:

- Abstractive features are fed into the training and forecasting module, reducing network training time.
- The BO algorithm is used due to its significantly faster convergence rate.

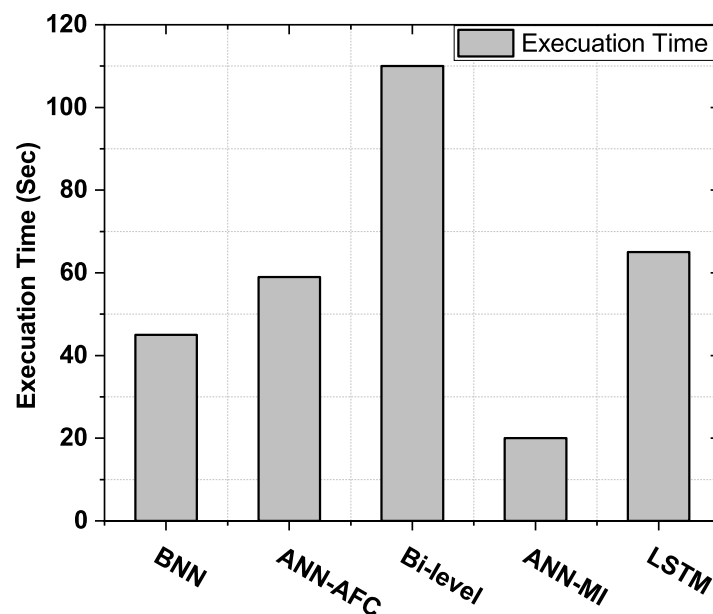
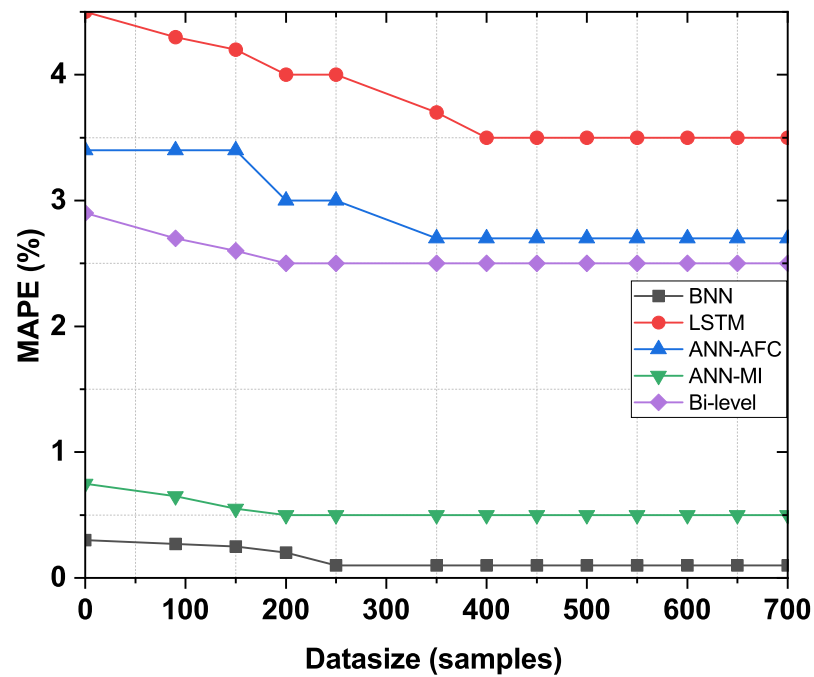


Figure 6. Convergence rate of daily load data of USA power grid (FE).

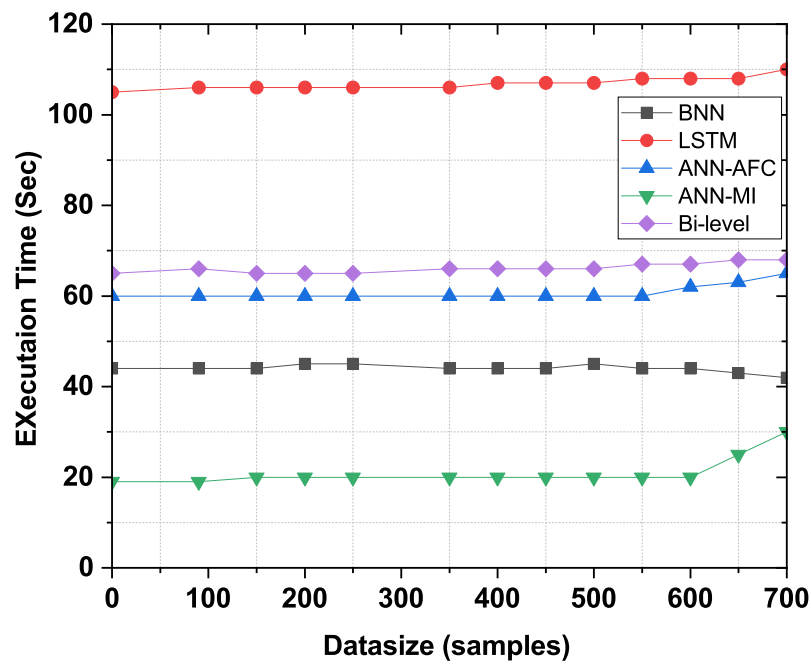
Due to the adjustments made to the proposed model, the proposed STLF framework lowered the execution time from 110 s to 42 s. In contrast, ANN-MI has excellent performance in term of convergence rate, even though this model has no optimization module integrated into it. This tendency is seen well in Figure 6.

#### 4.7. Scalability Analysis

Scalability research shows whether the framework under development is scalable or suitable for the scenario under consideration. Bias, threshold, input samples, and random weights are adjusted and tuned by Equation (1). These factors affect the accuracy by calculating the errors and convergence rate by calculating the execution time of the proposed framework depicted in Figure 7a,b. Forecast accuracy increases from 0 to 700 data samples and tends to stabilize while boosting samples. We can notice the effect from the value of  $l$  in expression (1). It is closely related to BNN training. An important value of  $l$  during the training process indicates fine-tuning and increases forecast accuracy. Similarly, Figure 7b shows the relationship between sample size and execution time. Using FE for feature selection, BNN for prediction, and BO algorithm for optimization, the developed model shows relatively good scalability compared to the benchmark models.



(a)



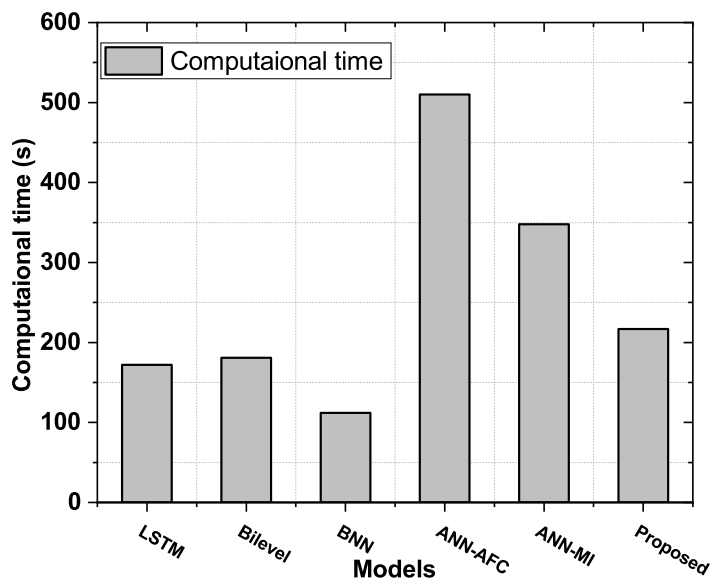
(b)

**Figure 7.** Scalability evaluation of the proposed model and benchmark models in terms of (a) accuracy and (b) convergence rate.

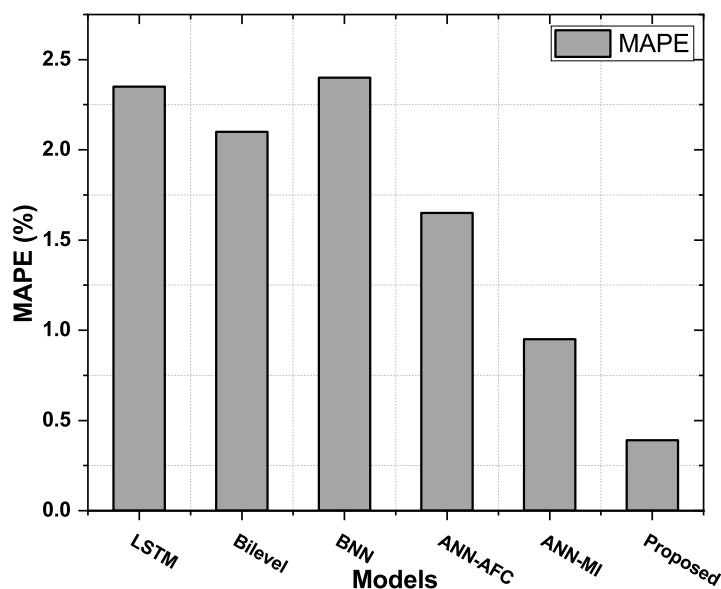
#### 4.8. Computational Time Analysis

Individual models, LSTM, BNN, and Bi-Level do not integrate both FE and optimization modules, resulting in short computational times ( $\tau_c$ ), and the worst error performances for daily time horizons are listed in Table 4. The performance analysis of proposed and benchmark frameworks in terms of computational time and MAPE is depicted in Figure 8a,b. However, when both the FE and the optimization modules are integrated into these individual models, the trade-off between accuracy and rate of convergence increases

$\tau_c$  and reduces errors. The individual models, LSTM, BNN, and Bi-Level have the shortest  $\tau_c$  of 172 s, 182 s, and 112 s, respectively, and the hybrid models (ANN-AFC, ANN-MI, and the proposed FE-BNN-BO) have  $\tau_c$  of 510 s, 348 s, and 217 s, respectively.



(a)



(b)

**Figure 8.** Performance analysis of proposed and benchmark frameworks in terms of computational time and MAPE. (a) Scalability evaluation of the devised and benchmark frameworks by error performance. (b) Comparative analysis of models with and without FE and optimization algorithm in terms of MAPE (%).

When the optimization module, the FE module, or both modules are integrated with the individual predictive models, they counteract the increased  $\tau_c$ . In addition, this increase in time is due to the trade-off between convergence speed and accuracy, thereby achieving higher accuracy at the expense of excessive  $\tau_c$ . The proposed FE-BNN-BO framework reduces  $\tau_c$  by developing changes to the BO algorithm. Therefore, the FE module modifies the functional space by removing redundant and irrelevant features, and

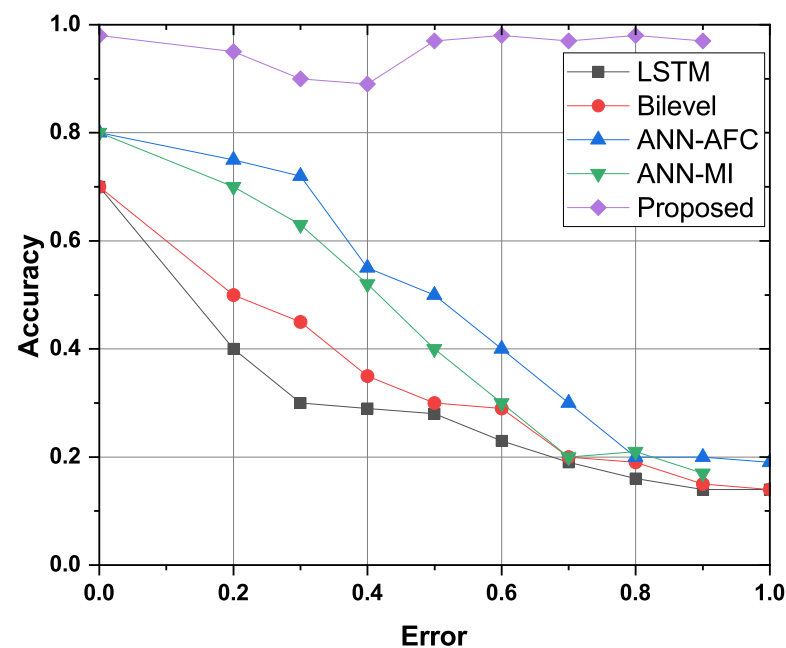
the BO-algorithm-based optimization module adjusts the control parameters of BNN to ensure an accurate ELF.

**Table 4.** Analysis of BNN model with and without FE and optimization modules for ELF.

Models without FE and Optimization Modules						Models with FE and Optimization Modules					
LSTM		Bi-Level		BNN		ANN-AFC		ANN-MI		Proposed	
$\tau_c$ (s)	MAPE (%)	$\tau_c$ (s)	MAPE (%)	$\tau_c$ (s)	MAPE (%)	$\tau_c$ (s)	MAPE (%)	$\tau_c$ (s)	MAPE (%)	$\tau_c$ (s)	MAPE (%)
172	2.31	182	2.1	112	2.3	510	1.45	348	0.95	217	0.39
240	2.31	276	2.1	187	2.3	579	1.45	408	0.95	265	0.39
308	2.31	321	2.1	245	2.3	456	1.45	456	0.95	412	0.39

#### 4.9. Robustness Evaluation

Stochastic noises (white noise, harmonic noise, asymmetric dichotomous noise, and Lévy noise) have a great adverse effect on the prediction accuracy of electric power load. Pre-filtering real-time can effectively improve measurement accuracy. Pretreating and statistically inspecting the electric power load data is essential to characterize the stochastic noise of the electric power load. The proposed feature engineering (FE) is used to denoise load data. FE is significantly reduced stochastic noise amplitude of power load data. Therefore, the proposed time series model and FE method can effectively suppress the stochastic noise of the power load data and improve the prediction accuracy of the power load in order to maintain the robustness of the proposed model. Figure 9 shows the robustness assessment of the proposed FE-BNN-BO model and benchmark models such as LSTM, Bi-Level, ANN-AFC, and MI-ANN. The evaluation is performed by adding an error (noise) to each function and observing the accuracy of each scheme. The proposed framework is more robust than the benchmark frameworks. This is because the noise in the feature has little effect on accuracy, reducing the number of important and irrelevant features dropped during the FE phase. Therefore, the proposed FE-BNN-BO framework is also robust against functional noise.



**Figure 9.** Robustness analysis of devised and other benchmark models.

**Remark 1.** *Energy consumption forecasting is of prime importance for the restructured environment of energy management in the electricity market. Accurate energy consumption forecasting is essential for efficient energy management in the smart grid (SG); however, the energy consumption pattern is non-linear with a high level of uncertainty and volatility. Keeping in view the non-linearity and complexity of the investigated problem, a BO algorithm is proposed for the optimization module of the proposed model to further improve accuracy with reasonable convergence of the forecasting results returned from the BNN-based forecaster. The proposed FE-BO-BNN model is examined on FE power grid data from the USA in terms of MAPE and convergence rate. Simulation results validated that the proposed FE-BO-BNN model achieved 0.4920 accuracy in terms of MAPE, which is better than the benchmark models, such as Bi-Level (2.4721), AFC-ANN (2.9286), and MI-ANN (4.3371). The proposed model reduced the average execution time by 21.1%, 35.5%, and 61% when compared to MI-ANN, AFC-ANN, and Bi-Level, respectively. It is concluded that our proposed FE-BO-BNN model outperformed benchmark electrical-energy-consumption forecasting models in terms of both accuracy and convergence rate.*

## 5. Conclusions

ELF is an essential component of the reliable operation of the energy system, since accurate LF is helpful in reducing the generation-demand mismatch through optimal decision-making and advance planning. However, the short- and/or long-term power generation or infrastructure planning depends on accurate forecast results with the possibility of marginal error, although a great effort is being given to the development of accurate forecasting algorithms. However, there is still a possibility to further improve the algorithmic accuracy by considering their control parameters, since the performance and accuracy depend on these control parameters. In this regard, the paper has presented a new and hybrid load-forecasting model based on BNN and BO. The proposed framework has used the BO algorithm to fine-tune the hyper-parameters of BNN to improve its accuracy. The FE module is integrated into the BNN model to further improve the computational efficiency and solve the problem of model dimensionality reduction. Through this combination, the proposed model simultaneously achieves higher stability, convergence, and accuracy. The devised framework is assessed using an hourly load dataset obtained from the USA energy grid (FE). The devised model is evaluated and compared with other latest models such as Bi-Level, ANN-AFC, ANN-MI, and LSTM, considering accuracy and convergence rate. In other words, the proposed ELF model outperforms Bi-Level by 15.73%, MI-ANN by 29.1%, and AFC-ANN by 3.97%.

**Author Contributions:** Conceptualization, M.Z., M.K. and M.B.R.; methodology, M.Z., M.B.R. and K.A.A.G.; software, M.Z., M.B.R.; validation, M.K., K.A.A.G. and M.B.R.; formal analysis, M.Z. and M.B.R.; investigation, M.Z., M.B.R. and K.A.A.G.; resources, M.K. and K.A.A.G.; writing—original draft preparation, M.Z., M.B.R. and K.A.A.G.; writing—review and editing, K.A.A.G. and M.B.R.; supervision, M.K. and M.B.R.; project administration, M.K. and K.A.A.G.; funding acquisition, K.A.A.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** This project has received funding from the European Union Horizon 2020 research and innovation programme under the Marie Skłodowska–Curie grant agreement, No. 754382, GOT ENERGY TALENT.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

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



## References

1. Masa-Bote, D.; Castillo-Cagigal, M.; Matallanas, E.; Caamaño-Martín, E.; Gutiérrez, A.; Monasterio-Huelín, F.; Jiménez-Leube, J. Improving photovoltaics grid integration through short time forecasting and self-consumption. *Appl. Energy* **2014**, *125*, 103–113. [[CrossRef](#)]
2. Feinberg, E.A.; Genethliou, D. Load forecasting. In *Applied Mathematics for Restructured Electric Power Systems*; Springer: Boston, MA, USA, 2005; pp. 269–285.
3. Xiao, L.; Wang, J.; Yang, X.; Xiao, L. A hybrid model based on data preprocessing for electrical power forecasting. *Int. J. Electr. Power Energy Syst.* **2015**, *64*, 311–327. [[CrossRef](#)]
4. Notton, G.; Voyant, C. Forecasting of intermittent solar energy resource. In *Advances in Renewable Energies and Power Technologies*; Elsevier: Amsterdam, The Netherlands, 2018; pp. 77–114.
5. Xiao, L.; Wang, J.; Hou, R.; Wu, J. A combined model based on data pre-analysis and weight coefficients optimization for electrical load forecasting. *Energy* **2015**, *82*, 524–549. [[CrossRef](#)]
6. Zhang, X.; Wang, J.; Zhang, K. Short-term electric load forecasting based on singular spectrum analysis and support vector machine optimized by Cuckoo search algorithm. *Electr. Power Syst. Res.* **2017**, *146*, 270–285. [[CrossRef](#)]
7. Lin, C.T.; Chou, L.D. A novel economy reflecting short-term load forecasting approach. *Energy Conv. Manag.* **2013**, *65*, 331–342. [[CrossRef](#)]
8. Zhang, M.; Bao, H.; Yan, L.; Cao, J.P.; Du, J.G. Research on processing of short-term historical data of daily load based on Kalman filter. *Power Syst. Technol.* **2003**, *10*, 200.
9. Irisarri, G.; Widergren, S.; Yehsakul, P. On-line load forecasting for energy control center application. *IEEE Trans. Power Appar. Syst.* **1982**, *PAS-101*, 71–78. [[CrossRef](#)]
10. Dordonnat, V.; Pichavant, A.; Pierrot, A. GEFCom2014 probabilistic electric load forecasting using time series and semi-parametric regression models. *In. J. Forecast.* **2016**, *32*, 1005–1011. [[CrossRef](#)]
11. Christiaanse, W. Short-term load forecasting using general exponential smoothing. *IEEE Trans. Power Appar. Syst.* **1971**, *2*, 900–911. [[CrossRef](#)]
12. Amral, N.; Ozveren, C.; King, D. Short term load forecasting using multiple linear regression. In Proceedings of the 2007 42nd International Universities Power Engineering Conference, Brighton, UK, 4–6 September 2007; pp. 1192–1198.
13. Wang, Y.; Wang, J.; Zhao, G.; Dong, Y. Application of residual modification approach in seasonal ARIMA for electricity demand forecasting: A case study of China. *Energy Policy* **2012**, *48*, 284–294. [[CrossRef](#)]
14. Lin, W.M.; Gow, H.J.; Tsai, M.T. An enhanced radial basis function network for short-term electricity price forecasting. *Appl. Energy* **2010**, *87*, 3226–3234. [[CrossRef](#)]
15. Xiao, L.; Shao, W.; Liang, T.; Wang, C. A combined model based on multiple seasonal patterns and modified firefly algorithm for electrical load forecasting. *Appl. Energy* **2016**, *167*, 135–153. [[CrossRef](#)]
16. Uyar, M.; Yildirim, S.; Gencoglu, M.T. An expert system based on S-transform and neural network for automatic classification of power quality disturbances. *Expert Syst. Appl.* **2009**, *36*, 5962–5975. [[CrossRef](#)]
17. Yang, J. Power System Short-Term Load Forecasting. Ph.D. Thesis, Technical University, Darmstadt, Germany, 2006.
18. Yildiz, B.; Bilbao, J.I.; Sproul, A.B. A review and analysis of regression and machine learning models on commercial building electricity load forecasting. *Renew. Sustain. Energy Rev.* **2017**, *73*, 1104–1122. [[CrossRef](#)]
19. Tong, C.; Li, J.; Lang, C.; Kong, F.; Niu, J.; Rodrigues, J.J. An efficient deep model for day-ahead electricity load forecasting with stacked denoising auto-encoders. *J. Parallel Distrib. Comput.* **2018**, *117*, 267–273. [[CrossRef](#)]
20. Metaxiotis, K.; Kagiannas, A.; Askounis, D.; Psarras, J. Artificial intelligence in short term electric load forecasting: A state-of-the-art survey for the researcher. *Energy Conv. Manag.* **2003**, *44*, 1525–1534. [[CrossRef](#)]
21. Kim, H.C.; Pang, S.; Je, H.M.; Kim, D.; Bang, S.Y. Constructing support vector machine ensemble. *Pattern Recogn.* **2003**, *36*, 2757–2767. [[CrossRef](#)]
22. Buntine, W.L. Bayesian backpropagation. *Complex Syst.* **1991**, *5*, 603–643.
23. MacKay, D.J.; Mac Kay, D.J. *Information Theory, Inference and Learning Algorithms*; Cambridge University Press: Cambridge, UK, 2003.
24. Neal, R.M. *Bayesian Training of Backpropagation Networks by the Hybrid Monte Carlo Method*; Technical Report; University of Toronto: Toronto, ON, Canada, 1992.
25. Snoek, J.; Larochelle, H.; Adams, R.P. Practical bayesian optimization of machine learning algorithms. *Adv. Neural Inf. Process. Syst.* **2012**, *25*.
26. Alizadeh, B.; Bafti, A.G.; Kamangir, H.; Zhang, Y.; Wright, D.B.; Franz, K.J. A novel attention-based LSTM cell post-processor coupled with bayesian optimization for streamflow prediction. *J. Hydrol.* **2021**, *601*, 126526. [[CrossRef](#)]
27. Ma, J.; Ding, Y.; Cheng, J.C.; Jiang, F.; Gan, V.J.; Xu, Z. A Lag-FLSTM deep learning network based on Bayesian Optimization for multi-sequential-variant PM2.5 prediction. *Sustain. Cities Soc.* **2020**, *60*, 102237. [[CrossRef](#)]
28. Abbasimehr, H.; Paki, R. Prediction of COVID-19 confirmed cases combining deep learning methods and Bayesian optimization. *Chaos Solitons Fractals* **2021**, *142*, 110511. [[CrossRef](#)] [[PubMed](#)]
29. Zhang, W.; Chen, Q.; Yan, J.; Zhang, S.; Xu, J. A novel asynchronous deep reinforcement learning model with adaptive early forecasting method and reward incentive mechanism for short-term load forecasting. *Energy* **2021**, *236*, 121492. [[CrossRef](#)]

30. Wang, L.; Wang, Z.; Qu, H.; Liu, S. Optimal forecast combination based on neural networks for time series forecasting. *Appl. Soft Comput.* **2018**, *66*, 1–17. [[CrossRef](#)]
31. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. The M4 Competition: 100,000 time series and 61 forecasting methods. *Int. J. Forecast.* **2020**, *36*, 54–74. [[CrossRef](#)]
32. Pelikan, M. Hierarchical Bayesian optimization algorithm. In *Hierarchical Bayesian Optimization Algorithm*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 105–129.
33. Khan, N.; Goldberg, D.E.; Pelikan, M. Multi-objective Bayesian optimization algorithm. In Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation, New York, NY, USA, 9–13 July 2002; p. 684.
34. Schwarz, J.; Ocenasek, J. A problem knowledge-based evolutionary algorithm KBOA for hypergraph bisectioning. In Proceedings of the 4th Joint Conference on Knowledge-Based Software Engineering, Brno, Czech Republic, 16–21 June 2000; IOS Press: Amsterdam, The Netherlands, 2000; pp. 51–58.
35. Liu, D.; Zeng, L.; Li, C.; Ma, K.; Chen, Y.; Cao, Y. A distributed short-term load forecasting method based on local weather information. *IEEE Syst. J.* **2016**, *12*, 208–215. [[CrossRef](#)]
36. Shi, H.; Xu, M.; Li, R. Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Trans. Smart Grid* **2017**, *9*, 5271–5280. [[CrossRef](#)]
37. Kong, W.; Dong, Z.Y.; Hill, D.J.; Luo, F.; Xu, Y. Short-term residential load forecasting based on resident behaviour learning. *IEEE Trans. Power Syst.* **2017**, *33*, 1087–1088. [[CrossRef](#)]
38. Huang, X.; Hong, S.H.; Li, Y. Hour-ahead price based energy management scheme for industrial facilities. *IEEE Trans. Ind. Inf.* **2017**, *13*, 2886–2898. [[CrossRef](#)]
39. Van der Meer, D.W.; Munkhammar, J.; Widén, J. Probabilistic forecasting of solar power, electricity consumption and net load: Investigating the effect of seasons, aggregation and penetration on prediction intervals. *Solar Energy* **2018**, *171*, 397–413. [[CrossRef](#)]
40. Carvallo, J.P.; Larsen, P.H.; Sanstad, A.H.; Goldman, C.A. Long term load forecasting accuracy in electric utility integrated resource planning. *Energy Policy* **2018**, *119*, 410–422. [[CrossRef](#)]
41. Wang, P.; Liu, B.; Hong, T. Electric load forecasting with recency effect: A big data approach. *Int. J. Forecast.* **2016**, *32*, 585–597. [[CrossRef](#)]
42. Gavrilas, M. *Heuristic and Metaheuristic Optimization Techniques with Application to Power Systems*; Technical University of Iasi: Iasi, Romania, 2010.
43. Binitha, S.; Sathya, S.S. A survey of bio inspired optimization algorithms. *Int. J. Soft Comput. Eng.* **2012**, *2*, 137–151.
44. Akbaripour, H.; Masehian, E. Efficient and robust parameter tuning for heuristic algorithms. *Int. J. Ind. Eng. Prod. Res.* **2013**, *24*, 143–150.
45. Raza, M.Q.; Khosravi, A. A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renew. Sustain. Energy Rev.* **2015**, *50*, 1352–1372. [[CrossRef](#)]
46. Yu, F.; Xu, X. A short-term load forecasting model of natural gas based on optimized genetic algorithm and improved BP neural network. *Appl. Energy* **2014**, *134*, 102–113. [[CrossRef](#)]
47. Liao, G.C. Hybrid improved differential evolution and wavelet neural network with load forecasting problem of air conditioning. *Int. J. Electr. Power Energy Syst.* **2014**, *61*, 673–682.
48. Jawad, M.; Ali, S.M.; Khan, B.; Mehmood, C.A.; Farid, U.; Ullah, Z.; Usman, S.; Fayyaz, A.; Jadoon, J.; Tareen, N.; et al. Genetic algorithm-based non-linear auto-regressive with exogenous inputs neural network short-term and medium-term uncertainty modelling and prediction for electrical load and wind speed. *J. Eng.* **2018**, *2018*, 721–729. [[CrossRef](#)]
49. Huyghues-Beaufond, N.; Tindemans, S.; Falugi, P.; Sun, M.; Strbac, G. Robust and automatic data cleansing method for short-term load forecasting of distribution feeders. *Appl. Energy* **2020**, *261*, 114405. [[CrossRef](#)]
50. Cai, M.; Pipattanasomporn, M.; Rahman, S. Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. *Appl. Energy* **2019**, *236*, 1078–1088. [[CrossRef](#)]
51. He, F.; Zhou, J.; Feng, Z.k.; Liu, G.; Yang, Y. A hybrid short-term load forecasting model based on variational mode decomposition and long short-term memory networks considering relevant factors with Bayesian optimization algorithm. *Appl. Energy* **2019**, *237*, 103–116. [[CrossRef](#)]
52. Wu, Z.; Zhao, X.; Ma, Y.; Zhao, X. A hybrid model based on modified multi-objective cuckoo search algorithm for short-term load forecasting. *Appl. Energy* **2019**, *237*, 896–909. [[CrossRef](#)]
53. Vrablcová, P.; Ezzeddine, A.B.; Rozinajová, V.; Šárik, S.; Sangaiah, A.K. Smart grid load forecasting using online support vector regression. *Comput. Electr. Eng.* **2018**, *65*, 102–117.
54. Li, Y.; Che, J.; Yang, Y. Subsampled support vector regression ensemble for short term electric load forecasting. *Energy* **2018**, *164*, 160–170. [[CrossRef](#)]
55. Zhang, F.; Deb, C.; Lee, S.E.; Yang, J.; Shah, K.W. Time series forecasting for building energy consumption using weighted Support Vector Regression with differential evolution optimization technique. *Energy Build.* **2016**, *126*, 94–103. [[CrossRef](#)]
56. Cao, G.; Wu, L. Support vector regression with fruit fly optimization algorithm for seasonal electricity consumption forecasting. *Energy* **2016**, *115*, 734–745. [[CrossRef](#)]
57. Kavousi-Fard, A.; Samet, H.; Marzbani, F. A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting. *Exp. Syst. Appl.* **2014**, *41*, 6047–6056. [[CrossRef](#)]

58. Xiao, L.; Shao, W.; Wang, C.; Zhang, K.; Lu, H. Research and application of a hybrid model based on multi-objective optimization for electrical load forecasting. *Appl. Energy* **2016**, *180*, 213–233. [[CrossRef](#)]
59. Amjady, N.; Keynia, F.; Zareipour, H. Short-term load forecast of microgrids by a new bilevel prediction strategy. *IEEE Trans. Smart Grid* **2010**, *1*, 286–294. [[CrossRef](#)]
60. Hafeez, G.; Alimgeer, K.S.; Wadud, Z.; Shafiq, Z.; Ali Khan, M.U.; Khan, I.; Khan, F.A.; Derhab, A. A novel accurate and fast converging deep learning-based model for electrical energy consumption forecasting in a smart grid. *Energies* **2020**, *13*, 2244. [[CrossRef](#)]
61. Zhang, Z.; Ding, S.; Sun, Y. A support vector regression model hybridized with chaotic krill herd algorithm and empirical mode decomposition for regression task. *Neurocomputing* **2020**, *410*, 185–201. [[CrossRef](#)]
62. Zeng, N.; Zhang, H.; Liu, W.; Liang, J.; Alsaadi, F.E. A switching delayed PSO optimized extreme learning machine for short-term load forecasting. *Neurocomputing* **2017**, *240*, 175–182. [[CrossRef](#)]
63. Ghadimi, N.; Akbarimajd, A.; Shayeghi, H.; Abedinia, O. Two stage forecast engine with feature selection technique and improved meta-heuristic algorithm for electricity load forecasting. *Energy* **2018**, *161*, 130–142. [[CrossRef](#)]
64. Shiri, A.; Afshar, M.; Rahimi-Kian, A.; Maham, B. Electricity price forecasting using Support Vector Machines by considering oil and natural gas price impacts. In Proceedings of the 2015 IEEE International Conference on Smart Energy Grid Engineering (SEGE), Oshawa, ON, Canada, 17–19 August 2015; pp. 1–5.
65. Jiang, H.; Zhang, Y.; Muljadi, E.; Zhang, J.J.; Gao, D.W. A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization. *IEEE Trans. Smart Grid* **2016**, *9*, 3341–3350. [[CrossRef](#)]
66. Fung, C.P. Manufacturing process optimization for wear property of fiber-reinforced polybutylene terephthalate composites with grey relational analysis. *Wear* **2003**, *254*, 298–306. [[CrossRef](#)]
67. Julong, D. Introduction to grey system theory. *J. Grey Syst.* **1989**, *1*, 1–24.
68. Deng, J.L. *A Course on Grey System Theory*; Huazhong University of Science and Technology Press: Wuhan, China, 1990.
69. Deng, J. *The Essential Methods of Grey Systems*; Huazhong University of Science and Technology Press: Wuhan, China, 1992.
70. Nabavi Karizi, S.; Kabir, E. A Two-Stage Method for Classifiers Combination. *Nashriyyah-i Muhandisi-i Barq va Muhandisi-i Kampyutar-i Iran* **2008**, *1*, 63.
71. Storn, R.; Price, K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [[CrossRef](#)]
72. Du, Z.; Wang, X.; Zheng, L.; Zheng, Z. Nonlinear system modeling based on KPCA and MKSVM. In Proceedings of the 2009 ISECS International Colloquium on Computing, Communication, Control, and Management, Sanya, China, 8–9 August 2009; IEEE: New York, NY, USA, 2009; Volume 3, pp. 61–64.
73. Ghofrani, M.; Ghayekhloo, M.; Arabali, A.; Ghayekhloo, A. A hybrid short-term load forecasting with a new input selection framework. *Energy* **2015**, *81*, 777–786. [[CrossRef](#)]
74. Neill, S.P.; Hashemi, M.R. *Fundamentals of Ocean Renewable Energy: Generating Electricity from the Sea*; Academic Press: Cambridge, MA, USA, 2018.
75. Woolf, B.P. *Building Intelligent Interactive Tutors: Student-Centered Strategies for Revolutionizing e-Learning*; Morgan Kaufmann: Burlington, MA, USA, 2010.
76. Fox, E.P. *Data Analysis: A Bayesian Tutorial*; OUP Oxford: Oxford, UK, 1998.
77. MacKay, D.J. A practical Bayesian framework for backpropagation networks. *Neural Comput.* **1992**, *4*, 448–472. [[CrossRef](#)]
78. Ford, W. *Numerical Linear Algebra with Applications: Using MATLAB*; Academic Press: Cambridge, MA, USA, 2014.
79. MacKay, D.J. Probable networks and plausible predictions—A review of practical Bayesian methods for supervised neural networks. *Netw. Comput. Neural Syst.* **1995**, *6*, 469–505. [[CrossRef](#)]
80. Hafeez, G.; Islam, N.; Ali, A.; Ahmad, S.; Usman, M.; Saleem Alimgeer, K. A modular framework for optimal load scheduling under price-based demand response scheme in smart grid. *Processes* **2019**, *7*, 499. [[CrossRef](#)]
81. Ahmad, A.; Javaid, N.; Guizani, M.; Alrajeh, N.; Khan, Z.A. An accurate and fast converging short-term load forecasting model for industrial applications in a smart grid. *IEEE Trans. Ind. Inform.* **2016**, *13*, 2587–2596. [[CrossRef](#)]
82. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.