NoSPF: Non-Stationary Long-Term Power Consumption Forecasting for Servers in Cloud Data Centers

Ruichao Mo, Weiwei Lin, *Senior Member, IEEE*, Shengsheng Lin, Simon Fong, and Keqin Li, *Fellow, IEEE*

***Abstract*—Accurately forecasting power consumption in data center servers requires addressing the temporal distribution shift caused by dynamic resource demands. However, existing methods rely on global normalization, which cannot capture short-term localized shift, leading to unsatisfactory performance when forecasting non-stationary time series. To address this challenge, we propose a novel bi-level optimization framework for forecasting non-stationary long-term power consumption, named *NoSPF*. The framework employs hierarchical optimization to separately model the stationary and local non-stationary components of power consumption time series, offering a flexible, model-agnostic paradigm for time-series forecasting. Using Discrete Wavelet Transform (DWT) for multi-scale time–frequency analysis, *NoSPF* decomposes the series into non-stationary components driven by short-term fluctuations and stationary components that capture long-term trends. Furthermore, *NoSPF* integrates a lightweight Multi-Layer Perceptron (MLP) to predict the local non-stationary components, enhancing the framework's forecasting accuracy by providing more precise approximations of the future power distribution. Extensive experiments on real-world server datasets demonstrate the superior performance and effectiveness of *NoSPF*.**

***Index Terms*—Non-stationary time-series, cloud data center, power data mining, predictive modeling.**

Received 22 February 2025; revised 29 August 2025; accepted 19 October 2025. Date of publication 23 October 2025; date of current version 11 December 2025. This work was supported in part by Guangdong Provincial Natural Science Foundation Project under Grant 2025A1515010113, in part by Guangzhou Development Zone Science and Technology Project under Grant 2023GH02, in part by Guangxi Key Research and Development Project under Grant 2024AB02018, in part by Shandong Provincial Natural Science Foundation Project under Grant ZR2024LZH012, and in part by the Major Key Project of PCL, China under Grant PCL2025AS11. Recommended for acceptance by Y. Hu. *(Corresponding author: Weiwei Lin.)*

Ruichao Mo and Shengsheng Lin are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China (e-mail: cs_moruichao@mail.scut.edu.cn; cslinshengsheng@mail.scut.edu.cn).

Weiwei Lin is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China, and also with Pengcheng Laboratory, Shenzhen 518066, China (e-mail: linww@scut.edu.cn).

Simon Fong is with the Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mail: ccfong@um.edu.mo).

Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561, USA (e-mail: lik@newpaltz.edu).

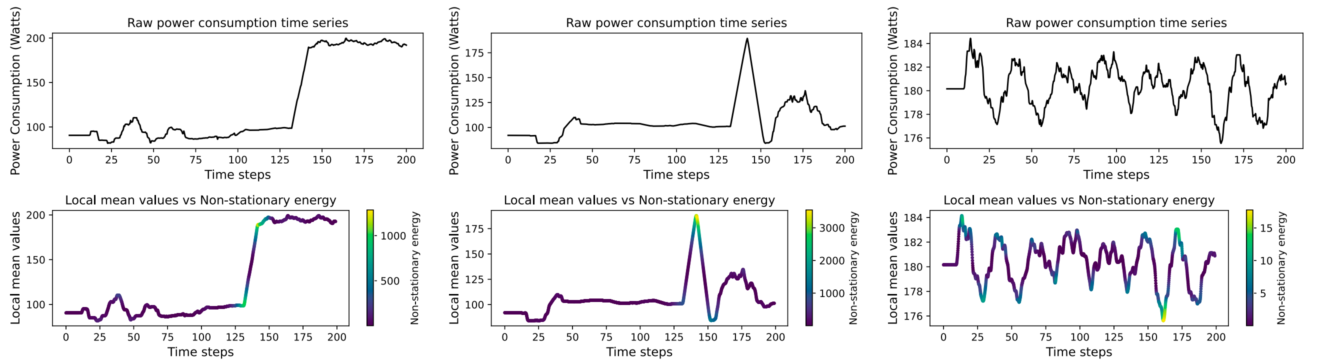
Digital Object Identifier 10.1109/TC.2025.3624486

I. Introduction

With the scale of data centers expanding rapidly, their carbon footprint has increased substantially, now accounting for approximately 30% of global carbon emissions [1]. This issue has become a pressing concern for major cloud service providers such as Google and Microsoft, both of which have committed to ambitious targets for significantly reducing data center carbon emissions by 2030 [2], [3]. In pursuit of these goals, data center operators have implemented various strategies, including optimizing the energy efficiency of non-IT equipment [4], [5], [6] and integrating renewable energy sources to power data center operations [7], [8], [9], [10]. Despite these efforts, the operation of massive servers, often under low utilization, continues to consume substantial amounts of energy, constituting the majority of total energy consumption in data centers. This highlights the energy management of servers as a critical challenge for achieving energy efficiency and reducing emissions in data center operations.

For optimizing server energy consumption, existing research is mostly devoted to improving the utilization of computing resources, such as task scheduling [11], [12] and resource allocation [13], to name a few. Notably, these studies require establishing the correlation between resource utilization and energy consumption of the server at first, which serves as the purpose of power consumption models. Besides, it also plays a crucial role for data center administrators to learn and understand the long-term power behavior of servers for overall energy plans of data centers [14]. This understanding is particularly significant in the era of large language models, which demand extensive computational resources and pose additional challenges to energy planning in data centers [15]. Thus, the power consumption modeling of servers in data centers has garnered significant attention in recent years [16], [17].

However, existing research still faces several critical challenges. On the one hand, most approaches model real-time server power consumption based on resource utilization by monitoring current resource usage, but they fail to capture and analyze long-term power characteristics, limiting their applicability for long-term energy planning in data centers. On the other hand, as shown in Fig. 1, highly dynamic resource requests introduce significant non-stationary behavior in



(a) Server (r10n10) (b) Server (r10n12) (c) Server (r12n13)

Fig. 1. Example of variation in local mean of power consumption values for different servers (averaged across every 10 time steps). the local mean power consumption values of different servers exhibit varying degrees of fluctuation over time. Notably, the regions where the local mean power consumption values undergo significant changes overlap with the high-energy regions of the local non-stationary components. *This observation highlights that the local non-stationary components are the primary drivers of temporal distribution shift.* Therefore, mitigating the impact of localized non-stationary components is key to effectively mining long-term power consumption patterns in time-series data, which is essential for accurate power consumption prediction.

power consumption, known as temporal distribution shift [18]. Addressing this issue requires eliminating the non-stationary characteristics of power consumption to extract meaningful long-term patterns from historical data.

Reversible instance normalization methods based on statistical features (e.g., mean and variance) of time-series data have recently been introduced as effective solutions for addressing the challenges posed by temporal distribution shift in long-term time-series forecasting [19], [20]. These methods mitigate the impact of temporal distribution shift by removing the non-stationary components of the input time-series data, enabling the model to better capture latent power consumption patterns. Additionally, the inverse normalization of non-stationary features applied to the model's stationary forecasting further enhances accuracy. However, these methods typically rely on the global statistical features of the input time-series, limiting their effectiveness in handling temporal distribution shifts arising from locally abrupt variations in statistical properties. Such abrupt changes are frequently observed in the power consumption patterns of servers in real-world data centers, posing a significant challenge to the applicability of these techniques in this context.

To address these challenges, we propose a novel bi-level optimization framework for NOn- Stationary long-term Power consumption Forecasting, named *NoSPF*. The framework separates stationary and local non-stationary components of the power series through hierarchical optimization, providing a flexible data-driven paradigm independent of specific forecasting models. Using Discrete Wavelet Transform (DWT) for multi-scale time–frequency analysis, *NoSPF* decomposes the series into short-term non-stationary fluctuations and long-term stationary patterns. A lightweight multi-layer perceptron (MLP) is then employed to predict the non-stationary components, which are integrated with stationary forecasts to better approximate the true power consumption distribution and improve long-term prediction accuracy.

The main contributions of this paper are summarized as follows:

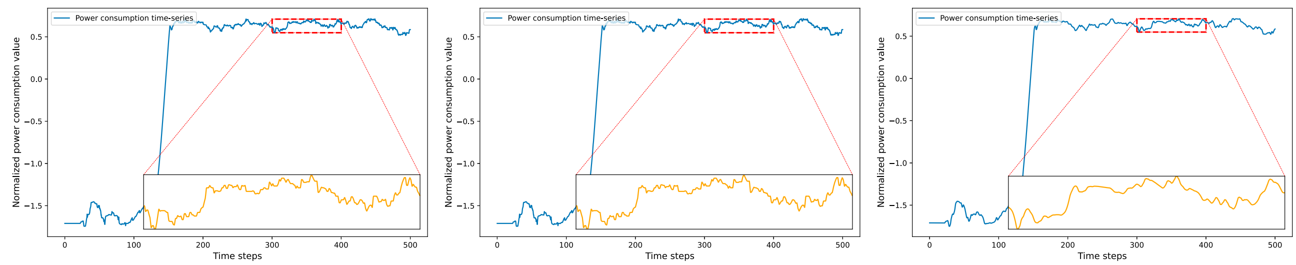
- We propose a novel bi-level optimization framework for NOn- Stationary long-term Power consumption Forecasting, called *NoSPF*. As a model-agnostic framework, *NoSPF* offers a flexible data-driven paradigm that is compatible with various time-series forecasting models.
- To effectively separate local non-stationary components, the framework employs DWT for multi-scale time-frequency analysis, extracting critical stationary components to improve the modeling of long-term power consumption patterns.
- Leveraging *NoSPF*'s powerful decomposition capability, it employs a lightweight MLP to accurately forecast local non-stationary components, achieving an optimal balance between forecasting accuracy and computational efficiency.

The remainder of the paper is organized as follows. Section II discusses the motivation and problem formulation. Section III presents the design details of the proposed framework. Section IV describes the experimental setup and experimental results. Section V reviews the related work, and Section VI provides the conclusion and future work.

II. Motivation and Problem Formulation

A. Motivation

1) Non-Stationary Power Consumption Behavior of Server Arising from Highly Dynamic Resource Requests: In real-world data centers, servers execute assigned computing tasks in accordance with the scheduling strategies defined by the cluster scheduler. This results in highly dynamic and stochastic resource consumption patterns. Additionally, factors like workload variations, hardware replacement, and environmental fluctuations further contribute to changes in the distribution of server power consumption. Consequently, the power consumption of servers exhibits significant non-stationary behavior over time, characterized by temporal distribution shift, such as dynamic changes in statistical properties (e.g., mean, variance).



(a) Raw power consumption time-series data. (b) Power consumption time-series data processed by RevIN. (c) Power consumption time-series data processed by DWT.

Fig. 2. Examples of server power time-series data processed by different non-stationary removal methods. It is evident that the *local non-stationary components* present in the server time-series power consumption data can be effectively reduced by DWT, which explains why DWT is exploited to implement non-stationary time-series power consumption forecasting.

Fig. 1 illustrates the variation in the average power consumption values across different servers, where larger variations indicate more pronounced temporal distribution shifts. To further investigate the relationship between local non-stationary components and temporal distribution shifts, we first decompose the time-series power consumption data using DWT, extracting the high-frequency portion as the non-stationary component, which captures abrupt local changes. Additionally, we visualize the correlation between changes in the energy of the non-stationary component and fluctuations in the local mean. The results show that regions with higher energy in the non-stationary component (represented by brighter colors) coincide with areas where the local mean undergoes significant changes. This emphasizes the pivotal role of non-stationary components in driving fluctuations in the mean of time-series power consumption data. This observation underscores that the non-stationary component is a primary contributor to distribution shift, making it a central challenge for achieving accurate long-term power consumption forecasting. Addressing the issues arising from these non-stationary components forms the core motivation of this work.

2) Non-Stationary Separation of Time-Series Power Data Based on Multi-Scale Time-Frequency Analysis: Currently, most methods for addressing non-stationary time-series forecasting, such as RevIN, are based on the statistical features of the time series and normalize the global characteristics of the input data. While these methods effectively address global trends in the input time series, they face significant limitations in handling finer-grained components, particularly when dealing with server power behavior influenced by short-term workload fluctuations and hardware state changes. As illustrated in Fig. 2(a) and 2(b), the raw time-series power consumption data and the corresponding data normalized using RevIN [19] are presented. While RevIN successfully normalizes the global characteristics of the time series, it fails to eliminate non-stationary features arising from short-term power consumption variations.

The multi-scale analysis capability of DWT in the time-frequency domain can be leveraged to effectively capture short-term abrupt changes in server power consumption, facilitating the separation of long-term trends from short-term fluctuations in the input time-series data. Fig. 2(c) demonstrates the results of applying the discrete wavelet transform to power consumption data, showing a smoother long-term trend alongside the successful separation of local short-term fluctuations. These observations motivate us to implement non-stationary separation of server power consumption time-series data using DWT.

B. Problem Formulation

Let the server power consumption data collected over $\mathbf{T}$ time steps be represented as $\mathbf{X_T} = [x_1, x_2, \ldots, x_\mathbf{T}]$, where $x_t \in \mathbb{R}$ denotes the power consumption at time step $t \in \mathbf{T}$. For any given time step t, suppose that $\mathbf{x}_t^L$ denote a input sequence of length L, specifically,

$$\mathbf{x}_t^L = [x_{t-L}, x_{t-L+1}, \ldots, x_{t-1}]. \tag{1}$$

Similarly, let $\mathbf{y}_t^H$ represent the future sequence of length H, denoted as,

$$\mathbf{y}_t^H = [x_t, x_{t+1}, \ldots, x_{t+H-1}]. \tag{2}$$

In the classical univariate time series forecasting task, the objective is to predict future values based on the historical sequence $\mathbf{x}_t^L$, which can be expressed as

$$\hat{\mathbf{y}}_t^H = F_\phi(\mathbf{x}_t^L), \tag{3}$$

where $F_\theta(\cdot) : \mathbb{R}^L \to \mathbb{R}^H$ represents the forecasting model, and θ denotes the model parameters. $\hat{\mathbf{y}}_t^H$ denotes the forecasting values. For simplicity of expression, $\mathbf{x}_t$ and $\mathbf{y}_t$ are used below in place of $\mathbf{x}_t^L$ and $\mathbf{y}_t^H$.

Furthermore, the main objective function of unvariate time series forecasting task is denoted as following:

$$\underset{\phi}{argmin} \sum_{\mathbf{x}_t, \mathbf{y}_t \subseteq \mathbf{X_T}} \mathcal{L}(F_\phi(\mathbf{x}_t), \mathbf{y}_t), \tag{4}$$

where $\mathcal{L}(\cdot)$ represents the loss function.

To address the dual requirements of non-stationary power time series forecasting, eliminating the non-stationary components that impair long-term forecasting performance while simultaneously enabling accurate long-term forecasting based on the stationary component of the input data, we reformulate the classical unvariate time series forecasting optimization problem detailed in Eq. 4 as a bi-level optimization problem, expressed as follows:

$$\underset{\phi}{argmin} \sum_{\mathbf{x}_t, \mathbf{y}_t \subseteq \mathbf{X_T}} \mathcal{L}_{sta}(\phi, \theta^*, (F_\phi(\mathbf{x}_t), \mathbf{y}_t)), \tag{5}$$

$$\text{s.t.} \quad \theta^* = \underset{\theta}{argmin} \sum_{\mathbf{x}_t, \mathbf{y}_t \subseteq \mathbf{X_T}} \mathcal{L}_{shift}(\phi, \theta, (g_\theta(\mathbf{x}_t), \mathbf{y}_t)), \tag{6}$$

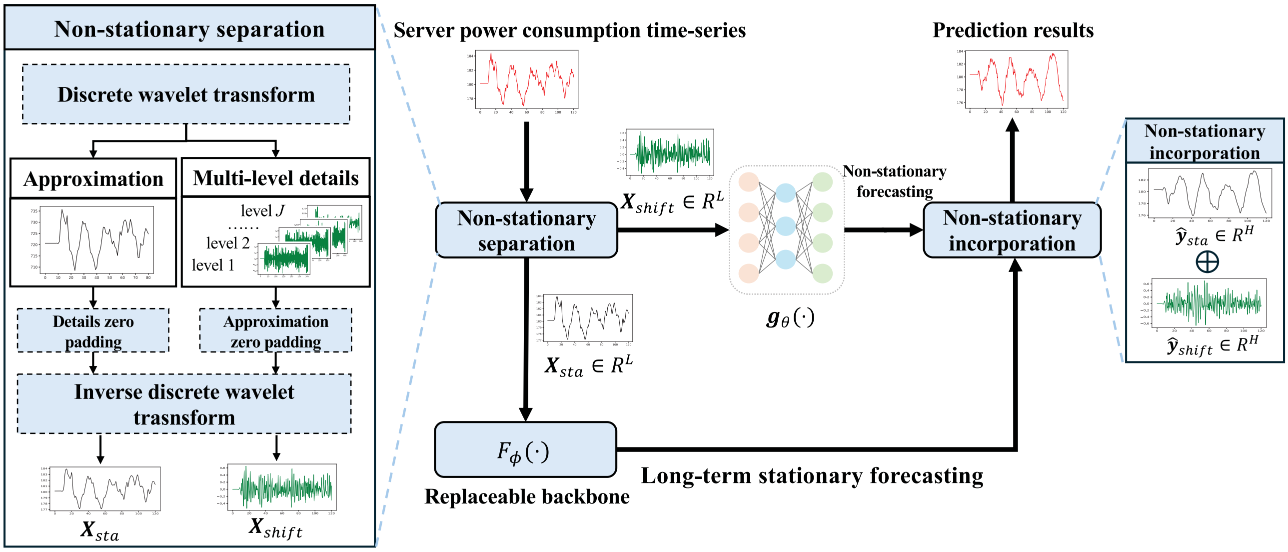


Fig. 3. The illustration of the proposed framework *NoSPF*. *NoSPF* adopts hierarchical optimization that separates the learning of stationary and non-stationary components of the input power consumption time-series, resulting in a flexible data-driven paradigm for non-stationary long-term power forecasting. This design does not depend on any specific time-series forecasting model, allowing for more accurate non-stationary long-term power consumption forecasting for servers in the cloud data center.

where $g_\theta(\cdot)$ represents the non-stationary components forecasting model. In this bi-level optimization framework, the inner loop focuses on capturing long-term stationary components by mitigating the effects of distributional shifts, ensuring the forecasting model captures latent long-term patterns for reliable and accurate predictions. Meanwhile, the outer loop focuses on learning the non-stationary components of the server power consumption time series, which contribute to temporal distribution shifts. By modeling and predicting these variations, the framework enhances the model's adaptability to volatile scenarios. Both optimization levels are easily implemented through data-driven models, making the approach practical and versatile. This bi-level optimization design delivers a model-agnostic solution for forecasting non-stationary power time series by effectively decoupling stationary and non-stationary components.

III. Design of *NoSPF*

In this section, we present *NoSPF*, a framework designed for non-stationary long-term power consumption forecasting. The framework addresses the challenges posed by local non-stationary components in server power consumption, significantly improving the accuracy of long-term predictions. *NoSPF* operates in three key phases: non-stationary separation, non-stationary forecasting, and non-stationary incorporation. The overall architecture of the *NoSPF* framework is depicted in Fig. 3, and the complete training process is outlined in Algorithm 1.

A. Primary Knowledge of DWT

Wavelet transform facilitates the analysis of time-series data in the time-frequency domain across multiple scales, overcoming the limitations of the Fourier transform, which are restricted to frequency domain analysis. This makes wavelet transform particularly effective for analyzing non-stationary time-series data. Its efficacy depends significantly on the selection of wavelet basis functions, mathematically defined as:

$$\varphi_{\alpha,\beta}(t) = \frac{1}{\sqrt{\alpha}}\varphi\left(\frac{t-\beta}{\alpha}\right), \tag{7}$$

where $\alpha \neq 0$ represents the scale factor, which controls the shrinkage of the wavelet basis function, and $\beta \in \mathbb{R}$ represents the translation factor.

Discrete Wavelet Transform (DWT) is a specific form of wavelet transform that discretizes the scale and translation parameters of the wavelet basis function. It decomposes signals into approximation coefficients and detail coefficients at different scales, making it suitable for analyzing time-series data at multiple resolutions. For a given time-series data $\mathbf{x}_t$, the DWT can be defined as:

$$DWT_{j,k}(\mathbf{x}_t) = \int_{-\infty}^{+\infty} \mathbf{x}_t \psi_{j,k}(t)dt, j \geq 0, k \in \mathbb{Z}, \tag{8}$$

where j and k denote the scale and translation parameters, respectively. The scale parameter j controls the level of decomposition: higher values of j correspond to coarser approximations that capture low-frequency, stationary patterns, while lower values focus on finer-grained, high-frequency components where non-stationary fluctuations typically reside. The translation parameter k determines the temporal positioning of the wavelet basis function, allowing DWT to localize transient changes in time. The wavelet basis function $\psi_{j,k}(t)$ is defined as:

$$\psi_{j,k}(t) = 2^{-\frac{j}{2}}\Psi(2^{-j}t - k). \tag{9}$$

Algorithm 1: Training Procedure of *NoSPF*.

Input: Input power series $\mathbf{X_T} = \{\mathbf{x}_t\}_{t=1}^{\mathbf{T}}$, future power series $\mathbf{Y_T} = \{\mathbf{y}_t\}_{t=1}^{\mathbf{T}}$, decomposition level j of DWT;
Output: Trained model $F_{\phi_*}(\cdot)$, $g_{\theta*}(\cdot)$;
Initialization: Initialize parameters ϕ, θ;
while *not converge* **do**
  foreach $\mathbf{x}_t \in \mathbf{X_T}$, $\mathbf{y}_t \in \mathbf{Y_T}$ **do**
    Decompose $\mathbf{x}_t$ by Eq.14 to obtain $\mathbf{z}_a$ and $\mathbf{z}_d$;
    Zero-padding $\mathbf{z}_d$, $\mathbf{z}_a$, respectively, to obtain $\mathbf{c}_l$ and $\mathbf{c}_h$;
    Calculate the stationary component $\mathbf{x}_{sta}$ according to the Eq.15 ;
    Calculate the non-stationary component $\mathbf{x}_{shift}$ according to the Eq.16;
    Predict future non-stationary component $\hat{\mathbf{y}}_{shift}$ based on $\hat{\mathbf{x}}_{shift}$ by $g_\theta(\cdot)$;
    Update parameters θ based on loss function $\mathcal{L}_{shift}(\cdot)$;
  end
  ; `// Training of the non-stationary forecasting model` $g_\theta(\cdot)$
end
while *not converge* **do**
  foreach $\mathbf{x}_t \in \mathbf{X_T}$, $\mathbf{y}_t \in \mathbf{Y_T}$ **do**
    Decompose $\mathbf{x}_t$ by Eq.14 to obtain $\mathbf{z}_a$ and $\mathbf{z}_d$;
    Zero-padding $\mathbf{z}_d$, $\mathbf{z}_a$, respectively, to obtain $\mathbf{c}_l$ and $\mathbf{c}_h$;
    Calculate the stationary component $\mathbf{x}_{sta}$ according to the Eq.15 ;
    Calculate the non-stationary component $\mathbf{x}_{shift}$ according to the Eq.16;
    Predict future non-stationary component $\hat{\mathbf{y}}_{shift}$ based on $\hat{\mathbf{x}}_{shift}$ by $g_{\theta_*}(\cdot)$; `// Trained` $g_{\theta_*}(\cdot)$
    Predict future stationary component $\hat{\mathbf{y}}_{sta}$ based on $\hat{\mathbf{x}}_{sta}$ by $F_\phi(\cdot)$;
    Incorporation $\hat{\mathbf{y}}_{shift}$ into $\hat{\mathbf{y}}_{sta}$ using Eq. 23;
    Update parameters ϕ based on loss function $\mathcal{L}_{sta}(\cdot)$;
  end
  ; `// Training of the stationary forecasting model` $F_\phi(\cdot)$
end

Furthermore, $\mathbf{x_t}$ can be expressed as the sum of approximation coefficients and detail coefficients across different scales j and translations k. Specifically:

$$DWT_{j,k}(\mathbf{x}_t) = A_{j,k} + D_{j,k}, \tag{10}$$

where

- $A_{j,k}$ represents the low-frequency approximation coefficients, capturing the long-term trends in the data.
- $D_{j,k}$ represents the high-frequency detail coefficients, capturing localized variations at finer scales.

The coefficients can be computed as follows:

$$A_{j,k} = 2^{-\frac{j}{2}} \int_{-\infty}^{+\infty} \mathbf{x}_t \psi_{j,k}(t) dt, \tag{11}$$

$$D_{j,k} = \int_{-\infty}^{+\infty} \mathbf{x}_t \Psi_{j,k}(t) dt. \tag{12}$$

Finally, $\mathbf{x}_t$ can be represented as a coefficient tuple $\mathbf{c} = (A_j, D_1, D_2, \ldots, D_j)$ after applying DWT. The Inverse DWT (IDWT) reconstructs $\mathbf{x}_t$ from its coefficients. The mathematical expression for IDWT is given as:

$$\hat{\mathbf{x}}_t = IDWT(\mathbf{c}) = \sum_{j,k} A_{j,k} \cdot \psi_{j,k}(t) + \sum_{j,k} D_{i,k} \Psi_{j,k}(t). \tag{13}$$

B. Non-Stationary Separation

To effectively capture both the stationary and non-stationary characteristics of the input power consumption time series, *NoSPF* employs DWT to decompose $\mathbf{x}_t$. The decomposition can be expressed as:

$$\mathbf{z}_a, \mathbf{z}_d = DWT(\mathbf{x}_t), \tag{14}$$

where $\mathbf{z}_a$ represents the low-frequency component of the input time series, capturing the long-term stationary trends in the power consumption data. $\mathbf{z}_d$ contains the high-frequency components derived from multiple scales, corresponding to the non-stationary variations in $\mathbf{x}_t$.

In addition, by zero-padding $\mathbf{z}_d$, we can obtain the decomposition coefficient $\mathbf{c}_l = [\mathbf{z}_a, \mathbf{0}]$, which contains only the low-frequency components. This enables us to perform IDWT-based separation to extract the stationary component of the input $\mathbf{x}_t$, denoted as:

$$\hat{\mathbf{x}}_{sta} = IDWT(\mathbf{c}_l). \tag{15}$$

Similarly, by zero-padding the $\mathbf{z}_a$ and combining it with the $\mathbf{z}_d$, the decomposition coefficient $\mathbf{c}_h = [\mathbf{0}, \mathbf{z}_d]$ is obtained, which contains only the high-frequency components. Thus, we can extract the non-stationary component of the input time series using IDWT, represented as:

$$\hat{\mathbf{x}}_{shift} = IDWT(\mathbf{c}_h). \tag{16}$$

After the separation process, the stationary component of the input data $\mathbf{x}_t$ captures the long-term features more effectively, facilitating the use of arbitrary sophisticated time-series forecasting models to implement $F_\phi(\cdot)$ for accurate long-term power consumption forecasting.

C. Non-Stationary Forecasting

As defined by the outer loop of the bilevel optimization objective in Eq. 5, our goal is to learn the separated non-stationary component so that it can subsequently be integrated into the long-term forecasting results of the stationary component. This procedure ensures that the predicted power consumption distribution more closely aligns with the actual distribution. To this end, we construct a simple yet effective MLP model,

$g_\theta(\cdot)$, which predicts the future non-stationary component based on $\hat{\mathbf{x}}_{shift}$.

The choice of an MLP is motivated both by the characteristics of the non-stationary component and by practical considerations. First, due to the decomposition property of the discrete wavelet transform, the non-stationary components fed into $g_\theta(\cdot)$ primarily consist of localized high-frequency fluctuations with limited long-term dependency. Compared to the global stationary trend, such patterns exhibit lower complexity and can be effectively captured by a lightweight feedforward network. Second, the compact architecture of the MLP helps avoid unnecessary complexity and reduces the risk of overfitting to transient noise, which is particularly important for short-term fluctuations. These characteristics make it a practical and reliable choice for modeling non-stationary power consumption dynamics in data center environments. The main structure of $g_\theta(\cdot)$ is given by the following equation:

$$\mathbf{z}_s = Linear(\hat{\mathbf{x}}_{shift}), \tag{17}$$

$$\mathbf{z}_t = \mathbf{W}_1 * Linear(\mathbf{x}_t), \tag{18}$$

$$\mathbf{z} = Concat[\mathbf{z}_s; \mathbf{z}_t], \tag{19}$$

$$\mathbf{z} = Relu(\mathbf{z}), \tag{20}$$

$$\hat{\mathbf{y}}_{shift} = \mathbf{W}_2 * Linear(\mathbf{z}), \tag{21}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are learnable parameters, and Relu$(\cdot)$ is the activation function. Moreover, inspired by [21], considering that $\hat{\mathbf{x}}_{shift}$ includes only partial information from the original power consumption time series, we concatenate the original time series $\mathbf{x}_t$ with non-stationary component to implement a residual learning for the non-stationary component, further improving forecasting accuracy.

D. Non-Stationary Incorporation

Based on the stationary component $\hat{\mathbf{x}}_{sta}$ derived from $\hat{\mathbf{x}}_t$, the future power consumption forecasting can be performed by $F_\phi(\cdot)$, denoted as:

$$\hat{\mathbf{y}}_{sta} = F_\phi(\hat{\mathbf{x}}_{sta}). \tag{22}$$

Lastly, by performing a simple linear combination of the future stationary power consumption values predicted by the backbone model $F_\phi(\cdot)$ and the future non-stationary component predicted by $g_\theta(\cdot)$, the final predicted values of the server power consumption can be obtained, denoted as:

$$\hat{\mathbf{y}} = \hat{\mathbf{y}}_{sta} + \hat{\mathbf{y}}_{shift}. \tag{23}$$

IV. Experimental Evaluations

A. Experimental Settings

Dataset. To evaluate the time-series power performance of servers in real-world data centers and validate the effectiveness of the proposed method, we utilized two distinct datasets from the LISA cluster of the Dutch National Infrastructure and the Seren cluster provided by the Shanghai AI Lab.

The LISA dataset, which serves as a widely recognized benchmark, contains 327-dimensional server performance data collected from December 2019 to August 2020 using a range of performance counters. For our experiments, we selected four CPU servers (r10n10, r0n12, r12n13, r12n15) and four GPU servers (r30n5, r31n2, r31n3, r32n5), extracting 40 days of power consumption data from January 4, 2020, to February 13, 2020, as validation data. This dataset is publicly available, with further details provided in their repository[1] and the raw data accessible at this site[2].

To further assess the generalizability of *NoSPF*, we included an additional dataset from the Seren cluster, which captures the power consumption profiles of high-performance GPU servers used primarily for LLM development and deployment. Unlike the LISA cluster, which includes a mix of CPU and GPU servers, the Seren cluster exclusively comprises GPU servers, each equipped with 8 A100 GPUs, providing a more homogeneous testbed for evaluating the robustness of our approach. Data collection for this dataset spans from May 15, 2023, to August 9, 2023. For our experiments, we selected eight servers from this cluster, identified by their internal IP addresses: 10.140.0.147, 10.140.0.166, 10.140.0.246, 10.140.0.254, 10.140.1.78, 10.140.1.90, 10.140.1.103, and 10.140.1.119. For consistency in our analysis, these servers are referred to as Node-147, Node-166, Node-246, Node-254, Node-78, Node-90, Node-103, and Node-119. Further details about this dataset are available in[3].

All power consumption data were subjected to z-score normalization to ensure consistent scaling across the servers. It is important to note that while z-score normalization standardizes the data, it does not alter the original distribution of the time-series, and thus does not address challenges related to temporal distribution shift in long-term forecasting tasks [22]. The dataset for each server was divided into training, testing, and validation sets in a 7:1:2 ratio.

Evaluation Metrics. We selected two evaluation metrics, Mean Squared Error (MSE) and Mean Absolute Error (MAE), to quantify the performance of the model. These are defined as follows:

$$\text{MSE} = \frac{1}{M}\sum_{m=1}^{M}\left(\mathbf{y}_m - \hat{\mathbf{y}}\right)^2, \tag{24}$$

$$\text{MAE} = \frac{1}{M}\sum_{m=1}^{M}\left|\mathbf{y}_m - \hat{\mathbf{y}}\right|, \tag{25}$$

where M is the number of test data sample size, $\mathbf{y}_m$ is the true value of server power consumption.

Backbones. Since the proposed *NoSPF* is model-agnostic, we selected several existing time series forecasting models as backbones $F_\phi(\cdot)$ to predict the long-term stationary component of server power consumption. These include DLinear [23], SCINet [24], TiDE [25], and TSMixer [26]. All of the above models were implemented based on the releases in this repository[4].

[1]https://github.com/sara-nl/SURFace
[2]https://zenodo.org/records/3878143#.ZCFNIMpBwQ-
[3]https://github.com/InternLM/AcmeTrace
[4]https://github.com/thuml/Time-Series-Library

Methods for addressing temporal distribution shift. We selected a range of normalization methods designed to address temporal distribution shift. These methods are listed as follows:

- Reversible instance normalization (RevIN) [19][5]
- Dish-TS [20][6]
- Slicing adaptive normalization (SAN) [21][7]
- Frequency adaptive network (FAN) [27][8]

Additionally, two model focused on long-term forecasting under temporal distribution shift scenario, were also implemented for comparison, listed as follow:

- Non-stationary Transformer [28] 4
- Koopa [29] 4

Experimental details. All experiments were conducted on a server equipped with 2x Intel Xeon Silver 4316 CPU @ 2.30GHz, 252GB RAM, and 2x NVIDIA RTX 4090 GPU (24GB). The software environment used is Ubuntu 22.04.4 LTS, with PyTorch 2.1.1. Besides, Aadm [30] is used as an optimizer across all the experiments to achieve optimization of the framework. The learning rate is set as $1e^{-3}$, and the batch size was set to 32. The $g_\theta(\cdot)$ has hidden sizes [128, 128, 256]. To ensure the computational efficiency of *NoSPF*, we implemented DWT and IDWT using the GPU-accelerated pytorch_wavelets[9]. The DWT was performed with Daubechies-4 (DB4) wavelets. The DB4 wavelet was chosen for DWT decomposition due to its effective balance between temporal localization and smoothness. Its orthogonality supports accurate signal reconstruction, while compact support enables detection of abrupt, non-stationary fluctuations of power consumption caused by dynamic workloads. Additionally, its two vanishing moments help separate localized transients from underlying stationary trends without the temporal smearing typical of higher-order wavelets. These properties make DB4 particularly effective for separating the stationary and non-stationary components of server power time series, which is critical for the *NoSPF*. The wavelet decomposition level j is optionally set to $\{2, 3, 4\}$ according to the particular prediction task.

B. Experimental Results

To demonstrate the effectiveness of the method proposed in this paper, we validated its performance from the following aspects.

1) Comparison of Long-Term Power Consumption Forecasting Performance with and Without NoSPF: We first evaluate the predictive performance of four time series forecasting models on the server from the LISA cluster, comparing their accuracy under two conditions: with and without *NoSPF* support. For this evaluation, the prediction horizon $H \in \{120, 240, 360, 480\}$, based on the definition of long-term forecasting in this context. The input window length is fixed at 120 for all experiments, ensuring a consistent context length across all prediction tasks.

[5]https://github.com/ts-kim/RevIN
[6]https://github.com/weifantt/Dish-TS
[7]https://github.com/icantnamemyself/SAN
[8]https://github.com/wayne155/FAN
[9]https://github.com/fbcotter/pytorch_wavelets

The experimental results are summarized in Table I. As indicated, the prediction accuracy of all models generally decreases with increasing prediction horizons. However, in most cases, models that incorporate *NoSPF* generally show better performance compared to those without *NoSPF*, suggesting that *NoSPF* offers some advantages in this context. Specifically, for the DLinear, incorporating *NoSPF* improves MSE by 16.89% and MAE by 11.93%. For SCINet as the backbone, *NoSPF* achieves improvements of 10.86% in MSE and 11.54% in MAE. Similarly, for TiDE, MSE and MAE are enhanced by 11.39% and 9.23%, respectively. Notably, TSMixer with *NoSPF* demonstrates the highest gains, with MSE improved by 19.32% and MAE by 17.54%. These results highlight the ability of the backbone combined with *NoSPF* to address temporal distribution shifts, leading to enhanced forecasting performance. Moreover, they demonstrate the flexibility of *NoSPF* to adapt seamlessly to various types of backbone.

2) Comparison of Power Consumption Forecasting Performance with Different Normalization Methods: In this subsection, we evaluate the performance of the proposed *NoSPF* framework against four normalization methods for non-stationary time series forecasting, using power consumption data from two clusters, LISA and Seren. The results, summarized in the Tables II and III, present the average MSE across all forecasting lengths for different servers in both datasets. The results suggest that *NoSPF* tends to outperform the four baseline methods in most cases, with only minor variations observed in a few specific scenarios.

Specifically, on the LISA server dataset, *NoSPF* achieves average MSE improvements of 9.61%, 4.96%, 18.7%, and 4.47% over RevIN, Dish-TS, FAN, and SAN, respectively. Similarly, on the Seren server dataset, *NoSPF* delivers average MSE gains of 8.15%, 4.30%, 14.22%, and 1.93% over the same baselines. Among the baselines, SAN ranks second overall, benefiting from its localized slice-based normalization strategy, which effectively captures short-term statistical variations. Dish-TS also demonstrates competitive performance, leveraging its adaptive distribution handling to better accommodate input variability. In contrast, RevIN and FAN, which rely primarily on global statistical features, struggle to accurately forecast long-term power consumption in server workloads, resulting in comparatively lower overall performance.

3) Comparison of Power Consumption Forecasting Performance with Different Non-Stationary Models: Additionally, we conducted a comparative evaluation against two models specifically designed for non-stationary time series forecasting: Koopa and the Non-stationary transformer. The average results across all forecasting lengths are presented in Table IV. On average, *NoSPF* with DLinear as the backbone tends to outperform these models across various forecasting tasks, although there are some variations in specific cases. Specifically, on the dataset from LISA, *NoSPF* achieves average MSE and MAE improvements of 14.26% and 3.38% over Koopa, and 14.12% and 4.61% over Non-stationary transformer, respectively. Similarly, on the dataset from Seren, it achieves average MSE and MAE improvements of 16.39% and 1.75% over Koopa, and 17.11% and 0.206% over Non-stationary transformer, respectively.

TABLE I
COMPARISON OF POWER CONSUMPTION TIME-SERIES FORECASTING PERFORMANCE WITH AND WITHOUT *NoSPF*. (THE BOLD VALUES INDICATE THE BEST PERFORMANCE)

Backbones		DLinear		+*NoSPF*		SCINet		+*NoSPF*		TiDE		+*NoSPF*		TSMixer		+*NoSPF*	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
r10n10	120	0.26167	0.31974	**0.23461**	**0.26764**	0.26376	0.31476	**0.23907**	**0.28254**	0.24568	0.30062	**0.23657**	**0.26890**	0.31874	0.36521	**0.24964**	**0.31530**
	240	0.38607	0.38434	**0.33616**	**0.35153**	0.37823	0.42771	**0.34523**	**0.36683**	0.33641	0.37310	**0.33168**	**0.36720**	0.34334	**0.37184**	**0.33993**	0.37717
	360	0.42016	0.44398	**0.39802**	**0.40373**	0.56013	0.53358	**0.41692**	**0.43773**	0.41069	0.41276	**0.39490**	**0.41276**	**0.41887**	0.44990	0.43760	**0.41056**
	480	0.48717	0.44710	**0.43633**	**0.43704**	0.54622	0.54907	**0.44975**	**0.46273**	0.46092	0.49607	**0.43099**	**0.45413**	0.46275	0.48535	**0.44709**	**0.45528**
r10n12	120	0.24825	0.27927	**0.22989**	**0.24046**	0.24138	0.26155	**0.23224**	**0.24042**	0.25655	0.31112	**0.23137**	**0.25745**	0.30231	0.30704	**0.23556**	**0.28824**
	240	0.39517	0.43766	**0.38738**	**0.37649**	0.43955	0.36793	**0.39493**	**0.36085**	**0.38701**	0.41211	0.39240	**0.34860**	0.41505	0.41714	**0.40450**	**0.35720**
	360	0.53675	0.51308	**0.52743**	**0.46824**	0.55381	0.48023	**0.54403**	**0.46731**	**0.52645**	0.50109	0.53754	**0.47624**	0.56680	0.66513	**0.54683**	**0.52163**
	480	0.72245	0.55176	**0.66476**	**0.54534**	0.69475	0.56697	**0.66114**	**0.52773**	**0.65940**	0.60145	0.66225	**0.54227**	0.68897	0.64239	**0.65279**	**0.60652**
r12n13	120	0.16135	0.21416	**0.13345**	**0.17370**	0.15243	0.24165	**0.13410**	**0.17433**	0.14249	0.20009	**0.13404**	**0.16307**	0.14075	0.18138	**0.13907**	**0.17497**
	240	0.20673	0.25396	**0.18173**	**0.23957**	0.19616	**0.23309**	**0.17968**	**0.24476**	0.18338	0.24001	**0.18264**	**0.23919**	0.20621	0.28388	**0.19508**	**0.23552**
	360	0.26670	0.31104	**0.22142**	**0.29077**	0.28515	0.41290	**0.21560**	**0.26080**	0.24385	0.31149	**0.22509**	**0.30510**	0.24003	0.31389	**0.24042**	**0.28867**
	480	0.27437	**0.30571**	**0.25647**	0.32029	0.26378	0.36419	**0.24344**	**0.27550**	0.27112	**0.32415**	**0.25207**	**0.33669**	0.48736	0.56132	**0.25621**	**0.34188**
r12n15	120	0.26399	0.36495	**0.21546**	**0.29341**	0.25083	0.33962	**0.22168**	**0.30704**	0.26341	0.34357	**0.21514**	**0.24679**	0.25786	0.35048	**0.22938**	**0.30483**
	240	0.28388	0.38306	**0.27171**	**0.36237**	0.29012	**0.37044**	**0.27644**	0.37710	0.33734	0.44989	**0.27233**	**0.36640**	0.29873	0.39595	**0.29650**	**0.39464**
	360	0.42694	0.51809	**0.33223**	**0.42399**	0.39653	0.50209	**0.34031**	**0.37042**	0.37586	0.46963	**0.33109**	**0.32146**	0.38931	0.51444	**0.34903**	**0.44622**
	480	0.41493	0.50264	**0.38840**	**0.46372**	0.43433	0.52396	**0.39913**	**0.38327**	0.41670	0.46229	**0.39381**	**0.42120**	0.42250	0.49609	**0.39743**	**0.49149**
r30n5	120	2.14586	0.61793	**1.53121**	**0.43904**	1.66413	0.50708	**1.58143**	**0.46858**	1.89798	0.55274	**1.61977**	**0.48842**	1.83444	0.55700	**1.60546**	**0.48417**
	240	2.54638	0.68067	**1.61318**	**0.46741**	2.03089	0.65887	**1.67164**	**0.49763**	2.11500	0.58320	**1.68847**	**0.50697**	2.51138	0.75522	**1.64500**	**0.49650**
	360	1.76550	0.53490	**1.66132**	**0.47865**	1.80389	0.53299	**1.69891**	**0.50064**	2.37985	0.67882	**1.72150**	**0.51136**	1.70620	0.55817	**1.67165**	**0.48398**
	480	2.17964	0.62827	**1.69388**	**0.48598**	2.00656	0.57969	**1.72179**	**0.50124**	1.94696	0.55699	**1.74146**	**0.51221**	1.80766	0.54032	**1.70703**	**0.49734**
r31n2	120	0.16627	0.25325	**0.15474**	**0.23029**	0.18781	0.28008	**0.17111**	**0.25922**	**0.15937**	0.24497	0.16190	**0.23378**	0.16793	0.26657	**0.16511**	**0.24927**
	240	0.26051	0.33577	**0.24370**	**0.30621**	0.30773	0.38618	**0.26678**	**0.35390**	**0.24428**	0.31286	0.24713	**0.31225**	0.31362	0.43422	**0.26371**	**0.35294**
	360	0.29333	0.35760	**0.28773**	**0.35261**	0.46439	0.53321	**0.33412**	**0.43194**	0.29744	0.35869	**0.29218**	**0.35392**	0.43585	0.49235	**0.33468**	**0.43776**
	480	0.37337	0.40774	**0.31351**	**0.37513**	0.57378	0.58436	**0.36955**	**0.46839**	0.34625	0.38522	**0.31683**	**0.37956**	1.98586	1.08851	**0.38948**	**0.48228**
r31n3	120	0.19098	0.30437	**0.14881**	**0.27377**	**0.16134**	**0.28342**	0.17133	0.29870	**0.15107**	**0.27437**	0.15172	0.28131	0.16361	0.29083	**0.15912**	**0.28203**
	240	0.23512	0.35860	**0.20554**	**0.32952**	**0.21444**	**0.35037**	0.22828	0.35201	**0.20623**	**0.33641**	0.21231	0.34511	0.23022	0.36146	**0.21836**	**0.34293**
	360	0.24907	0.38184	**0.23920**	**0.37029**	**0.25300**	**0.37368**	0.25954	0.39308	0.29487	0.43678	**0.25184**	**0.38727**	0.26974	0.39624	**0.26192**	**0.38487**
	480	0.30200	0.43253	**0.26700**	**0.40080**	**0.24858**	**0.37617**	0.29799	0.42537	**0.26143**	**0.38211**	0.27848	0.41487	0.29262	0.42855	**0.26532**	**0.40095**
r32n5	120	0.14288	0.23062	**0.13318**	**0.20154**	0.14682	0.21629	**0.13376**	**0.19685**	**0.13895**	0.21308	0.14150	**0.19708**	0.14437	0.20172	**0.13914**	**0.20561**
	240	0.23615	0.31536	**0.22366**	**0.25883**	0.21869	**0.26278**	**0.21637**	0.26961	**0.22638**	**0.29874**	0.22829	0.26642	0.23248	0.27480	**0.22822**	**0.25415**
	360	**0.26055**	0.33246	0.26233	**0.30345**	**0.25073**	**0.28523**	0.26360	0.30487	0.27917	0.31321	**0.27130**	**0.29257**	0.26818	0.32254	**0.26280**	**0.31779**
	480	0.27770	0.35588	**0.25163**	**0.30342**	0.27627	0.37294	**0.25792**	**0.30176**	**0.24591**	0.31906	0.25724	**0.29632**	0.25446	0.30710	**0.25424**	**0.30541**

TABLE II
THE AVERAGE MSE PERFORMANCE ACROSS ALL FORECASTING STEPS BY DIFFERENT NORMALIZATION METHODS FOR SERVERS IN THE LISA.(THE BOLD VALUES INDICATE THE BEST PERFORMANCE, AND THE SECOND BEST RESULT IS UNDERLINED)

Backbones	DLinear					SCINet					TiDE					TSMixer				
Methods	RevIN	Dish-TS	FAN	SAN	*NoSPF*	RevIN	Dish-TS	FAN	SAN	*NoSPF*	RevIN	Dish-TS	FAN	SAN	*NoSPF*	RevIN	Dish-TS	FAN	SAN	*NoSPF*
r10n10	0.38704	0.41194	0.41137	0.36601	**0.35128**	0.40485	0.39195	0.41143	0.42401	**0.36274**	0.38881	0.39115	0.40959	0.36008	**0.34853**	0.41855	0.45542	0.41944	0.37059	**0.36856**
r10n12	0.51575	0.46976	0.48591	0.47324	**0.45237**	0.50902	0.49472	0.48834	0.47210	**0.45808**	0.51915	0.47340	0.48867	0.47051	**0.45589**	0.58460	0.55769	0.49151	0.47195	**0.45992**
r12n13	0.19928	0.21215	0.22286	0.21088	**0.19827**	0.22502	0.22001	0.21898	0.22605	**0.19321**	0.19914	0.21374	0.22193	0.20888	**0.19846**	0.22984	0.35450	0.21166	**0.20646**	0.20770
r12n15	0.31385	0.36427	0.33844	0.31141	**0.30195**	0.31979	0.39646	0.33473	0.31849	**0.30939**	0.34556	0.36453	0.33790	0.30188	**0.30309**	0.33100	0.35553	0.36124	**0.30763**	0.31808
r30n5	1.80739	1.65777	1.86729	1.69091	**1.62490**	1.86367	**1.70922**	1.81770	1.71416	1.78267	1.81249	**1.67817**	1.83699	1.68715	1.69280	2.23863	1.70649	1.77937	1.73706	**1.65729**
r31n2	0.28818	0.29172	0.45911	0.26665	**0.24992**	0.30212	0.33065	0.44321	**0.27080**	0.28539	0.28920	0.28483	0.45681	0.26517	**0.25451**	0.31088	0.29501	0.44782	**0.27766**	0.28824
r31n3	0.22785	0.21654	0.26195	0.23144	**0.21514**	0.24327	0.23806	0.26316	**0.23429**	0.23928	0.22665	**0.22159**	0.26929	0.22885	0.22359	0.61766	0.24043	0.25573	0.22712	**0.22618**
r32n5	0.24069	0.22980	0.24475	0.22414	**0.21770**	0.24486	0.24257	0.24339	0.22804	**0.22780**	0.24033	0.24068	0.24636	**0.22395**	0.22458	0.27029	0.24937	0.23978	0.22376	**0.22110**

4) Comparison of Cross-Server Power Consumption Time-Series Forecasting Performance: Due to the substantial variability in resource demands during server operations, cross-server power prediction presents additional challenges related to temporal distribution shifts. To assess the effectiveness of the proposed method in such contexts, we designed a series of cross-server prediction tasks using the LISA cluster dataset. In these tasks, the model is trained exclusively on the data from one server (source server) and subsequently evaluated on the data from a different server (target server) without any retraining. Specifically, we selected six representative servers from the LISA cluster, resulting in 24 cross-server prediction tasks. For each task, the model trained on the source server is directly tested on the target server data, providing a rigorous evaluation of the model's ability to generalize across servers.

The cross-server power consumption forecasting performance of two baselines, with and without *NoSPF*, is reported in Table V. Empirical results show that integrating *NoSPF* consistently improves performance across all tasks. In general, with the help of *NoSPF*, cross-server power prediction achieves average improvements of 16.53% in MSE and 19.94% in MAE for DLinear, and 9.31% in MSE and 14.18% in MAE for SCINet, compared to direct prediction. These results demonstrate that *NoSPF* effectively mitigates temporal distribution shifts and enhances the generalization ability of forecasting models in cross-server scenarios.

5) Analysis of NoSPF: In this section, we examine the effects of varying configurations within the proposed *NoSPF* framework on the server in LISA. First, we investigate the impact of *NoSPF* with the number of wavelet transform decomposition level j. The decomposition levels were varied across $j \in \{2, 3, 4, 5, 6\}$ using the DLinear, and experiments were conducted on power consumption data from the servers r10n10 and r31n2. The results, depicted in Fig. 4, reveal that the model's forecasting performance remains relatively stable as the decomposition level increases. However, when $j > 5$,

TABLE III
THE AVERAGE MSE PERFORMANCE ACROSS ALL FORECASTING STEPS BY DIFFERENT NORMALIZATION METHODS FOR SERVERS IN THE SEREN

Backbones	DLinear					SCINet					TiDE					TSMixer				
Methods	RevIN	Dish-TS	FAN	SAN	*NoSPF*	RevIN	Dish-TS	FAN	SAN	*NoSPF*	RevIN	Dish-TS	FAN	SAN	*NoSPF*	RevIN	Dish-TS	FAN	SAN	*NoSPF*
Node-147	0.95237	0.91966	1.08634	0.90357	**0.88482**	0.95890	0.90730	1.16810	0.90826	**0.90406**	0.95497	0.90138	1.09878	0.90655	**0.88239**	1.09645	0.90285	1.08105	0.90696	**0.89212**
Node-166	0.43305	0.42717	0.43774	0.41793	**0.41606**	0.43054	0.42217	0.45255	0.41968	**0.40701**	0.43062	0.42459	0.44438	0.42493	**0.41193**	0.44230	0.45957	0.44512	0.43080	**0.42715**
Node-246	0.64707	0.60187	**0.59550**	0.61067	0.59712	0.62584	0.62262	0.65994	0.62400	**0.59349**	0.64420	0.60703	0.62126	0.61124	**0.59834**	0.61914	0.62279	0.59957	0.62268	**0.59905**
Node-256	0.91675	0.83068	0.86189	0.84556	**0.80669**	0.92827	0.83016	0.93943	0.84862	**0.80067**	0.91227	0.82946	0.89017	0.84281	**0.80458**	0.92287	0.85962	0.86879	**0.85205**	0.85283
Node-78	**0.14975**	0.20544	0.21289	0.20197	0.20005	**0.18227**	0.21886	0.22860	0.19540	0.21569	0.58390	0.55808	0.63485	0.55535	**0.54687**	0.60753	0.58303	0.62494	**0.56632**	0.58409
Node-90	0.58328	0.54622	0.60938	0.55353	**0.54350**	0.59032	0.56079	0.67569	0.55643	**0.55136**	0.22572	0.21440	0.21767	**0.19882**	0.20029	0.21915	0.22604	0.21380	0.22059	**0.21351**
Node-103	0.80384	0.75261	0.74931	**0.73060**	0.73564	0.80602	0.76196	0.83021	**0.72738**	0.73337	0.79955	0.74803	0.76676	0.73052	**0.71186**	0.76179	0.76092	0.74659	0.72807	**0.72740**
Node-119	0.89913	0.80011	0.81282	0.80655	**0.78567**	0.84641	0.80439	0.89732	0.79528	**0.76374**	0.89869	0.81968	0.84048	0.80060	**0.78589**	0.85652	0.79537	0.82490	0.79943	**0.78142**

TABLE IV
PERFORMANCE COMPARISON ACROSS ALL FORECASTING STEPS FOR SERVERS IN LISA AND SEREN WITH DIFFERENT NON-STATIONARY MODELS

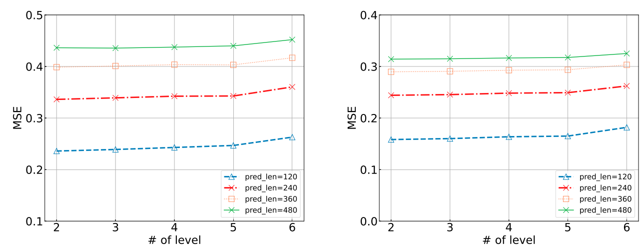
Dataset	Backbones	Koopa		Non-stationary transformer		DLinear + *NoSPF*	
	Metrics	MSE	MAE	MSE	MAE	MSE	MAE
LISA	r10n10	0.44486	0.40109	0.39286	0.37271	**0.34994**	**0.36727**
	r10n12	0.57115	0.41385	0.50630	0.40382	**0.45339**	**0.40928**
	r12n13	0.25870	**0.23800**	0.26259	0.27390	**0.19951**	0.26052
	r12n15	0.33646	0.37763	0.35679	0.39018	**0.30267**	**0.37655**
	r30n5	1.75937	0.50599	1.84215	0.47109	**1.62490**	**0.46777**
	r31n2	0.33662	0.37149	0.32109	0.36631	**0.24992**	**0.31606**
	r31n3	0.24339	**0.33804**	0.28342	0.39537	**0.21514**	0.34359
	r32n5	0.26355	**0.26013**	0.24188	0.27027	**0.21770**	0.26681
Seren	Node-147	1.07682	0.80097	1.02669	0.78528	**0.88482**	**0.76106**
	Node-166	0.50640	0.51368	0.47143	**0.46481**	**0.41606**	0.51469
	Node-246	0.68077	0.65188	0.65644	**0.60764**	**0.59712**	0.64709
	Node-256	0.94898	0.78927	0.93614	0.72201	**0.80669**	**0.72077**
	Node-78	**0.17828**	**0.22293**	0.29592	0.30269	0.20005	0.33590
	Node-90	0.68675	0.52841	0.65387	0.52513	**0.54350**	**0.52485**
	Node-103	0.96850	0.79132	1.01345	0.78842	**0.73564**	**0.72810**
	Node-119	0.89750	0.76041	0.94108	0.78471	**0.78567**	**0.73800**

TABLE V
COMPARISON OF CROSS-SERVER POWER CONSUMPTION TIME-SERIES FORECASTING PERFORMANCE

Backbones		DLinear		DLinear + *NoSPF*		SCINet		SCINet + *NoSPF*	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
r10n10	r10n12	0.26357	0.29759	**0.22852**	**0.22986**	0.28789	0.29905	**0.24386**	**0.23009**
	r12n13	0.17160	0.26671	**0.13564**	**0.18831**	0.15559	0.22348	**0.13411**	**0.19241**
	r12n15	0.24977	0.34201	**0.21944**	**0.28806**	0.24115	0.30560	**0.22913**	**0.29282**
r10n12	r10n10	0.27673	0.32509	**0.23566**	**0.27372**	0.25319	0.31948	**0.23997**	**0.29193**
	r12n13	0.18017	0.26350	**0.13714**	**0.19868**	0.18129	0.27822	**0.13937**	**0.19579**
	r12n15	0.30108	0.36472	**0.22277**	**0.29506**	0.24952	0.33690	**0.22671**	**0.29935**
r12n13	r10n10	0.27618	0.30127	**0.24049**	**0.26184**	0.26344	0.34388	**0.24632**	**0.29088**
	r10n12	0.25932	0.25043	**0.23644**	**0.21719**	0.26432	0.30873	**0.25491**	**0.26231**
	r12n15	0.25926	0.31934	**0.21363**	**0.27322**	0.24615	0.34144	**0.21906**	**0.28913**
r12n15	r10n10	0.26609	0.33566	**0.23365**	**0.27488**	0.24508	0.31104	**0.23424**	**0.29568**
	r10n12	0.25584	0.30978	**0.22819**	**0.24188**	0.23605	0.26196	**0.23145**	**0.23830**
	r12n13	0.19745	0.30702	**0.13798**	**0.20601**	0.17177	0.26393	**0.13611**	**0.20553**
r30n5	r31n2	0.43118	0.43163	**0.22950**	**0.32029**	0.32953	0.37713	**0.20075**	**0.30247**
	r31n3	0.60249	0.60331	**0.24385**	**0.40294**	0.42224	0.51809	**0.24238**	**0.40313**
	r32n5	0.40809	0.49018	**0.20280**	**0.32504**	0.32797	0.41910	**0.19826**	**0.32007**
r31n2	r30n5	1.91599	0.50093	**1.85589**	**0.47289**	2.26667	0.56253	**1.82432**	**0.52190**
	r31n3	0.16940	0.31241	**0.15408**	**0.29662**	0.16896	0.28750	**0.15494**	**0.28366**
	r32n5	0.13756	0.22936	**0.13147**	**0.21735**	0.14791	0.21310	**0.14053**	**0.21859**
r31n3	r30n5	2.16966	0.57025	**1.94810**	**0.47074**	2.37609	0.53430	**1.82283**	**0.48149**
	r31n2	0.20027	0.25937	**0.15725**	**0.22423**	0.16756	0.24936	**0.16112**	**0.23800**
	r32n5	0.17621	0.23807	**0.13166**	**0.19420**	0.13509	0.20692	**0.13394**	**0.20617**
r32n5	r30n5	2.15155	0.55615	**1.92902**	**0.49053**	2.20447	0.54947	**2.04111**	**0.51051**
	r31n2	0.16799	0.25238	**0.15960**	**0.22851**	0.17732	0.25628	**0.16998**	**0.25586**
	r31n3	0.17881	0.31675	**0.14826**	**0.27479**	0.15451	0.28125	**0.13999**	**0.26521**

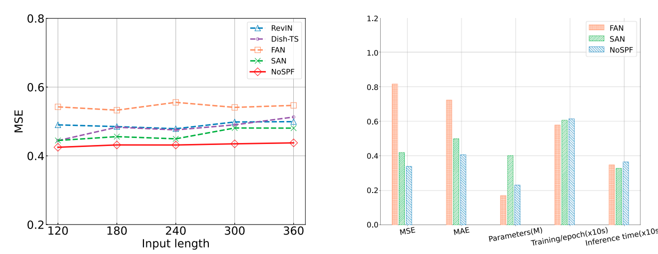
the forecasting error starts to increase, indicating that excessive levels of wavelet decomposition may lead to over-processing of the input time-series data. This over-processing could result in the loss of some important long-term features, ultimately negatively impacting the forecasting accuracy.

Furthermore, we analyzed the performance of *NoSPF* under varying input sequence lengths L using the r10n10 server dataset. The L were set to $\{120, 180, 240, 300, 360\}$, with the forecasting horizon H fixed at 480. The experimental results, presented in Fig. 5(a), demonstrate that the forecasting performance of *NoSPF* remains consistent and does not exhibit significant improvement or degradation as the input length changes. Additionally, *NoSPF* consistently outperforms four baseline normalization methods across all tested input lengths. This robustness highlights the effectiveness of *NoSPF* in predicting power consumption for non-stationary time series under varying input configurations.



(a) MSE with different level for r10n10. (b) MSE with different level for r31n2.

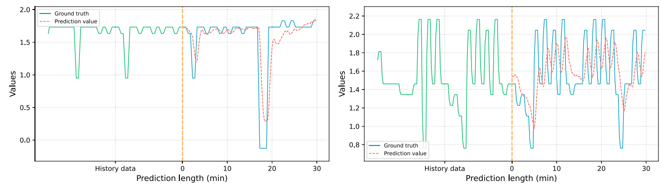
Fig. 4. Comparison of forecasting performance with different level for *NoSPF*.



(a) MSE performace with different input length. (b) Efficiency comparsion of *NoSPF*.

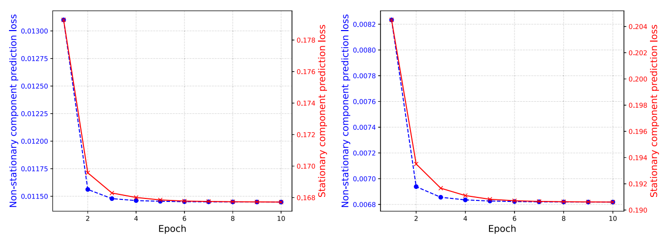
Fig. 5. Analysis of *NoSPF*.

To evaluate the computational complexity of the proposed *NoSPF*, we conducted both temporal and spatial analyses of a power forecasting task, using DLinear as the backbone model and data from the r31n2. The input sequence length L was set to 360, and the output sequence length H was set to 720. The results in Fig. 5(b) demonstrate that NoSPF significantly outperforms methods like SAN and FAN, achieving 19.12% lower MSE and 18.16% lower MAE, while maintaining a competitive computational footprint. This performance advantage is primarily attributed to the DWT used in *NoSPF* for decomposing power consumption time series, which effectively isolates non-stationary components. Although DWT introduces additional computational overhead due to its hierarchical convolution and



(a) r12n13 (CPU server) (b) r31n3 (GPU server)

Fig. 6. Showcase of power consumption prediction of *NoSPF*.



(a) r10n10 (b) r10n12

Fig. 7. Training loss curves for the stationary component (red) and non-stationary component (blue) predictions during the *NoSPF* training process.

downsampling structure, with a time complexity of $O(NlogN)$, where N is the length of the input time series, this cost is justified by the substantial gains in prediction accuracy. Moreover, despite the added processing steps, *NoSPF* maintains competitive inference times, comparable to methods like FAN and SAN. This efficient runtime, combined with the significant forecasting accuracy improvements, highlights the practical advantages of *NoSPF* for real-world deployment, where both precision and computational efficiency are critical.

In addition, we present prediction showcase on two types of servers within the LISA cluster, highlighting the alignment between *NoSPF* and the ground-truth, as shown in Fig. 6. The figure demonstrates that the predictions generated by *NoSPF* closely to the actual power consumption and effectively capture its short-term fluctuations. Notably, the divergence between forecast and observed values did not accumulate significantly over the forecast horizon, which underscores the stability of the *NoSPF*. These qualitative results serve as an intuitive complement to extensive quantitative experiments, further validating the robustness of *NoSPF* in forecasting non-stationary power consumption.

C. Analysis of Error Sources of NoSPF

To provide a more granular analysis of the primary error sources within the *NoSPF*, we examined the training losses associated with the prediction of its constituent stationary and non-stationary components. For this analysis, the DLinear was employed as the backbone for stationary component forecasting. Fig. 7 illustrates the convergence of the training losses for both the stationary and non-stationary component predictions on the r10n10 and r10n12.

A key observation from Fig. 7 is the significant disparity in the converged training loss magnitudes between the two component predictors. Specifically, the prediction loss for the stationary component, modeled by the DLinear backbone, is substantially higher than that of the non-stationary component, which is predicted by a lightweight MLP following DWT decomposition. This disparity suggests that *NoSPF*'s strategy for handling non-stationarity, namely, the dedicated MLP applied to DWT detail coefficients, is highly effective in capturing and minimizing errors associated with short-term, high-frequency variations. Conversely, the considerably larger prediction loss for the stationary component, which represents longer-term power consumption trends, indicates that this component is the predominant contributor to the overall prediction error within the current *NoSPF* configuration.

Moreover, according to the properties of the DWT, when applying DWT-based time series decomposition, higher decomposition levels j generate more high-frequency detail coefficients, thereby increasing the complexity and variability of the non-stationary components fed into g_θ. This, in principle, raises the difficulty of forecasting the non-stationary component. If the final forecast accuracy were highly sensitive to errors in the non-stationary prediction, we would expect to observe a marked deterioration in performance as decomposition depth increases. However, as shown in Fig. 4, increasing the number of j does not cause a significant degradation in overall forecasting accuracy. This result indicates that although the two optimization levels are coupled, their dependency is relatively moderate: errors in non-stationary prediction are effectively contained and do not substantially propagate to the final forecast. The non-stationary predictor primarily serves to capture transient fluctuations and mitigate high-frequency variations, thereby stabilizing the overall forecasting process. In this sense, it acts as a corrective component that enhances robustness, while the stationary predictor remains the principal determinant of long-term forecasting accuracy.

In summary, while *NoSPF* demonstrates a clear advantage in isolating and accurately modeling the challenging non-stationary components, the relatively higher error associated with stationary component prediction highlights a potential area for further model enhancement. Future improvements could potentially be achieved by exploring more sophisticated backbone architectures or refined strategies for forecasting these underlying stationary patterns.

V. Related Work

A. Time-Series Power Consumption Forecasting Models

Accurate server power forecasting is vital for energy-aware optimization and long-term planning in data centers. Extensive research has explored this domain, as summarized in surveys such as [31], [32]. Early works primarily utilized statistical and regression-based models, but recent advances have focused on learning-based time-series models to capture temporal dependencies more effectively.

For example, Wu et al. [33] developed a power prediction model based on the Elman neural network, addressing the limitations of traditional regression models with fixed forms and limited generalization. Lin et al. [34] introduced an LSTM-based model for server power forecasting, incorporating fine-grained CPU, memory, and disk performance analysis under varying task loads. Similarly, Yi et al. [35] trained an LSTM model using server performance counter and task arrival trajectories to enhance power prediction accuracy. Motaki et al. [36] proposed a power forecasting model that treats workload and power consumption as random variables, leveraging non-parametric methods to capture their statistical dependencies. Zheng et al. [37] improved GRU-based power forecasting by incorporating selective state updates and adaptive gradient optimization, effectively mitigating long-term memory decay. Shen et al. [38] presented a two-stage attention LSTM model that transforms time-series data into tensors for capturing complex temporal patterns, combining tensor decomposition, LSTM, and attention mechanisms to predict non-linear power consumption. Zhou et al. [39] proposed a BiLSTM-based model optimized with an improved resonance optimization algorithm (SLCOA) for data center energy forecasting. Jing et al. [40] introduced a hybrid CNN-BiLSTM model with attention mechanisms for power forecasting in heterogeneous computing servers, demonstrating improved prediction performance. Long et al. [41] proposed Li-MSA, a few-shot learning approach for server time-series power consumption forecasting. The method integrates a linear interpolation module for data augmentation and smoothing with a multi-head sparse temporal pattern attention mechanism, thereby enhancing the generalization capability of server power consumption forecasts on small-scale datasets.

Although these models achieve promising results in specific server power forecasting scenarios, they commonly rely on the assumption of stable and periodic consumption patterns. However, they lack explicit mechanisms to address abrupt fluctuations induced by dynamic workload variations during server operation. Consequently, their long-term forecasting performance may degrade in the presence of temporal distribution shifts, limiting their robustness and generalization in real-world data center environments.

B. Non-Stationary Time-Series Forecasting Methods

To address the challenges posed by temporal distribution shifts in time-series forecasting models, statistical feature normalization methods have gained significant attention due to their simplicity and effectiveness. Among these, Passalis et al. [22] introduced the deep adaptive input normalization (DAIN) method, which adaptively normalizes input data based on the distribution of the time-series measurements, enhancing the model's ability to handle non-stationary data. Kim et al. [19] proposed the reversible instance normalization method (RevIN), which uses learnable affine transformations for symmetric normalization and denormalization of time-series data, significantly improving long-term forecasting performance. Du et al. [18] approached temporal changes in time-series statistical features from a distribution perspective, introducing an adaptive RNN model that reduces distribution mismatch through time-series distribution matching, thereby enhancing predictive accuracy under temporal distribution shifts. Fan et al. [20] developed Dish-TS, which maps the input sequence to learnable distribution coefficients, effectively addressing distributional shifts between input and output spaces in time-series forecasting. Ye et al. [27] proposed the frequency adaptive normalization (FAN) method, which utilizes Fourier transforms to capture dynamic trends and seasonal patterns, modeling frequency differences between input and output sequences to improve forecasting accuracy. Liu et al. [21] introduced a slice-level adaptive normalization method that handles non-stationarity in local time slices and independently models the evolution of time-series statistical features, offering flexibility in managing distribution differences between input and forecast sequences. In addition to these instance normalization methods, Liu et al. [28] introduced the Non-stationary Transformer, which incorporates series stabilization and destabilization attention modules to effectively remove and recover non-stationary information. Liu et al. [29] proposed a time-series forecasting method based on Koopman theory, which separates time-varying and time-invariant components, using Fourier filters and Koopman operators to capture underlying dynamics while addressing the locality of time-varying dynamics through context-aware operators.

Although the aforementioned methods were not originally designed for server power consumption forecasting, the core issue they address, namely temporal distribution shifts, is highly relevant to the abrupt fluctuations commonly observed in data center server power consumption. Analyzing the principles, strengths, and limitations of these methods helps to understand the challenges that may arise when applying them directly to power consumption prediction. While reversible instance normalization methods have been shown to be effective in mitigating temporal distribution shifts, they often rely on global statistical features of the input time series. This reliance limits their ability to handle shifts caused by abrupt, localized changes in statistical properties. Such sudden variations frequently occur in server power consumption patterns in real-world data centers, posing a significant challenge to the applicability of these methods in such scenarios. These observations naturally motivate the design of *NoSPF*, which explicitly models non-stationary components to improve robustness under real-world server power scenarios.

VI. Conclusion and Future Work

Accurate long-term forecasting of server power consumption in data centers is essential for improving energy efficiency. However, the non-stationary nature of power consumption, driven by dynamic resource demands, complicates forecasting tasks. Existing methods that rely on global statistical normalization struggle with localized variations, reducing forecasting accuracy. This paper proposes a framework to implement non-stationary long-term power consumption forecasting, named *NoSPF*. In general, by separating local non-stationary components and simplifying the learning of long-term patterns, our framework enhances the forecasting of future power consumption. A lightweight but effective MLP module is employed to predict non-stationary components, thus improving the ability of *NoSPF* to approximate the real power data distribution by integrating the predicted non-stationary components with the stationary forecasting. Experimental results on real-world data show that our method outperforms existing approaches in accuracy, offering a practical solution for energy management in real-world cloud data centers.

In future work, we plan to develop a long-term energy consumption optimization strategy for cloud data center servers, leveraging accurate power consumption time-series forecasts from *NoSPF*. This will enable more efficient dynamic energy management and energy-saving optimization across multiple time scales in cloud data center operations.

REFERENCES

[1] J. Wang et al., "Designing cloud servers for lower carbon," in *Proc. ACM/IEEE 51st Annu. Int. Symp. Comput. Archit. (ISCA),* Piscataway, NJ, USA: IEEE Press, 2024, pp. 452–470.

[2] Google Cloud, "Carbon free energy for google cloud regions," 2024. Accessed: Nov. 4, 2025. [Online]. Available: https://cloud.google.com/sustainability/region-carbon

[3] Brad Smith, "Microsoft will be carbon negative by 2030," 2020. Accessed: Nov. 4, 2025. [Online]. Available: https://blogs.microsoft.com/blog/2020/01/16/microsoft-will-be-carbon-negative-by-2030/

[4] M. Kodnongbua et al., "Dense server design for immersion cooling," *ACM Trans. Graph. (TOG*, vol. 43, no. 6, pp. 1–20, 2024.

[5] M. Jalili et al., "Cost-efficient overclocking in immersion-cooled datacenters," in *Proc. ACM/IEEE 48th Annu. Int. Symp. Computer Archit. (ISCA),* Piscataway, NJ, USA: IEEE Press, 2021, pp. 623–636.

[6] Q. Pei et al., "CoolEdge: Hotspot-relievable warm water cooling for energy-efficient edge datacenters," in *Proc. 27th ACM Int. Conf. Archit. Support Program. Lang. Operat. Syst.*, 2022, pp. 814–829.

[7] D. Zhao, J. Zhou, J. Zhai, and K. Li, "A reinforcement learning based framework for holistic energy optimization of sustainable cloud data centers," *IEEE Trans. Services Comput.*, vol. 18, no. 1, pp. 15–28, Jan./Feb. 2025.

[8] A. Jahanshahi, N. Yu, and D. Wong, "PowerMorph: QoS-aware server power reshaping for data center regulation service," *ACM Trans. Archit. Code Optim.*, vol. 19, no. 3, pp. 1–27, 2022.

[9] T. Sukprasert, A. Souza, N. Bashir, D. Irwin, and P. Shenoy, "On the limitations of carbon-aware temporal and spatial workload shifting in the cloud," in *Proc. 19th Eur. Conf. Comput. Syst.*, 2024, pp. 924–941.

[10] A. Jayanetti, S. Halgamuge, and R. Buyya, "Multi-agent deep reinforcement learning framework for renewable energy-aware workflow scheduling on distributed cloud data centers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 35, no. 4, pp. 604–615, Apr. 2024.

[11] W. Fan, X. Liu, H. Yuan, N. Li, and Y. Liu, "Time-slotted task offloading and resource allocation for cloud-edge-end cooperative computing networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 8, pp. 8225–8241, Aug. 2024.

[12] S. Tang et al., "A survey on scheduling techniques in computing and network convergence," *IEEE Commun. Surveys & Tut.*, vol. 26, no. 1, pp. 160–195, 1st Quart. 2024.

[13] G. Zhou, W. Tian, R. Buyya, R. Xue, and L. Song, "Deep reinforcement learning-based methods for resource scheduling in cloud computing: A review and future directions," *Artif. Intell. Rev.*, vol. 57, no. 5, p. 124, 2024.

[14] C.-H. Hsu, Q. Deng, J. Mars, and L. Tang, "SmoothOperator: Reducing power fragmentation and improving power utilization in large-scale datacenters," in *Proc. 23rd Int. Conf. Archit. Support Program. Lang. Operat. Syst.*, 2018, pp. 535–548.

[15] P. Patel et al., "Characterizing power management opportunities for LLMs in the cloud," in *Proc. 29th ACM Int. Conf. Archit. Support Program. Lang. Operat. Syst.*, vol. 3, 2024, pp. 207–222.

[16] R. Bianchini, C. Belady, and A. Sivasubramaniam, "Datacenter power and energy management: past, present, and future," *IEEE Micro*, vol. 44, no. 5, pp. 30–36, Sep./Oct. 2024.

[17] W. Lin, F. Shi, W. Wu, K. Li, G. Wu, and A.-A. Mohammed, "A taxonomy and survey of power models and power modeling for cloud servers," *ACM Comput. Surveys (CSUR)*, vol. 53, no. 5, pp. 1–41, 2020.

[18] Y. Du et al., "AdarNN: Adaptive learning and forecasting of time series," in *Proc. 30th ACM Int. Conf. Inform. Knowl. Manage.*, 2021, pp. 402–411.

[19] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *Proc. Int. Conf. Learn. Representations*, 2021.

[20] W. Fan, P. Wang, D. Wang, D. Wang, Y. Zhou, and Y. Fu, "Dish-TS: A general paradigm for alleviating distribution shift in time series forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 7522–7529.

[21] Z. Liu et al., "Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective," in *Proc. 38th Annu. Conf. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 14273–14292.

[22] N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Deep adaptive input normalization for time series forecasting," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3760–3765, Sep. 2020.

[23] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proc. AAAI Conf. Artif. Intell.,* 2023, pp. 11121–11128.

[24] M. Liu et al., "SciNet: Time series modeling and forecasting with sample convolution and interaction," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 5816–5828.

[25] A. Das, W. Kong, A. Leach, S. K. Mathur, R. Sen, and R. Yu, "Long-term forecasting with tide: Time-series dense encoder," *Trans. Mach. Learn. Res.*.

[26] S.-A. Chen, C.-L. Li, S. O. Arik, N. C. Yoder, and T. Pfister, "TSMixer: An all-MLP architecture for time series forecasting," *Trans. Mach. Learn. Res.*, Sep. 2023.

[27] W. Ye, S. Deng, Q. Zou, and N. Gui, "Frequency adaptive normalization for non-stationary time series forecasting," in *Proc. 38th Annu. Conf. Neural Inf. Process. Syst.*, 2024, pp. 31350–31379.

[28] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 9881–9893.

[29] Y. Liu, C. Li, J. Wang, and M. Long, "Koopa: Learning non-stationary time series dynamics with Koopman predictors," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2023, pp. 12271–12290.

[30] D. P. Kingma, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[31] C. Jin, X. Bai, C. Yang, W. Mao, and X. Xu, "A review of power consumption models of servers in data centers," *Appl. Energy*, vol. 265, 2020, Art. no. 114806.

[32] L. Ismail and H. Materwala, "Computing server power modeling in a data center: Survey, taxonomy, and performance evaluation," *ACM Comput. Surveys (CSUR)*, vol. 53, no. 3, pp. 1–34, 2020.

[33] W. Wu, W. Lin, L. He, G. Wu, and C.-H. Hsu, "A power consumption model for cloud servers based on Elman neural network," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1268–1277, Oct./Dec. 2019.

[34] W. Lin, G. Wu, X. Wang, and K. Li, "An artificial neural network approach to power consumption model construction for servers in cloud data centers," *IEEE Trans. Sustain. Comput.*, vol. 5, no. 3, pp. 329–340, Sep. 2020.

[35] D. Yi, X. Zhou, Y. Wen, and R. Tan, "Efficient compute-intensive job allocation in data centers via deep reinforcement learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1474–1485, Jun. 2020.

[36] S. El Motaki, A. Yahyaouy, and H. Gualous, "Modeling the correlation between the workload and the power consumed by a server using stochastic and non-parametric approaches," *Softw.: Pract. Experience*, vol. 52, no. 10, pp. 2177–2190, 2022.

[37] W. Zheng and G. Chen, "An accurate GRU-based power time-series prediction approach with selective state updating and stochastic optimization," *IEEE Trans. Cybern.*, vol. 52, no. 12, pp. 13902–13914, Dec. 2021.

[38] Z. Shen, B. Liu, Q. Zhou, Z. Liu, B. Xia, and Y. Li, "Cost-sensitive tensor-based dual-stage attention LSTM with feature selection for data center server power forecasting," *ACM Trans. Intell. Syst. Technol.*, vol. 14, no. 2, pp. 1–20, 2023.

[39] J. Zhou, Y. Wang, and J. Lil, "Data center energy consumption prediction model based on deep neural network BiLSTM," in *Proc. IEEE 48th Annu. Comput., Softw., Appl. Conf. (COMPSAC),* Piscataway, NJ, USA: IEEE Press, 2024, pp. 737–745.

[40] C. Jing and J. Li, "CBLA_PM: An improved ANN-based power consumption prediction algorithm for multi-type jobs on heterogeneous computing server," *Cluster Comput.*, vol. 27, no. 1, pp. 377–394, 2024.

[41] S. Long, Y. Li, Z. Li, G. Xie, W. Lin, and K. Li, "Li-MSA: Power consumption prediction of servers based on few-shot learning," *IEEE Trans. Services Comput.*, vol. 18, no. 2, pp. 926–939, Mar./Apr. 2025.

Ruichao Mo received the bachelor's and master's degrees from the School of Computer and Software, Nanjing University of Information Science and Technology, in 2019 and 2022, respectively. He is currently working toward the Ph.D. degree with the School of Computer and Engineering, South China University of Technology. His research interests include cloud computing, service computing, and machine learning.

Weiwei Lin (Senior Member, IEEE) received the B.S. and M.S. degrees from Nanchang University, in 2001 and 2004, respectively, and the Ph.D. degree in computer application from the South China University of Technology, in 2007. He has been a Visiting Scholar with Clemson University from 2016 to 2017. He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. His research interests include distributed systems, cloud computing, and AI application technologies. He has published more than 200 papers in refereed journals and conference proceedings. He has been a reviewer for many international journals, including ICML, NeurIPS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON CLOUD COMPUTING, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON MOBILE COMPUTING, etc. He is a Distinguished Member of CCF.

Shengsheng Lin received the bachelor's degree from the South China University of Technology, in 2022. He is currently working toward the Ph.D. degree in computer technology with the School of Computer Science and Engineering, South China University of Technology, Guangdong, China. His research interests include machine learning and time series forecasting. He has published over five research papers as the first author and serves as a Reviewer for high-impact journals and conferences such as IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, KDD, and NeurIPS.

Simon Fong received the B.Eng. degree (Hons.) in computer systems and the Ph.D. degree (Hons.) in computer science from La Trobe University, Australia, in 1993 and 1998, respectively. He is currently working as an Associate Professor with the Computer and Information Science Department, University of Macau. He is a Co-Founder of the Data Analytics and Collaborative Computing Research Group, Faculty of Science and Technology. He has published over 500 international conference and peer-reviewed journal papers, mostly in the areas of data mining, data stream mining, big data analytics, meta-heuristics optimization algorithms, and AI medical applications.

Keqin Li (Fellow, IEEE) received the B.S. degree from Tsinghua University, in 1985, and the Ph.D. degree from the University of Houston, in 1990, both in computer science. He is a SUNY Distinguished Professor with the State University of New York and a National Distinguished Professor with Hunan University, China. He has authored or co-authored more than 1200 journal articles, book chapters, and refereed conference papers. He holds nearly 80 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top few most influential scientists in parallel and distributed computing, regarding single-year impact (ranked #2) and career-long impact (ranked #3) based on a composite indicator of the Scopus citation database. He is listed in Scilit Top Cited Scholars (2023–2025) and is among the top 0.02% out of over 20 million scholars worldwide based on top-cited publications in the last ten years. He is listed in ScholarGPS Highly Ranked Scholars (2022–2024) and is among the top 0.002% out of over 30 million scholars worldwide based on a composite score of three ranking metrics for research productivity, impact, and quality in the recent five years. He received the IEEE TCCLD Research Impact Award from the IEEE CS Technical Committee on Cloud Computing in 2022 and the IEEE TCSVC Research Innovation Award from the IEEE CS Technical Community on Services Computing in 2023. He won the IEEE Region 1 Technological Innovation Award (Academic) in 2023. He was a recipient of the 2022–2023 International Science and Technology Cooperation Award and the 2023 Xiaoxiang Friendship Award of Hunan Province, China. He is a Member of the SUNY Distinguished Academy. He is an AAAS Fellow, an AAIA Fellow, an ACIS Fellow, and an AIIA Fellow. He is a member of the European Academy of Sciences and Arts. He is a member of Academia Europaea (Academician of the Academy of Europe).