sustainability

MDPI

Article

Load Forecasting for Sustainable Demand-Side Energy Management Under User Heterogeneity and Temporal Dynamics

Haoheng Qin [1,*], Liang Xu [1], Garrett Ge Gao [1], Yang Helen Ouyang [2], Tianyu Xiao [3] and Yunhan Xia [4]

[1] School of Business Administration, Southwestern University of Finance and Economics, Chengdu 611130, China; arecxuliang1112@gmail.com (L.X.); gaogegarrett@gmail.com (G.G.G.)
[2] Sichuan Energy Investment Wind Power Development Co., Ltd., Chengdu 610041, China; oyy19@tsinghua.org.cn
[3] SWUFE-UD Institute of Data Science at SWUFE, Southwestern University of Finance and Economics, Chengdu 611130, China; 42453058@smail.swufe.edu.cn
[4] School of Management Science and Engineering, Southwestern University of Finance and Economics, Chengdu 611130, China; 42427161@smail.swufe.edu.cn
* Correspondence: 1231201z5015@smail.swufe.edu.cn

Abstract

The increasing integration of renewable energy is a key pathway toward sustainable energy transitions, but it also introduces supply-side uncertainty that makes demand flexibility management increasingly important. In this context, accurate load forecasting becomes a key task for balancing uncertain supply with demand-side behavior and supporting more reliable energy management. Most existing studies approach load forecasting by developing advanced forecasting models. However, in real-world commercial-user forecasting, load data often exhibit inter-user heterogeneity and temporal dynamics, making forecasting performance depend not only on model architecture, but also on how models are trained, shared, combined, and used; we refer to these design choices as forecasting strategies. Therefore, we study day-ahead individual load forecasting from a strategy-level perspective using historical load observations only. We compare homogeneous strategies, inter-user-heterogeneity-aware strategies, and temporal-dynamics-aware strategies using three real-world electricity consumption datasets from China. The results show that forecasting strategy design substantially affects prediction performance, and that both inter-user heterogeneity and temporal dynamics provide useful and complementary information for improving day-ahead individual load forecasting. We find that appropriate strategy design can improve MSE by more than 6% compared with using only a single model. Overall, the study provides useful methodological tools and practical insights for load forecasting and sustainable energy management.

Keywords: load forecasting; demand-side energy management; user heterogeneity; temporal dynamics; forecasting strategies

Check for updates

Academic Editor: Seung-Hoon Yoo

Received: 10 June 2026
Revised: 4 August 2026
Accepted: 5 August 2026
Published: 10 August 2026

Copyright: © 2026 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.

1. Introduction

As power systems transition toward low-carbon and sustainable operation with increasing renewable energy integration, accurate load forecasting has become essential for managing supply–demand balance under greater generation variability and uncertainty. From a broader sustainability perspective, the growing penetration of renewable energy also increases the importance of coordinated electrical-system design and adaptive operational capabilities for supporting reliable and sustainable energy systems [1]. Reliable

load forecasts support decisions such as generation scheduling, system operation, resource planning, demand response, peak-load management, and electricity cost control [2–4].

In practice, accurate load forecasting supports sustainable energy management by reducing uncertainty in both supply-side and demand-side decisions. On the supply side, improved forecasting accuracy enables more efficient unit commitment and economic dispatch, thereby lowering power-system operating costs [5,6]. On the demand side, reliable forecasts help system operators and energy managers anticipate the timing and magnitude of electricity demand, providing an essential information basis for demand-response design, flexible-load coordination, and market participation [7,8]. The potential sustainability benefits of these applications are substantial. Prior studies have reported demand flexibility equivalent to up to 18% of nominal consumption in commercial systems [9], peak-demand reductions of approximately 29% through intelligent load management [10], and carbon-emission reductions of approximately 14–20% under different demand-side management scenarios [11]. Taken together, these findings highlight the important role of accurate load prediction in advancing sustainable energy development.

Recently, the deployment of smart meters provides fine-grained observations of electricity use at the user side, making large-scale consumption data available for forecasting and energy management applications. This increasing data availability has made data-driven methods a major technical pathway for load forecasting. Methodologically, data-driven load forecasting has evolved from classical statistical and time-series models to more complex machine learning and deep learning approaches. Early studies commonly relied on linear or structured time-series methods, which provide interpretable formulations but may be limited in capturing nonlinear consumption patterns and complex temporal dependencies. Recent reviews in the energy literature show that machine learning and deep learning methods have been widely adopted for short-term load forecasting, including tree-based models, recurrent neural networks, convolutional architectures, and attention-based models [12–14]. These data-driven methods provide a diverse set of modeling tools for learning load patterns directly from historical observations and have shown promising empirical performance in load forecasting applications. Evidence from other industrial and operational forecasting contexts also indicates that machine learning methods can provide effective predictive support under complex and evolving conditions [15]. Along this line, a large body of research has sought to improve forecasting accuracy by developing more advanced model architectures.

However, compared to general time-series forecasting tasks, user-level load forecasting is further complicated by the heterogeneous and time-varying nature of user-side electricity consumption. Specifically, users may differ in consumption scale, operating rhythm, weekly regularity, and volatility, leading to distinct load patterns across users [16,17]; we refer to this characteristic as inter-user heterogeneity. Meanwhile, the load pattern of the same user may also evolve over time due to seasonal changes, holidays, business cycles, operational adjustments, or external disruptions [18–21]; we refer to this characteristic as temporal dynamics. From the perspective of the bias–variance trade-off in statistical learning, these two characteristics suggest that forecasting accuracy is affected not only by the choice of the model itself, but also by how training data are organized, how models are trained, and how predictions are selected or combined when multiple models are used. Specifically, inter-user heterogeneity and temporal dynamics indicate that load series from different users or temporal regimes may follow different underlying distributions. Therefore, if we pool all data across users and time regimes together to train a single model, it can increase sample size and reduce estimation variance, but may introduce bias by smoothing out user-specific or regime-specific load patterns. Consequently, such a strategy may yield less accurate predictions for instances whose patterns deviate from the pooled distribution. In

contrast, training models on more specialized subsets can better capture local regularities, but may increase variance due to limited data coverage [22,23].

Motivated by this observation, this study investigates day-ahead individual load forecasting from a forecasting strategy perspective. As shown in Figure 1, we define forecasting strategies as the set of procedures that determine how historical data are organized, how forecasting models are trained, and how trained models are applied, selected, or combined to generate user-level load forecasts. Based on this definition, we systematically compare three categories of forecasting strategies: homogeneous strategies that ignore both inter-user heterogeneity and temporal dynamics, inter-user heterogeneity-aware strategies that account for inter-user heterogeneity but do not explicitly model temporal dynamics, and temporal-dynamics-aware strategies that consider both inter-user heterogeneity and temporal dynamics. Such a strategy-level perspective allows us to examine how different ways of sharing training data, specializing models, and adapting model usage affect forecasting performance. The main contributions of this paper can be summarized as follows:

- We revisit day-ahead user level load forecasting from the perspective of inter-user heterogeneity and temporal dynamics in individual load patterns. This perspective highlights that forecasting performance depends not only on model architecture, but also on how models are trained and used across different users and time periods.
- We formulate a strategy-level framework for organizing and comparing forecasting methods, which allows us to examine how inter-user heterogeneity and temporal dynamics affect forecasting performance through different choices of data organization, model training, model selection, and prediction combination.
- We conduct extensive experiments on three real-world electricity consumption datasets from China. The empirical results compare representative machine learning and deep learning models under different forecasting strategies, providing evidence on how inter-user heterogeneity and temporal dynamics affect individual load forecasting performance. They further show how these characteristics can be leveraged through appropriate strategy design to improve forecasting accuracy.

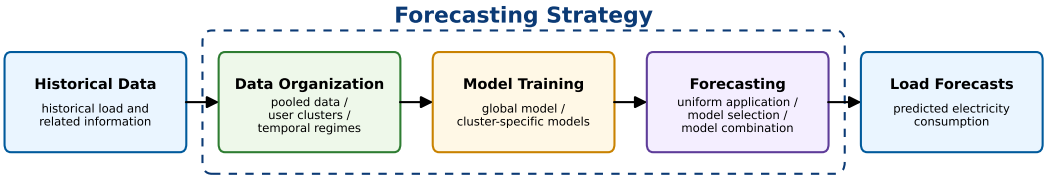


Figure 1. Overview of the user-level load forecasting process and the role of forecasting strategy. The forecasting strategy covers three key procedural decisions: how historical data are organized, how forecasting models are trained, and how trained models are applied, selected, or combined to generate load forecasts.

2. Related Work

2.1. Electricity Load Forecasting

Electricity load forecasting has long been an important topic in power system operation, electricity market management, demand response, and energy planning. With the development of smart meters and data-driven energy management systems, forecasting methodologies have gradually shifted from traditional statistical methods to machine learning and deep learning approaches. Recent review studies show that short-term electricity demand forecasting increasingly relies on data-driven methods because these approaches can better capture nonlinear demand patterns and complex temporal dependencies than conventional statistical forecasting models [24].

Machine learning methods have been widely adopted to improve forecasting accuracy under complex operating conditions. Models such as support vector machines, random forests, gradient boosting methods, and ensemble learning frameworks are capable of capturing nonlinear relationships between electricity demand and exogenous variables, including temperature, calendar effects, holidays, and operational schedules. Recent studies on smart-grid forecasting further demonstrate that machine learning methods are especially useful when load data contain irregular fluctuations and multiple external drivers [25].

In recent years, deep learning methods have become increasingly popular. Recurrent neural networks, long short-term memory networks, gated recurrent unit models, and convolutional neural network-based hybrid architectures have been widely used to capture temporal dependencies and hidden patterns in load time series. Hybrid deep learning frameworks further combine temporal feature extraction, nonlinear representation learning, and optimization strategies to improve forecasting performance under highly dynamic load conditions [26]. Recent review studies also emphasize that future developments in load forecasting should move beyond simply increasing model complexity and should instead consider learning paradigms, data structures, and forecasting contexts [27].

Despite the rapid advancement of data-driven load forecasting methods, most existing studies remain largely model-centric, focusing on improving predictive accuracy through more sophisticated forecasting architectures. However, in day-ahead forecasting at the user level, performance depends not only on the model class itself, but also on how models are trained, combined, and assigned to forecasting instances. This becomes especially important under inter-user heterogeneity and temporal dynamics, where pooled training may improve estimation stability but smooth out user- or state-specific patterns, while specialized training may improve local adaptation but reduce data coverage. Therefore, beyond developing stronger standalone models, it is also necessary to examine forecasting strategies that organize data sharing, model specialization, and adaptive model usage under heterogeneous and time-varying individual load patterns.

2.2. Inter-User Heterogeneity in Load Forecasting

Electricity users are highly heterogeneous in their consumption behaviors. Different users may exhibit distinct load patterns because of differences in business type, operating schedule, equipment composition, occupancy behavior, and electricity usage habits. Recent studies on building energy consumption prediction show that load classification and user-type differences are important factors affecting forecasting performance [28]. Similarly, studies on residential electricity forecasting demonstrate that driving factors and consumption characteristics may vary substantially across customers, even within the same broad user category [29].

This heterogeneity introduces important challenges for electricity load forecasting. A forecasting model that performs well for one type of user may not necessarily perform well for another. Users with relatively smooth and stable load curves may be adequately modeled using simple forecasting methods, whereas users with nonlinear or highly volatile load patterns may require more sophisticated machine learning or deep learning approaches. Studies on demand response and user consumption behavior further show that user-level electricity patterns can differ significantly due to operational flexibility, occupancy schedules, and behavioral diversity.

To address user heterogeneity, clustering and classification methods have been widely adopted in electricity load forecasting research. These methods group users with similar load characteristics into clusters so that forecasting models can be trained on more homogeneous datasets. By reducing intra-group variability, clustering-based approaches can improve forecasting interpretability and sometimes improve forecasting accuracy. Recent

work on electricity demand profile clustering demonstrates that unsupervised learning can reveal meaningful hidden structures in user-level electricity consumption data [30].

Another important issue concerns the relationship between data scale and model specialization. A specialized forecasting model trained for a small group of similar users may better capture local load characteristics, but limited sample size may prevent the model from learning stable parameters. In contrast, a generalized model trained on larger pooled datasets may exhibit stronger generalization ability, but may also smooth out distinctive user-specific characteristics. This trade-off between personalization and generalization is closely related to recent studies on deriving electricity consumption typologies from smart-meter datasets [31]. These findings suggest that user heterogeneity should not only be viewed as a data characteristic, but also as a factor influencing forecasting model suitability.

2.3. Temporal Dynamics in Load Forecasting

In addition to differences among users, electricity consumption behavior may also vary significantly over time for the same user. This means that heterogeneity is not only cross-sectional but also temporal. A user's electricity demand may change across seasons, weekdays and weekends, business cycles, holidays, operational adjustments, or external disruptions. Recent studies on real-time industrial load profiling show that electricity demand may contain structural heterogeneity and scale-dependent temporal dynamics that cannot be fully captured by static forecasting strategies [32].

Temporal heterogeneity implies that the forecasting model best suited for a particular user may not remain fixed over time. A forecasting model that performs well during one period may perform poorly in another if the user's load pattern changes substantially. For day-ahead individual forecasting, this issue is particularly important because forecasting systems must continuously adapt to evolving consumption behavior. Dynamic clustering studies demonstrate that updating user groups and load representations over time can better capture changing consumption structures [33].

Adaptive forecasting strategies aim to address this issue by allowing forecasting procedures to change according to user conditions and temporal states. Instead of relying on a single fixed forecasting model, adaptive frameworks combine multiple forecasting models and dynamically select the most suitable model according to current load characteristics. Pattern-clustering-based adaptive forecasting approaches have been used to improve short-term load forecasting performance under changing market conditions and demand variability [34].

However, most existing dynamic forecasting studies mainly focus on improving model flexibility itself without explicitly analyzing how temporal heterogeneity affects model suitability. Although many studies recognize that load patterns evolve over time, they often do not systematically investigate how such temporal changes influence forecasting model selection and forecasting performance. Recent spatio-temporal load portrait methods suggest that temporal dynamics and user differences should be jointly modeled, but the explicit relationship between temporal heterogeneity and forecasting strategy design remains insufficiently explored [35].

2.4. Research Gap

Existing studies remain largely model-centric, focusing mainly on improving forecasting accuracy through stronger model architectures or learning paradigms. However, in user-level load forecasting, prediction performance may also depend on how models are trained, combined, and used under heterogeneous and time-varying load patterns.

Although clustering and personalized learning methods have considered user differences, they often treat heterogeneity as a static grouping or personalization problem.

Load-profile similarity, however, does not necessarily imply similarity in forecasting model suitability, and the load pattern of the same user may also evolve over time. Therefore, it remains necessary to examine forecasting strategies that explicitly consider both inter-user heterogeneity and temporal dynamics.

To address this issue, this study investigates day-ahead individual load forecasting from a strategy-level perspective. We compare homogeneous global models, inter-user-heterogeneity-aware strategies, temporal-dynamics-aware strategies, and global–local hybrid strategies to understand how different ways of sharing data, specializing models, and adapting model usage affect forecasting performance.

3. Preliminaries and Problem Formulation

3.1. Day-Ahead Individual Load Forecasting

This study considers a day-ahead individual load forecasting problem for commercial users. Formally, let $\mathcal{U} = \{1, 2, \ldots, N\}$ denote the set of commercial users. For each user $i \in \mathcal{U}$, the historical daily electricity consumption is represented as $\mathbf{y}_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,T_i}\}$, where $y_{i,t}$ denotes the electricity consumption of user i on day t. In this study, we restrict the forecasting input to historical electricity consumption and do not incorporate exogenous variables such as weather, calendar information, or electricity prices. This choice is made to ensure that all forecasting models are evaluated under a common and controlled information set. While exogenous variables can be directly incorporated into some models, their integration into other architectures may involve different feature-encoding, fusion, and input-design choices, which can themselves affect forecasting performance. Including such model-specific implementations would therefore introduce an additional source of variation and make it more difficult to isolate the effects of the modeling strategies examined in this study. Accordingly, we focus on historical load observations to provide a consistent basis for comparing alternative strategies for individual load forecasting.

Formally, for each forecasting instance (i, t), the model input is defined as a fixed-length historical load window:

$$\mathbf{x}_{i,t} = \{y_{i,t-L+1}, y_{i,t-L+2}, \ldots, y_{i,t}\}, \tag{1}$$

where L is the look-back window length. A forecasting model f produces the next-day prediction as

$$\hat{y}_{i,t+1} = f(\mathbf{x}_{i,t}), \tag{2}$$

where $\hat{y}_{i,t+1}$ is the predicted electricity consumption of user i on day $t+1$.

Given a training set $\mathcal{D}^{\text{train}}$ consisting of forecasting instances from multiple users, the forecasting model is learned by minimizing the mean squared error (MSE) (We use the MSE loss in this study because the squared loss setting allows a standard closed-form bias–variance decomposition, which is useful for analytically illustrating how different forecasting strategies affect prediction error. Similar bias–variance trade-offs can also arise under other loss functions, such as the L1 loss, although their specific forms are loss-dependent.):

$$\min_{f \in \mathcal{F}} \frac{1}{|\mathcal{D}^{\text{train}}|} \sum_{(i,t) \in \mathcal{D}^{\text{train}}} (y_{i,t+1} - f(\mathbf{x}_{i,t}))^2, \tag{3}$$

where $\mathcal{F}$ denotes the hypothesis space of forecasting models.

3.2. Inter-User Heterogeneity and Temporal Dynamics

Commercial users' daily electricity consumption is shaped by their operational routines, business activities, equipment usage, and external conditions. As a result, individual

load patterns are not necessarily homogeneous across users or stable over time. In this study, we focus on two common characteristics of commercial-user load data: inter-user heterogeneity and temporal dynamics in individual load patterns.

Inter-user heterogeneity refers to differences in load patterns across users. Different commercial users may differ in electricity consumption scale, weekly usage regularity, seasonal behavior, variability level, and operating rhythm. For example, some users may exhibit clear weekday-weekend differences, whereas others may maintain relatively stable daily consumption throughout the week. Some users may show strong seasonal fluctuations, while others may present more irregular or weakly seasonal patterns. Even when user attributes such as business type, operating schedule, or equipment composition are not explicitly observed, their effects can still be reflected in the historical load series.

Temporal dynamics refer to changes in the load pattern of the same user over time. For a fixed commercial user, daily electricity consumption may evolve due to seasonal changes, holidays, business cycles, changes in operating conditions, or adjustments in energy-use behavior. Therefore, the load pattern observed in one period may not fully represent that in another period. Such temporal changes indicate that individual load series may contain different states over time, rather than following a single static pattern.

Taken together, inter-user heterogeneity and temporal dynamics suggest that commercial-user load data are heterogeneous across users and time-varying within users. From a data-driven perspective, these characteristics affect forecasting mainly through data distribution and model suitability. When samples from different users and time periods are pooled together, the training data may contain multiple consumption patterns with different statistical properties. A model trained on such pooled data can learn common regularities shared across users and periods, but may also smooth out patterns that are specific to certain users or temporal states. In contrast, restricting the training or selection process to more homogeneous users or periods may improve adaptation to local load patterns, but it also reduces the effective sample size and may make parameter estimation or model selection less reliable. Therefore, these domain characteristics motivate a strategy-level investigation of how forecasting models should share data, specialize to local patterns, and be assigned or combined for different prediction instances.

3.3. Forecasting Strategy

The basic forecasting formulation in Section 3.1 focuses on learning a forecasting model f for next-day load prediction. While the choice of model architecture is important, forecasting performance also depends on how models are trained, combined, and used across different users and time periods. We refer to this broader decision process as a forecasting strategy and provide its formal definition below.

Definition 1 (Forecasting Strategy)

Given a set of candidate model classes $\mathcal{M} = \{M_1, M_2, \ldots, M_K\}$, a forecasting strategy $\mathcal{S}$ specifies how forecasting models are trained and how their predictions are used for each forecasting instance. Specifically, a forecasting strategy consists of two components:

- Training component: defines the training scope used to fit each forecasting model. A training scope may correspond to the pooled data from all users, the data from a user group, the data from a temporal segment, or other subsets defined by the strategy.
- Prediction component: defines how the trained models are assigned, selected, or combined to generate the final prediction for a given forecasting instance (i, t).

Formally, a forecasting strategy $\mathcal{S}$ can be viewed as a procedure that takes the training data, candidate model classes, and a forecasting instance as inputs, and produces the final

prediction. Given the training set $\mathcal{D}^{\text{train}}$ and the candidate model classes $\mathcal{M}$, the training component of $\mathcal{S}$ first generates a set of trained predictors, denoted by

$$\mathcal{F}^{\mathcal{S}} = \mathcal{S}_{\text{train}}(\mathcal{D}^{\text{train}}, \mathcal{M}). \tag{4}$$

For a forecasting instance (i, t), the prediction component of $\mathcal{S}$ then assigns, selects, or combines predictors from $\mathcal{F}^{\mathcal{S}}$ to produce the final prediction:

$$\hat{y}^{\mathcal{S}}_{i,t+1} = \mathcal{S}_{\text{pred}}(\mathbf{x}_{i,t}, i, t; \mathcal{F}^{\mathcal{S}}). \tag{5}$$

This definition provides a unified view of the strategies considered in this study. For example, a simple homogeneous strategy trains a predictor on the pooled training set $\mathcal{D}^{\text{train}}$ and applies it to all forecasting instances. A user-cluster-specific strategy, in contrast, partitions users into clusters, trains predictors on the corresponding cluster-level subsets, and applies the predictor associated with the group of user i when forecasting $y_{i,t+1}$. The next subsection analyzes why strategy design is non-trivial and critical for individual load forecasting from a statistical learning perspective.

3.4. A Statistical Learning View of Forecasting Strategy Design

From the model perspective, a forecasting strategy is built upon a set of candidate model classes $\mathcal{M} = \{M_1, M_2, \ldots, M_K\}$. The use of multiple model classes is motivated by the fact that different models have different inductive biases. Each model class M_k imposes its own hypothesis space and learning preference, and thus may be more suitable for certain types of load patterns than others [36,37]. Under inter-user heterogeneity and temporal dynamics, the relationship between the historical input $\mathbf{x}_{i,t}$ and the next-day load $y_{i,t+1}$ may vary across users, user groups, or temporal states. Therefore, a single model class is not necessarily expected to be uniformly optimal for all forecasting instances. Including multiple candidate models allows a forecasting strategy $\mathcal{S}$ to exploit model-specific strengths by assigning, selecting, or combining models for different users or time periods.

From the data perspective, how training data are organized is also critical to forecasting performance. Let P denote the overall distribution of the forecasting target. Due to inter-user heterogeneity and temporal dynamics, the overall data may contain multiple segments with different load patterns. Here, a segment refers to a subset of forecasting instances that share similar user-level or temporal load characteristics. Depending on the strategy, a segment may correspond to an individual user, a user cluster, a temporal regime, or their combination. Accordingly, the overall distribution can be viewed as:

$$P = \sum_{g \in \mathcal{G}} p_g P_g, \tag{6}$$

where $g \in \mathcal{G}$ denotes a segment. P_g is the distribution of forecasting instances in segment g, and p_g is the proportion of this segment in the overall data.

Given a forecasting strategy $\mathcal{S}$, applying it to the available training data induces an effective prediction system, denoted by $\hat{h}^{\mathcal{S}}$. Depending on the strategy, this prediction system may correspond to a single trained model, a selected model from multiple candidates, or a combination of several models. For a forecasting instance (i, t), the prediction produced by strategy $\mathcal{S}$ can be written as

$$\hat{y}^{\mathcal{S}}_{i,t+1} = \hat{h}^{\mathcal{S}}(\mathbf{x}_{i,t}), \tag{7}$$

Thus, the expected prediction risk of $\hat{h}^{\mathcal{S}}$ over the overall distribution can then be represented as

$$\mathcal{R}(\hat{h}^{\mathcal{S}}) = \mathbb{E}_{(\mathbf{x},y)\sim P}\left[\left(y - \hat{h}^{\mathcal{S}}(\mathbf{x})\right)^2\right] = \sum_{g\in\mathcal{G}} p_g \mathcal{R}_g(\hat{h}^{\mathcal{S}}), \tag{8}$$

where

$$\mathcal{R}_g(\hat{h}^{\mathcal{S}}) = \mathbb{E}_{(\mathbf{x},y)\sim P_g}\left[\left(y - \hat{h}^{\mathcal{S}}(\mathbf{x})\right)^2\right]. \tag{9}$$

The above risk can be conceptually decomposed into irreducible error, squared bias, and variance:

$$\mathcal{R}(\hat{h}^{\mathcal{S}}) = \sum_{g\in\mathcal{G}} p_g \sigma_g^2 + \sum_{g\in\mathcal{G}} p_g B_g^2(\hat{h}^{\mathcal{S}}) + \sum_{g\in\mathcal{G}} p_g V_g(\hat{h}^{\mathcal{S}}). \tag{10}$$

where σ_g^2 denotes the irreducible error in segment g, which is independent of the forecasting strategy. The term $B_g^2(\hat{h}^{\mathcal{S}})$ captures the systematic mismatch between the effective prediction system $\hat{h}^{\mathcal{S}}$ and the load pattern in segment g. The term $V_g(\hat{h}^{\mathcal{S}})$ captures the estimation instability of $\hat{h}^{\mathcal{S}}$ in segment g, which mainly arises from finite-sample model training. Appendix A provides the details about this decomposition.

Different training scopes can therefore affect both $B_g^2(\hat{h}^{\mathcal{S}})$ and $V_g(\hat{h}^{\mathcal{S}})$ by changing the distribution and the size of the data used for model fitting. Specifically, a strategy based on pooled training data can use more samples and may therefore reduce the variance term $V_g(\hat{h}^{\mathcal{S}})$, but it may also learn an averaged pattern that does not fit some user groups or temporal states well, leading to a larger bias term $B_g^2(\hat{h}^{\mathcal{S}})$ in those segments. In contrast, a strategy based on more localized training data may better capture the load patterns of particular segments, thereby reducing the bias term, but the smaller effective sample size may increase the variance term. Therefore, forecasting strategy design should balance this trade-off between bias reduction and variance control. In this paper, we examine several representative strategies to understand how different ways of organizing training data and using forecasting models affect prediction performance.

4. Forecasting Strategies

4.1. Candidate Forecasting Models and Strategy Overview

In this study, we consider a diverse set of representative forecasting models as candidate model classes, denoted by

$$\mathcal{M} = \{\text{LightGBM}, \text{XGBoost}, \text{N-BEATS}, \text{LSTM}, \text{Transformer}, \text{PatchTST}\}.$$

These models cover two major categories of data-driven forecasting methods: tree-based machine learning models and deep learning models.

The tree-based models include LightGBM [38] and XGBoost [39], which are widely used for structured prediction tasks and capture nonlinear relationships through ensemble decision trees. In this study, they are trained using features constructed from historical load observations. The deep learning models include N-BEATS [40], LSTM [41], Transformer [42], and PatchTST [43], which directly learn temporal representations from historical load sequences and provide different inductive biases for modeling sequential dependencies and temporal patterns.

The six models serve as the common candidate pool for all forecasting strategies. The main focus of this study is to examine how different strategy designs exploit inter-user heterogeneity and temporal dynamics in individual load patterns. Specifically, we organize the strategies according to whether they assume a homogeneous global forecasting rule, exploit static inter-user heterogeneity, or further account for temporal dynamics in

individual load patterns. Figure 2 summarizes the forecasting strategies examined in this study.

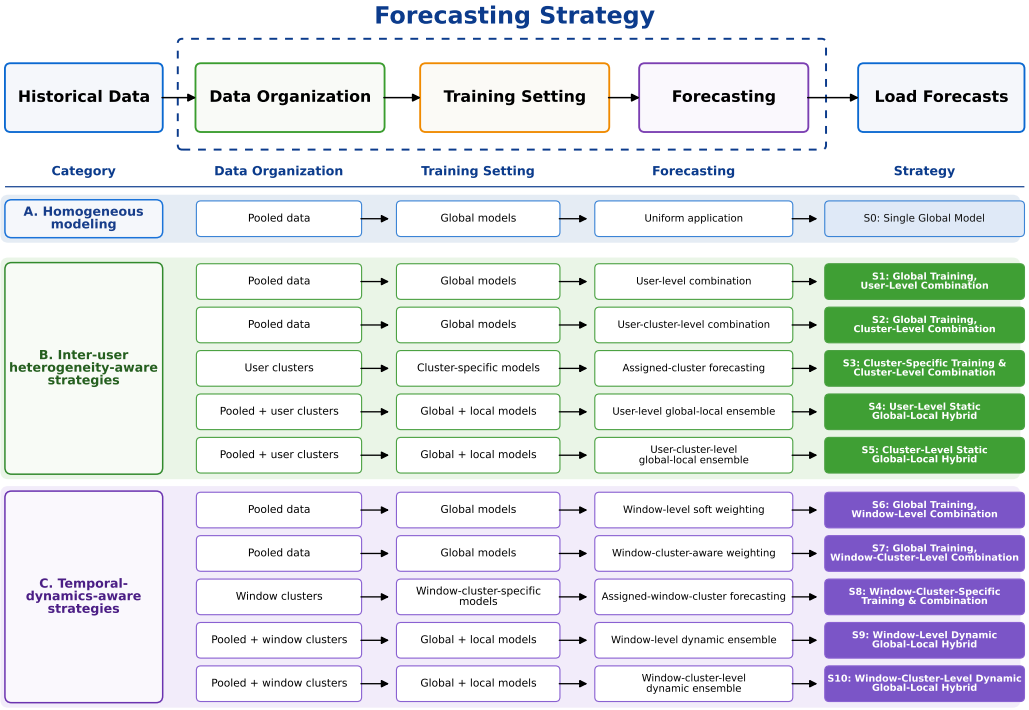


Figure 2. Overview of the forecasting strategy construction. The upper panel illustrates the overall user-level load forecasting process. The lower panel shows how the compared strategies are constructed through different combinations of these three design components.

4.2. Inter-User-Heterogeneity-Aware Strategies

In this subsection, we first focus on strategies that explicitly account for inter-user heterogeneity but do not explicitly model temporal dynamics in individual load patterns. These strategies are motivated by the observation that different users, or different groups of users, may exhibit different load patterns and therefore may benefit from different forecasting models or training scopes. Five specific strategies fall into this category: global training with user-level combination, global training with cluster-level combination, cluster-specific training and combination, user-level static global–local hybrid, and cluster-level static global–local hybrid.

4.2.1. Global Training with User-Level Combination

The first strategy exploits inter-user heterogeneity by allowing each user to use a personalized combination of global candidate predictors. Here, global indicates that each candidate model is trained using the pooled training data from all users. Formally, for each candidate model class $M_k \in \mathcal{M}$, the global predictor is trained as

$$\hat{f}_k^G = A_k(\mathcal{D}^{\text{train}}), \tag{11}$$

where A_k denotes the learning algorithm of model class M_k, and $\mathcal{D}^{\text{train}}$ denotes the pooled training data from all users. Let

$$\mathcal{F}^G = \{\hat{f}_k^G : M_k \in \mathcal{M}\} \tag{12}$$

denote the pool of globally trained candidate predictors. After obtaining these predictors, the strategy learns a user-specific combination of these predictors based on validation performance. For each user i, the combination weight vector is obtained by

$$\mathbf{w}_i^G = \arg\min_{\mathbf{w}\in\Delta^K} \sum_{t\in\mathcal{V}_i} \left(y_{i,t+1} - \sum_{k=1}^{K} w_k \hat{f}_k^G(\mathbf{x}_{i,t}) \right)^2, \tag{13}$$

where $\mathcal{V}_i$ denotes the validation samples of user i, and $\Delta^K = \{\mathbf{w} \in \mathbb{R}_+^K : \sum_{k=1}^K w_k = 1\}$ is the probability simplex. The prediction system induced by this strategy for user i is

$$\hat{h}_i^{\mathrm{UL}}(\mathbf{x}_{i,t}) = \sum_{k=1}^{K} w_{i,k}^G \hat{f}_k^G(\mathbf{x}_{i,t}), \tag{14}$$

and the prediction is given by

$$\hat{y}_{i,t+1}^{\mathrm{UL}} = \hat{h}_i^{\mathrm{UL}}(\mathbf{x}_{i,t}). \tag{15}$$

4.2.2. Global Training with Cluster-Level Combination

The second strategy follows the same global training procedure as the user-level combination strategy, but learns the model combination at the cluster level. This design reduces the instability that may arise from excessive user-level heterogeneity by allowing users within the same cluster to share a common combination rule, thereby examining the appropriate granularity at which inter-user heterogeneity should be captured. Specifically, users are first clustered according to the similarity of their historical load patterns. For each user, the historical daily load series is aggregated into a representative weekly load profile, which summarizes the user's average consumption pattern across the seven days of a week. We then apply a DTW-based k-medoids algorithm to these weekly profiles, where Dynamic Time Warping (DTW) measures the similarity between users and each cluster is represented by its medoid. Let $c_i \in \{1, 2, \ldots, C\}$ denote the resulting static cluster assignment of user i. For each user cluster c, the strategy learns the combination weights by:

$$\mathbf{w}_c^G = \arg\min_{\mathbf{w}\in\Delta^K} \sum_{(i,t)\in\mathcal{V}:c_i=c} \left(y_{i,t+1} - \sum_{k=1}^{K} w_k \hat{f}_k^G(\mathbf{x}_{i,t}) \right)^2, \tag{16}$$

where $\Delta^K = \{\mathbf{w} \in \mathbb{R}_+^K : \sum_{k=1}^K w_k = 1\}$ denotes the probability simplex, and $\mathcal{V}$ denotes the validation set. The final prediction system for cluster c is then defined as

$$\hat{h}_c^{\mathrm{CL}}(\mathbf{x}_{i,t}) = \sum_{k=1}^{K} w_{c,k}^G \hat{f}_k^G(\mathbf{x}_{i,t}). \tag{17}$$

For user i belonging to cluster c_i, the prediction is generated by

$$\hat{y}_{i,t+1}^{\mathrm{CL}} = \hat{h}_{c_i}^{\mathrm{CL}}(\mathbf{x}_{i,t}). \tag{18}$$

4.2.3. Cluster-Specific Training and Cluster-Level Combination

This strategy further incorporates inter-user heterogeneity into the model training stage. Unlike the previous strategies, which train candidate predictors on the pooled training data, this strategy trains separate predictors for different user clusters. Using the same DTW-based user clustering results, for each cluster c, we construct a cluster-specific

training set $\mathcal{D}_c^{\text{train}}$ using only the training samples from users in that cluster. Then, for each candidate model class $M_k \in \mathcal{M}$, a local predictor for cluster c is trained as

$$\hat{f}_{k,c}^{C} = A_k(\mathcal{D}_c^{\text{train}}), \tag{19}$$

where $\hat{f}_{k,c}^{C}$ denotes the predictor of model class M_k trained on cluster c. Let

$$\mathcal{F}_c^{C} = \{\hat{f}_{k,c}^{C} : M_k \in \mathcal{M}\} \tag{20}$$

denote the pool of cluster-specific local predictors for cluster c.

After obtaining the local predictors, the strategy learns the combination weights of them for each cluster based on validation performance. Specifically, for cluster c, the combination weight vector is obtained by

$$\mathbf{w}_c^{C} = \arg\min_{\mathbf{w}\in\Delta^K} \sum_{(i,t)\in\mathcal{V}:c_i=c} \left(y_{i,t+1} - \sum_{k=1}^{K} w_k \hat{f}_{k,c}^{C}(\mathbf{x}_{i,t}) \right)^2, \tag{21}$$

where $\Delta^K = \{\mathbf{w} \in \mathbb{R}_+^K : \sum_{k=1}^K w_k = 1\}$ denotes the probability simplex. The final prediction system induced by this strategy for cluster c is

$$\hat{h}_c^{\text{CS}}(\mathbf{x}_{i,t}) = \sum_{k=1}^{K} w_{c,k}^{C} \hat{f}_{k,c}^{C}(\mathbf{x}_{i,t}). \tag{22}$$

For user i belonging to cluster c_i, the prediction is generated by

$$\hat{y}_{i,t+1}^{\text{CS}} = \hat{h}_{c_i}^{\text{CS}}(\mathbf{x}_{i,t}). \tag{23}$$

4.2.4. Static Global–Local Hybrid Variants

After considering strategies that use globally trained predictors or local predictors separately, we further examine whether the two sources of information can be combined for performance improvements. Therefore, global–local hybrid strategies integrate globally trained predictors with cluster-specific local predictors. We consider two variants, which differ in the granularity at which the global–local ensemble is learned. For user i belonging to cluster c_i, the global–local candidate pool consists of the six globally trained predictors and the six local predictors trained within cluster c_i:

$$\mathcal{E}_i^{\text{SH}} = \mathcal{F}^{G} \cup \mathcal{F}_{c_i}^{C}. \tag{24}$$

The user-level variant learns a separate ensemble for each user, while the cluster-level variant learns one shared ensemble for each static user cluster.

For the user-level static hybrid, we apply forward selection on the validation samples of user i to construct an ensemble from $\mathcal{E}_i^{\text{SH}}$. Compared with the previous combination strategies, the global–local candidate pool has a substantially higher dimension, making direct optimization of continuous combination weights more difficult and potentially less stable on the limited user-level validation data. We therefore adopt forward selection as a computationally efficient heuristic for constructing the ensemble. Specifically, starting from an empty ensemble, the procedure repeatedly adds the predictor whose inclusion yields the lowest validation MSE of the ensemble average. Repeated selection of the same predictor is

allowed. Let $\mathcal{A}_i^{\mathrm{SH,U}} = \{a_{i,1}, a_{i,2}, \ldots, a_{i,J}\}$ denote the selected multiset of predictors, where J is the ensemble size. The prediction is then given by

$$\hat{y}_{i,t+1}^{\mathrm{SH,U}} = \frac{1}{J} \sum_{j=1}^{J} a_{i,j}(\mathbf{x}_{i,t}). \tag{25}$$

For the cluster-level static hybrid, the same forward-selection procedure is applied at the user-cluster level. For each cluster c, we construct the global–local candidate pool

$$\mathcal{E}_c^{\mathrm{SH}} = \mathcal{F}^G \cup \mathcal{F}_c^C, \tag{26}$$

and learn a shared ensemble using the validation samples from users in cluster c. Let $\mathcal{A}_c^{\mathrm{SH,C}} = \{a_{c,1}, a_{c,2}, \ldots, a_{c,J}\}$ denote the selected multiset of predictors for cluster c. For user i belonging to cluster c_i, the prediction is generated by

$$\hat{y}_{i,t+1}^{\mathrm{SH,C}} = \frac{1}{J} \sum_{j=1}^{J} a_{c_i,j}(\mathbf{x}_{i,t}). \tag{27}$$

4.3. Temporal-Dynamics-Aware Strategies

The strategies in the previous subsection exploit inter-user heterogeneity by combining or training models according to users or static user groups. They implicitly assume that the load pattern of a user does not change over time and treat each user's model usage as fixed. To further account for temporal dynamics within individual load patterns, we consider strategies that operate at the forecasting-window level. These strategies allow different forecasting windows, even from the same user, to be associated with different temporal states and therefore different model combinations or global–local routing decisions. Five strategies are considered in this category: global training with window-level combination, global training with window-cluster-level combination, window-cluster-specific training and combination, window-level dynamic global–local hybrid, and window-cluster-level dynamic global–local hybrid.

4.3.1. Global Training with Window-Level Combination

The first strategy keeps the training process global but allows the model combination to vary across forecasting windows. As in the previous globally trained strategies, each candidate model class $M_k \in \mathcal{M}$ is trained using the pooled training data from all users:

$$\hat{f}_k^G = A_k(\mathcal{D}^{\mathrm{train}}). \tag{28}$$

Let

$$\mathcal{F}^G = \{\hat{f}_k^G : M_k \in \mathcal{M}\} \tag{29}$$

denote the pool of globally trained candidate predictors.

Different from static user-level combination, the combination weights are no longer fixed for each user. Instead, they are determined for each forecasting window according to its current historical load state. Specifically, after the candidate predictors in $\mathcal{F}^G$ are trained, we use their historical forecasting errors to train a window-level expected-error model implemented as a multilayer perceptron (MLP). Given the current historical load window, the MLP estimates the expected forecasting error of each candidate predictor and is trained by minimizing the loss between the estimated and observed errors. For each forecasting instance (i, t) and each candidate predictor $\hat{f}_k^G$, the selector estimates the expected forecasting error of that predictor under the current window state. These estimated errors are then converted into window-specific combination weights over the

globally trained predictors. Let $w^{G}_{i,t,k}$ denote the resulting weight assigned to predictor $\hat{f}^{G}_{k}$ for forecasting instance (i,t). The effective prediction system is defined as

$$\hat{h}^{\mathrm{WL}}_{i,t}(\mathbf{x}_{i,t}) = \sum_{k=1}^{K} w^{G}_{i,t,k} \hat{f}^{G}_{k}(\mathbf{x}_{i,t}), \tag{30}$$

and the prediction is then generated by

$$\hat{y}^{\mathrm{WL}}_{i,t+1} = \hat{h}^{\mathrm{WL}}_{i,t}(\mathbf{x}_{i,t}). \tag{31}$$

4.3.2. Global Training with Window-Cluster-Level Combination

The second strategy also keeps the training process global, but incorporates window-cluster information into the model combination rule. Forecasting windows are first grouped according to the similarity of their historical load patterns. Specifically, we apply a DTW-based k-medoids algorithm to the historical load windows, where Dynamic Time Warping (DTW) is used to measure the similarity between windows and each cluster is represented by its medoid. Let $s_{i,t} \in \{1, 2, \ldots, S\}$ denote the window-cluster assignment of forecasting instance (i,t), where S is the number of window clusters. For a forecasting instance (i,t), we assign its historical window $\mathbf{x}_{i,t}$ to the closest window cluster according to its distance to the cluster centroids:

$$s_{i,t} = \arg \min_{s \in \{1,\ldots,S\}} d(\mathbf{x}_{i,t}, \boldsymbol{\mu}_{s}), \tag{32}$$

where $\boldsymbol{\mu}_{s}$ denotes the medoid of window cluster s, and $d(\cdot,\cdot)$ is the distance measure used for window clustering.

The candidate predictors are still globally trained, but their combination is learned for each window cluster. Specifically, for each window cluster s, we determine a cluster-specific weight vector by minimizing the forecasting error of the combined prediction over the validation samples assigned to that cluster:

$$\mathbf{w}^{G,\mathrm{WCL}}_{s} = \arg \min_{\mathbf{w} \in \Delta^{K}} \sum_{(i,t) \in \mathcal{V}: s_{i,t}=s} \left(y_{i,t+1} - \sum_{k=1}^{K} w_{k} \hat{f}^{G}_{k}(\mathbf{x}_{i,t}) \right)^{2}, \tag{33}$$

where $\Delta^{K} = \left\{ \mathbf{w} \in \mathbb{R}^{K}_{+} : \sum_{k=1}^{K} w_{k} = 1 \right\}$ denotes the probability simplex. The learned weights are shared by all forecasting instances assigned to the same window cluster. Let $w^{G,\mathrm{WCL}}_{s,k}$ denote the weight assigned to predictor $\hat{f}^{G}_{k}$ in window cluster s. For forecasting instance (i,t), the effective prediction system is defined as

$$\hat{h}^{\mathrm{WCL}}_{i,t}(\mathbf{x}_{i,t}) = \sum_{k=1}^{K} w^{G,\mathrm{WCL}}_{s_{i,t},k} \hat{f}^{G}_{k}(\mathbf{x}_{i,t}), \tag{34}$$

and the prediction is generated by

$$\hat{y}^{\mathrm{WCL}}_{i,t+1} = \hat{h}^{\mathrm{WCL}}_{i,t}(\mathbf{x}_{i,t}). \tag{35}$$

4.3.3. Window-Cluster-Specific Training and Combination

The third strategy further uses window clusters in the training stage. In this dynamic setting, a local predictor refers to a window-cluster-specific predictor trained only on the data from a particular window cluster. For each window cluster s, we construct a window-cluster-specific training set $\mathcal{D}^{\mathrm{train}}_{s}$ using only the forecasting instances assigned to that

window cluster. For each candidate model class $M_k \in \mathcal{M}$, a local predictor for window cluster s is trained as

$$\hat{f}_{k,s}^{W} = A_k(\mathcal{D}_s^{\text{train}}), \tag{36}$$

where $\hat{f}_{k,s}^{W}$ denotes the predictor of model class M_k trained on window cluster s. Let

$$\mathcal{F}_s^{W} = \{\hat{f}_{k,s}^{W} : M_k \in \mathcal{M}\} \tag{37}$$

denote the pool of window-cluster-specific local predictors for cluster s. After obtaining the local predictors, the strategy learns the combination weights of them for each window cluster based on validation performance. For window cluster s, the combination weight vector is obtained by

$$\mathbf{w}_s^{W} = \arg\min_{\mathbf{w}\in\Delta^K} \sum_{(i,t)\in\mathcal{V}:s_{i,t}=s} \left(y_{i,t+1} - \sum_{k=1}^{K} w_k \hat{f}_{k,s}^{W}(\mathbf{x}_{i,t}) \right)^2, \tag{38}$$

where $\Delta^K = \{\mathbf{w} \in \mathbb{R}_+^K : \sum_{k=1}^K w_k = 1\}$ denotes the probability simplex. The effective prediction system induced by this strategy for window cluster s is

$$\hat{h}_s^{\text{WCS}}(\mathbf{x}_{i,t}) = \sum_{k=1}^{K} w_{s,k}^{W} \hat{f}_{k,s}^{W}(\mathbf{x}_{i,t}). \tag{39}$$

For a forecasting instance (i,t), its historical window $\mathbf{x}_{i,t}$ is first assigned to a window cluster $s_{i,t}$ according to Equation (32). The prediction is then generated by

$$\hat{y}_{i,t+1}^{\text{WCS}} = \hat{h}_{s_{i,t}}^{\text{WCS}}(\mathbf{x}_{i,t}). \tag{40}$$

4.3.4. Dynamic Global–Local Hybrid Variants

We further develop dynamic global–local hybrid strategies that adapt model usage to the current forecasting window. For each forecasting instance, the strategy combines globally trained predictors with local predictors associated with the current temporal regime. The combination weights are allowed to vary across windows, enabling the forecasting system to balance global stability and regime-specific information according to the user's current load state.

For a forecasting instance (i,t), the historical load window $\mathbf{x}_{i,t}$ is assigned to a window cluster $s_{i,t}$ according to Equation (32). The candidate model pool then consists of two parts: the global predictors and the local ones for the assigned window cluster

$$\mathcal{E}_{i,t} = \{\hat{f}_m^{\text{G}} : m \in \mathcal{M}\} \cup \{\hat{f}_{m,s_{i,t}}^{\text{L}} : m \in \mathcal{M}\}. \tag{41}$$

Thus, each forecasting window can use both global information shared across all users and local information specific to its current temporal state.

The window-level dynamic global–local hybrid learns a shared combination function for all forecasting windows. Following the window-level model-selection approach described above, we implement this function using a multilayer perceptron (MLP). The MLP takes the current historical load window $\boldsymbol{x}_{i,t}$ as input and outputs a set of combination weights over the available global and local predictors:

$$\boldsymbol{\alpha}_{i,t} = G(\boldsymbol{x}_{i,t}). \tag{42}$$

These weights are used to form the final weighted prediction:

$$\hat{y}_{i,t+1}^{\mathrm{DH,W}} = \sum_{e \in \mathcal{E}_{i,t}} \alpha_{i,t,e} \hat{y}_{i,t+1}^{e}. \tag{43}$$

The MLP is trained on the validation data by minimizing the MSE between the weighted prediction and the observed load:

$$\min_{G} \sum_{(i,t) \in \mathcal{V}} \left(y_{i,t+1} - \sum_{e \in \mathcal{E}_{i,t}} \alpha_{i,t,e} \hat{y}_{i,t+1}^{e} \right)^2. \tag{44}$$

The window-cluster-level dynamic global–local hybrid follows the same prediction form as the window-level variant, but learns the weight-generation function separately for each temporal regime. Specifically, each forecasting window is first assigned to a window cluster using the DTW-based k-medoids model, and a separate MLP $G_s(\cdot)$ is trained using only the validation samples in cluster s. Thus, the window-level variant shares one MLP across all forecasting windows, whereas the window-cluster-level variant allows different temporal regimes to learn distinct combination mechanisms.

5. Experiments

5.1. Experimental Setup

5.1.1. Dataset and Forecasting Task

The experiments are conducted on three real-world user-level electricity consumption datasets from China. Table 1 summarizes the basic information of these datasets. Following the problem formulation in Section 3, we focus on day-ahead individual load forecasting, where the historical load observations from the past $L = 14$ days are used to predict the electricity consumption of the next day. For D1, we use 1 March to 30 June 2024 for training, 1 July to 31 July 2024 for validation, and 1 August to 31 August 2024 for testing. For D2, we use 1 March to 31 May 2026 for training, 1 June to 15 June 2026 for validation, and 16 June to 30 June 2026 for testing. For D3, we use 1 September to 30 November 2024 for training, 1 December to 15 December 2024 for validation, and 16 December to 31 December 2024 for testing. Only historical load observations are used as forecasting inputs, without incorporating any exogenous variables. We conduct a descriptive analysis of D1 to show inter-user heterogeneity and temporal dynamics, with the results reported in Appendix B.

Table 1. Summary of the datasets used in the experiments

Dataset	Source	Period	Days	Users	Avg. Daily Load per User (kWh)	Median Daily Load per User (kWh)
D1	Eastern China	1 March 2024 to 31 August 2024	184	1855	3864.42	2164.32
D2	Western China	1 March 2026 to 30 June 2026	122	1374	1086.22	122.67
D3	Eastern China	1 September 2024 to 31 December 2024	122	1971	3767.14	2047.10

5.1.2. Evaluation Metrics

The main evaluation metrics are mean absolute error (MAE), mean squared error (MSE), and weighted absolute percentage error (WAPE). MAE is computed as the average absolute error over all users and all test forecasting points:

$$\mathrm{MAE} = \frac{1}{|\mathcal{D}^{\mathrm{test}}|} \sum_{(i,t)\in\mathcal{D}^{\mathrm{test}}} |y_{i,t+1} - \hat{y}_{i,t+1}|. \tag{45}$$

MSE is computed as the average squared error over all users and all test forecasting points:

$$\mathrm{MSE} = \frac{1}{|\mathcal{D}^{\mathrm{test}}|} \sum_{(i,t)\in\mathcal{D}^{\mathrm{test}}} (y_{i,t+1} - \hat{y}_{i,t+1})^2. \tag{46}$$

WAPE is computed as

$$\mathrm{WAPE} = \frac{\sum_{(i,t)\in\mathcal{D}^{\mathrm{test}}} |y_{i,t+1} - \hat{y}_{i,t+1}|}{\sum_{(i,t)\in\mathcal{D}^{\mathrm{test}}} y_{i,t+1}}. \tag{47}$$

MAE and MSE measure absolute forecasting errors, while WAPE measures the aggregate absolute forecasting error relative to the total actual load. These metrics are used to compare the overall forecasting performance of different strategies.

5.2. Overall Performance Comparison

Tables 2–4 report the overall forecasting performance across the three datasets. Several findings emerge from the results. First, forecasting strategy design has a substantial impact on prediction performance. Most strategies that explicitly account for inter-user heterogeneity or temporal dynamics outperform the best standalone global model, confirming that how models are trained, selected, and combined is an important determinant of forecasting accuracy.

Table 2. Overall forecasting performance on Dataset D1. Values are reported as mean ± standard deviation. Values in parentheses indicate relative improvements over the best single global model for each metric. *** denotes statistical significance at the 1% level based on two-sided paired *t*-tests.

Strategy	MAE	MSE	WAPE (%)
LightGBM	521.80 ± 23.24	3,573,919 ± 1,214,387	13.36 ± 0.40
XGBoost	508.26 ± 19.22	2,666,578 ± 808,087	13.02 ± 0.34
N-BEATS	466.00 ± 10.59	1,468,177 ± 330,856	11.94 ± 0.32
LSTM	455.61 ± 10.89	1,526,477 ± 342,823	11.67 ± 0.31
Transformer	463.62 ± 14.45	1,621,450 ± 381,729	11.88 ± 0.37
PatchTST	453.89 ± 11.68	1,478,340 ± 342,828	11.63 ± 0.34
Global Training, User-Level Combination	449.23 ± 11.07 (1.03%) ***	1,390,338 ± 283,391 (5.30%) ***	11.51 ± 0.32 (1.03%) ***
Global Training, Cluster-Level Combination	448.27 ± 10.82 (1.24%) ***	1,426,222 ± 316,429 (2.86%) ***	11.49 ± 0.31 (1.24%) ***
Cluster-Specific Training & Cluster-Level Combination	450.84 ± 9.80 (0.67%) ***	1,409,894 ± 310,804 (3.97%) ***	11.55 ± 0.33 (0.67%) ***
User-Level Static Global–Local Hybrid	446.86 ± 10.12 (1.55%) ***	1,393,103 ± 308,615 (5.11%) ***	11.45 ± 0.30 (1.55%) ***
Cluster-Level Static Global–Local Hybrid	449.42 ± 10.29 (0.99%) ***	1,418,260 ± 318,889 (3.40%) ***	11.52 ± 0.28 (0.99%) ***
Global Training, Window-Level Combination	448.68 ± 10.89 (1.15%) ***	1,403,470 ± 277,184 (4.41%) ***	11.50 ± 0.32 (1.15%) ***
Global Training, Window-Cluster-Level Combination	445.48 ± 11.18 (1.85%) ***	1,392,300 ± 290,463 (5.17%) ***	11.41 ± 0.34 (1.85%) ***
Window-Cluster-Specific Training & Combination	447.26 ± 10.86 (1.46%) ***	1,404,740 ± 289,985 (4.32%) ***	11.46 ± 0.31 (1.46%) ***
Window-Level Dynamic Global–Local Hybrid	444.98 ± 11.42 (1.96%) ***	1,370,739 ± 286,659 (6.64%) ***	11.40 ± 0.33 (1.96%) ***
Window-Cluster-Level Dynamic Global–Local Hybrid	445.15 ± 11.27 (1.93%) ***	1,421,830 ± 304,326 (3.16%) ***	11.41 ± 0.36 (1.93%) ***

Second, both inter-user heterogeneity and temporal dynamics provide useful information for strategy design. Most inter-user-heterogeneity-aware strategies improve upon the single-model baseline, indicating that model suitability differs across users. Incorporating temporal dynamics on top of inter-user heterogeneity often yields further gains, suggesting that the appropriate model may also change over time for the same user.

Table 3. Overall forecasting performance on Dataset D2. Values are reported as mean ± standard deviation. Values in parentheses indicate relative improvements over the best single global model for each metric. *** denotes statistical significance at the 1% level based on two-sided paired *t*-tests.

Strategy	MAE	MSE	WAPE(%)
LightGBM	167.07 ± 14.44	561,128 ± 183,927	14.94 ± 0.98
XGBoost	168.89 ± 14.28	581,136 ± 193,351	15.10 ± 0.97
N-BEATS	168.71 ± 14.30	580,799 ± 182,577	15.08 ± 0.95
LSTM	167.13 ± 14.22	557,512 ± 181,635	14.94 ± 0.96
Transformer	169.88 ± 14.44	599,745 ± 185,533	15.19 ± 0.99
PatchTST	169.08 ± 14.93	555,107 ± 187,866	15.11 ± 0.93
Global Training, User-Level Combination	166.52 ± 14.48 (0.33%)	561,032 ± 186,467 (−1.07%) ***	14.88 ± 0.94 (0.35%)
Global Training, Cluster-Level Combination	165.33 ± 14.07 (1.04%) ***	548,620 ± 178,463 (1.17%) ***	14.78 ± 0.95 (1.04%) ***
Cluster-Specific Training & Cluster-Level Combination	166.11 ± 14.32 (0.57%)	539,246 ± 179,299 (2.86%) ***	14.85 ± 0.98 (0.57%)
User-Level Static Global–Local Hybrid	165.21 ± 13.85 (1.11%) ***	533,936 ± 168,331 (3.81%) ***	14.77 ± 0.97 (1.09%) ***
Cluster-Level Static Global–Local Hybrid	165.52 ± 14.25 (0.93%) ***	536,620 ± 176,203 (3.33%) ***	14.80 ± 0.95 (0.94%) ***
Global Training, Window-Level Combination	164.21 ± 14.29 (1.71%) ***	538,553 ± 177,186 (2.98%) ***	14.68 ± 0.91 (1.74%) ***
Global Training, Window-Cluster-Level Combination	165.45 ± 14.27 (0.97%) ***	549,276 ± 180,150 (1.05%) ***	14.79 ± 0.93 (0.98%) ***
Window-Cluster-Specific Training & Combination	166.72 ± 13.77 (0.21%)	543,215 ± 171,903 (2.14%) ***	14.91 ± 0.99 (0.18%)
Window-Level Dynamic Global–Local Hybrid	164.16 ± 14.20 (1.74%) ***	526,743 ± 168,630 (5.11%) ***	14.67 ± 0.92 (1.76%) ***
Window-Cluster-Level Dynamic Global–Local Hybrid	166.48 ± 14.30 (0.35%) ***	542,568 ± 171,563 (2.26%) ***	14.88 ± 0.97 (0.35%) ***

Third, cluster-specific training does not consistently improve performance, whether the strategies consider only inter-user heterogeneity or also temporal dynamics. One possible explanation is that clusters formed according to load-profile similarity are not necessarily fully aligned with prediction-relevant differences. We further examine this issue through the empirical bias–variance analysis in Section 5.5.

Finally, the relative performance of forecasting strategies varies across datasets. The Window-Level Dynamic Global–Local Hybrid is among the strongest strategies on D1 and D2, but does not achieve the best result on D3. Since the three datasets differ in user populations, regions, and seasonal coverage, this finding suggests that the effectiveness of a forecasting strategy may depend on the specific application context. Understanding how these factors influence strategy selection is therefore an important direction for future research.

5.3. Inter-User Heterogeneity

Among the inter-user-heterogeneity-aware strategies, different implementations lead to substantially different levels of forecasting performance. This suggests that inter-user heterogeneity is useful for individual load forecasting, but its value depends on how it is incorporated into the forecasting pipeline, including whether adaptation is performed at the user or cluster level and whether models are trained globally or on cluster-specific subsets.

This subsection therefore takes a closer look at these strategies to understand how inter-user heterogeneity affects model suitability and forecasting performance, and to identify how this characteristic can be more effectively exploited in practical forecasting systems.

Table 4. Overall forecasting performance on Dataset D3. Values are reported as mean ± standard deviation. Values in parentheses indicate relative improvements over the best single global model for each metric. *** denotes statistical significance at the 1% level based on two-sided paired *t*-tests.

Strategy	MAE	MSE	WAPE(%)
LightGBM	405.10 ± 15.06	1,172,214 ± 223,531	9.88 ± 0.17
XGBoost	407.42 ± 15.08	1,175,926 ± 214,153	9.93 ± 0.19
N-BEATS	409.71 ± 15.99	1,220,512 ± 256,810	9.99 ± 0.19
LSTM	409.63 ± 16.08	1,186,922 ± 234,988	9.99 ± 0.17
Transformer	422.98 ± 17.34	1,311,224 ± 257,551	10.31 ± 0.24
PatchTST	410.55 ± 16.40	1,259,209 ± 299,984	10.01 ± 0.16
Global Training, User-Level Combination	398.92 ± 15.05 (1.53%) ***	1,159,424 ± 238,673 (1.09%) ***	9.72 ± 0.17 (1.53%) ***
Global Training, Cluster-Level Combination	398.96 ± 15.07 (1.51%) ***	1,150,957 ± 231,122 (1.81%) ***	9.73 ± 0.18 (1.51%) ***
Cluster-Specific Training & Cluster-Level Combination	404.05 ± 15.53 (0.26%)	1,197,641 ± 264,299 (−2.17%) ***	9.85 ± 0.17 (0.26%)
User-Level Static Global–Local Hybrid	398.76 ± 15.17 (1.56%) ***	1,155,454 ± 236,458 (1.43%) ***	9.72 ± 0.21 (1.57%) ***
Cluster-Level Static Global–Local Hybrid	399.36 ± 15.14 (1.42%) ***	1,161,228 ± 242,206 (0.94%) ***	9.74 ± 0.19 (1.42%) ***
Global Training, Window-Level Combination	398.92 ± 15.06 (1.52%) ***	1,146,994 ± 235,153 (2.15%) ***	9.72 ± 0.17 (1.53%) ***
Global Training, Window-Cluster-Level Combination	399.07 ± 15.13 (1.49%) ***	1,143,785 ± 234,129 (2.43%) ***	9.73 ± 0.20 (1.49%) ***
Window-Cluster-Specific Training & Combination	401.94 ± 15.56 (0.78%) ***	1,175,656 ± 246,771 (−0.29%)	9.80 ± 0.16 (0.78%) ***
Window-Level Dynamic Global–Local Hybrid	398.57 ± 15.07 (1.61%) ***	1,149,013 ± 236,007 (1.98%) ***	9.72 ± 0.17 (1.61%) ***
Window-Cluster-Level Dynamic Global–Local Hybrid	399.02 ± 15.12 (1.50%) ***	1,156,477 ± 237,034 (1.34%) ***	9.73 ± 0.18 (1.50%) ***

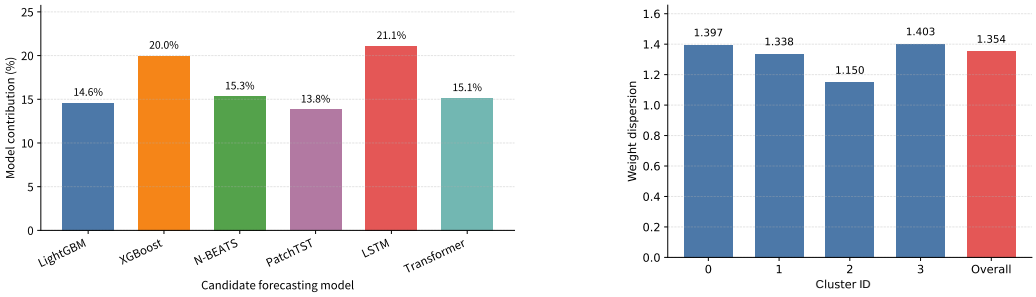
To understand how inter-user heterogeneity contributes to forecasting improvement, we examine the model combinations learned for individual users. Specifically, we aggregate the user-level weights assigned to the candidate models, as shown in Figure 3. The contributions are not concentrated on a single dominant model; instead, all candidate models make non-negligible contributions. This suggests that users differ in their preferred model combinations, and that such user-specific model suitability is an important source of the observed forecasting gains.

The results further show that inter-user heterogeneity is more effectively captured at the user-level than at the cluster level. As shown in Figure 3, user-level combination weights still vary substantially within several static clusters. This indicates that load-profile similarity does not necessarily align with forecasting model suitability, limiting the effectiveness of cluster-level adaptation and cluster-specific training. From a practical perspective, this finding highlights the importance of clustering quality, as poorly aligned clusters can directly reduce the forecasting gains.

5.4. Temporal Dynamics

This subsection further examines how temporal dynamics are captured by the dynamic forecasting strategies. Rather than assuming a fixed model-combination rule for each user, the temporal-dynamics-aware strategies allow model usage to vary across forecasting windows. We therefore focus on whether the learned strategy produces different model-

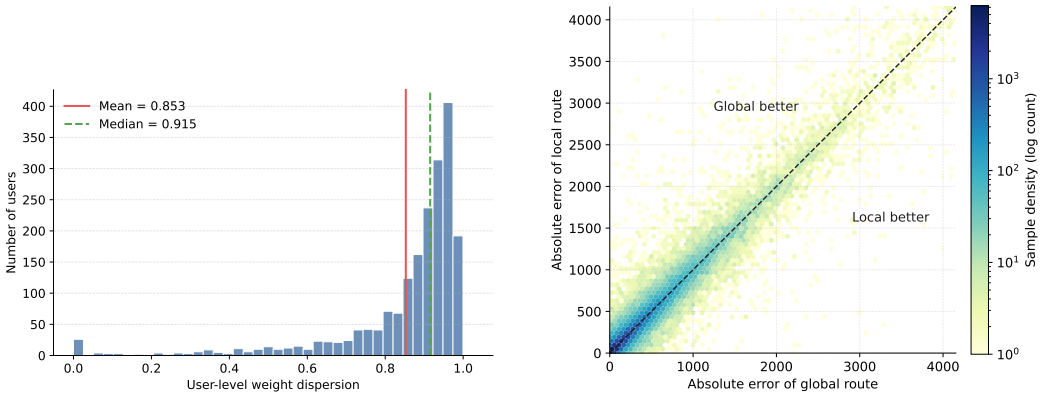
combination patterns for different historical load states and whether the global and local routes provide complementary information across windows.



(**a**) Model contribution distribution (**b**) Within-cluster weight dispersion

Figure 3. Panel (**a**) shows the contribution of each candidate model in the user-level combination strategy. Panel (**b**) shows the within-cluster dispersion of user-level model combination weights, where the overall bar denotes the dispersion computed across all users.

To examine the variation in model usage over time, we compute the dispersion of window-level model-combination weights for each user. Specifically, for each user, we compare the weight vector learned for each forecasting window with the user's average weight vector across all test windows. A larger dispersion value indicates that the same user relies on more different model combinations over time. As shown in Figure 4a, the dispersion values are not concentrated near zero, suggesting that the learned strategy does not assign a nearly fixed model-combination rule to each user. Instead, model usage varies across forecasting windows, indicating that the current historical load state contains useful information for determining model suitability.



(**a**) User-level weight dispersion (**b**) Global–local error comparison

Figure 4. Analysis of temporal dynamics and global–local complementarity. Panel (**a**) shows the distribution of user-level model-weight dispersion across forecasting windows, where a larger value indicates stronger variation in the model combination used by the same user over time. Panel (**b**) compares the absolute errors of the global route and the local route on test forecasting windows.

We further examine the relationship between the global route and the local route in the dynamic global–local hybrid strategy. Figure 4b compares their absolute errors on test forecasting windows. Points below the diagonal correspond to windows where the local route performs better, whereas points above the diagonal correspond to windows where the global route performs better. The distribution shows that neither route uniformly dominates the other. Instead, both routes have windows in which they produce smaller errors. This pattern indicates that the local route provides conditional complementary

information: it may not be reliable enough to replace the global route, but it can be useful for certain temporal states when combined through a dynamic routing mechanism.

5.5. Empirical Bias–Variance Analysis

As discussed in Section 3.4, the effects of different forecasting strategies can be understood through their trade-off between bias and variance. To further investigate the mechanisms underlying the observed performance differences, we conduct a bootstrap-based empirical analysis of these two components based on dataset D1. This analysis provides additional insights into how training scope and adaptation granularity affect forecasting accuracy and stability, thereby offering more practical guidance for forecasting strategy design.

For the inter-user-heterogeneity-aware strategies, we conduct a user-level bootstrap analysis while keeping the validation and test sets fixed. In each bootstrap run, users are sampled with replacement from the training population, and the complete strategy-learning procedure is repeated, including user clustering, global and cluster-specific model training, and validation-based model combination. We conduct 30 bootstrap runs, yielding 30 predictions for each fixed test instance under different resampled training sets. We then estimate the empirical squared-bias proxy from the deviation between the average bootstrap prediction and the observed target, and estimate the variance from the dispersion of predictions around their bootstrap mean. Table 5 reports the square roots of the empirical squared-bias proxy and variance.

Table 5. Empirical bias–variance analysis of the inter-user-heterogeneity-aware strategies on D1. The best-performing results are shown in bold.

Strategy	$\sqrt{\widehat{\mathbf{Bias}}^2}$	$\sqrt{\widehat{\mathbf{Var}}}$
Global Training, User-Level Combination	1205.12	168.95
Global Training, Cluster-Level Combination	1228.63	**140.47**
Cluster-Specific Training & Cluster-Level Combination	1265.21	251.98
User-Level Static Global–Local Hybrid	**1177.51**	263.79
Cluster-Level Static Global–Local Hybrid	1197.57	202.36

The results reveal two important findings from the perspectives of model training and combination. From the training perspective, Cluster-Specific Training leads to a substantial increase in variance, which is expected because each local model is trained on a smaller subset of users. However, as reflected in its weaker overall performance in Section 5.2, this strategy does not obtain sufficient bias reduction to compensate for the loss of stability. This finding reinforces our earlier observation that exploiting inter-user heterogeneity through specialized training requires the clustering or classification procedure to be well aligned with differences in forecasting model suitability. From the model-combination perspective, the bias–variance trade-off is particularly clear. In both the comparison between Global Training, User-Level Combination and Global Training, Cluster-Level Combination, and the comparison between the user-level and cluster-level static global–local hybrids, cluster-level combination consistently produces lower variance but higher bias than the corresponding finer-grained user-level approach. This result is consistent with the theoretical analysis: sharing a common combination rule across a broader group improves estimation stability, but also smooths out user-specific model preferences. These findings highlight the importance of carefully selecting the granularity of model combination to balance bias reduction and variance control.

For the temporal-dynamics-aware strategies, we conduct a time-block bootstrap analysis while keeping the validation and test sets fixed. Specifically, the training period is

divided into consecutive time blocks, which are sampled with replacement to construct 30 resampled training sets while preserving the temporal structure within each block. For each resampled training set, we repeat the complete strategy-learning procedure, including window clustering, global and window-cluster-specific model training, and validation-based model combination. This also yields 30 predictions for each fixed test instance, from which the empirical squared-bias proxy and variance are estimated using the same procedure described above. Table 6 reports the corresponding results.

Table 6. Empirical bias–variance analysis of the temporal-dynamics-aware strategies on D1. The best-performing results are shown in bold.

Strategy	$\sqrt{\widehat{\text{Bias}}^2}$	$\sqrt{\widehat{\text{Var}}}$
Global Training, Window-Level Combination	1192.03	353.82
Global Training, Window-Cluster-Level Combination	1196.32	295.80
Window-Cluster-Specific Training & Combination	1206.84	293.90
Window-Level Dynamic Global–Local Hybrid	**1189.24**	**278.65**
Window-Cluster-Level Dynamic Global–Local Hybrid	1199.15	338.88

These strategies exhibit two findings similar to those observed in the inter-user setting. Window-Cluster-Specific Training does not achieve the expected bias reduction relative to Global Training, Window-Cluster-Level Combination, indicating again that specialized training is beneficial only when the clustering structure is well aligned with prediction-relevant differences. At the same time, the comparison between window-level and window-cluster-level combination reveals a clear bias–variance trade-off. Another notable finding is that the Window-Level Dynamic Global–Local Hybrid achieves both the lowest bias and the lowest variance among the five strategies. This indicates that dynamically combining global and local predictors at the individual-window level can simultaneously improve adaptation and maintain stability. In contrast, the window-cluster-level dynamic hybrid relies on separate combination functions trained on smaller cluster-specific samples, resulting in higher bias and variance.

6. Conclusions

This study examines day-ahead individual load forecasting for commercial users from a strategy-level perspective. Rather than focusing only on the design of a single forecasting model, we investigate how forecasting models should be trained, combined, and used when individual load data exhibit inter-user heterogeneity and temporal dynamics. We define forecasting strategies based on their training component and prediction component, and compare homogeneous global models, inter-user-heterogeneity-aware strategies, and temporal-dynamics-aware strategies under the same forecasting task.

The empirical results show that forecasting strategy design has a substantial impact on prediction performance. Among the single global models, the performance differences are relatively small, and no model class consistently dominates all evaluation metrics, suggesting that each model class has its own strengths and inductive biases. More importantly, when user-level combination is adopted, almost all candidate models contribute to the final predictions, which improves forecasting performance by leveraging their complementary strengths across users. An interesting finding is that cluster-based strategies are less effective. This suggests that unsupervised load-profile clustering does not necessarily align with prediction-oriented model suitability. As a result, strategies that rely on static user clusters, especially cluster-specific training, provide limited or even negative gains.

Temporal-dynamics-aware strategies provide stronger and more consistent improvements, showing that temporal dynamics bring additional value beyond inter-user hetero-

geneity. This suggests that, in addition to differences across users, the time-varying load patterns of the same user should also be considered in forecasting strategy design. Notably, strategies that adapt model usage at the forecasting-window level already achieve strong performance, indicating that the current historical load state contains useful information for model combination. A similar pattern to the inter-user setting is also observed: window-cluster-level strategies are generally less effective than window-level strategies, suggesting that coarse cluster-based representations may still lose part of the prediction-relevant variation across windows. Interestingly, although window-cluster-specific local models perform poorly when used alone, they become useful when adaptively combined with globally trained models. This indicates that local models may not be reliable as standalone predictors, but can provide complementary information for certain forecasting windows within a dynamic global–local hybrid strategy.

Overall, these findings provide two main implications for individual load forecasting. First, although forecasting model architecture remains important, the design of forecasting strategies is equally critical. How models are trained, combined, and used can substantially affect prediction performance, even when the same candidate model pool is used. Second, effective strategy design should explicitly account for both inter-user heterogeneity and temporal dynamics. User-level differences determine which models or model combinations are suitable for different users, while temporal dynamics indicate that the suitable model usage may also change across forecasting windows for the same user. Therefore, practical commercial-user load forecasting systems should not only pursue stronger standalone models, but also carefully design strategies for data sharing, model specialization, and dynamic model usage. Our empirical results provide evidence that such strategy-level design can improve forecasting performance and offer useful guidance for real-world user-side energy management applications.

This study also has several limitations that suggest directions for future research. First, the datasets cover only selected regions and time periods and primarily consist of commercial and industrial users. Therefore, the applicability of the empirical findings to residential users, special-load users, and other electricity-consumption contexts should be examined carefully through broader validation. Second, the current experiments focus on historical load observations to isolate the effect of forecasting strategy design. An important extension is to incorporate weather conditions, calendar information, electricity prices, and other exogenous variables into the candidate forecasting models. Although the proposed strategy-level framework remains applicable under these richer input settings, further experiments are needed to determine whether the relative performance of different strategies remains consistent. Moreover, the effects of exogenous variables on electricity consumption may themselves vary across users and over time, making their interaction with inter-user heterogeneity and temporal dynamics a particularly meaningful direction for future research. Another promising direction is to move beyond forecasting performance alone and assess the practical value of the proposed strategies through downstream tasks, such as energy management, electricity trading, and other operational decision-making problems.

Author Contributions: Conceptualization, L.X.; methodology, H.Q., G.G.G., T.X. and Y.X.; software, G.G.G. and Y.H.O.; validation, G.G.G.; formal analysis, G.G.G.; investigation, H.Q. and T.X.; data curation, Y.H.O. and Y.X.; writing—original draft preparation, H.Q. and Y.X.; writing—review and editing, H.Q.; visualization, G.G.G.; supervision, L.X.; project administration, L.X.; funding acquisition, Y.H.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 72342012, and the National Social Science Foundation of China, grant number 25AZD026.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Acknowledgments: During the preparation of this manuscript, the authors used ChatGPT-5.5 for the purposes of grammar correction, language polishing, and improving the readability of author-written text. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: Author Yang Helen Ouyang is employed by Sichuan Energy Investment Wind Power Development Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A. Derivation of the Bias–Variance Decomposition

This appendix provides the derivation of the bias–variance decomposition used in Section 3.4. The derivation is based on the squared loss, which is consistent with the training objective used in this study.

Appendix A.1. Overall Risk Under Segment-Level Distributions

Let a forecasting instance be represented by $(\mathbf{x}_{i,t}, y_{i,t+1})$, where $\mathbf{x}_{i,t}$ is the historical load input and $y_{i,t+1}$ is the next-day load. For notational simplicity, we denote

$$\xi = (\mathbf{x}, i, t),$$

where ξ collects the information used by a forecasting strategy to produce a prediction: $\mathbf{x}$ denotes the model input, while i and t may be used for strategy-level design. The target value is denoted by y.

Due to inter-user heterogeneity and temporal dynamics, the overall data distribution can be viewed as a mixture of segment-level distributions:

$$P = \sum_{g \in \mathcal{G}} p_g P_g, \tag{A1}$$

where $g \in \mathcal{G}$ denotes a segment, P_g is the distribution of forecasting instances in segment g, and p_g is the proportion of this segment in the overall distribution.

Given a forecasting strategy $\mathcal{S}$, applying it to the available training data induces an effective prediction system. To make the dependence on training data explicit, we write this predictor as $\hat{h}^{\mathcal{S}}_{\mathcal{D}}$ where $\mathcal{D}$ denotes the data used for model training and, when applicable, model selection or prediction combination. For an input $\xi = (\mathbf{x}, i, t)$, the prediction is

$$\hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi).$$

The expected prediction risk of strategy $\mathcal{S}$ over the overall distribution is

$$\mathcal{R}(\hat{h}^{\mathcal{S}}) = \mathbb{E}_{\mathcal{D}} \mathbb{E}_{(\xi, y) \sim P} \left[\left(y - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi) \right)^2 \right]. \tag{A2}$$

Using the mixture representation in Equation (A1), this overall risk can be written as the weighted sum of segment-level risks:

$$\mathcal{R}(\hat{h}^{\mathcal{S}}) = \sum_{g \in \mathcal{G}} p_g \mathbb{E}_{\mathcal{D}} \left[\mathcal{R}_g(\hat{h}^{\mathcal{S}}_{\mathcal{D}}) \right], \tag{A3}$$

where

$$\mathcal{R}_g(\hat{h}^{\mathcal{S}}_{\mathcal{D}}) = \mathbb{E}_{(\xi,y)\sim P_g}\left[\left(y - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)\right)^2\right]. \tag{A4}$$

Appendix A.2. Segment-Level Bias–Variance Decomposition

We now derive the decomposition for a fixed segment g. Under squared loss, define the segment-specific optimal predictor as

$$h^*_g(\xi) = \mathbb{E}_{P_g}[y \mid \xi]. \tag{A5}$$

This is the optimal prediction function for segment g under squared loss.

For any forecasting instance from segment g, the target value can be written as

$$y = h^*_g(\xi) + \varepsilon_g, \tag{A6}$$

where ε_g is the unpredictable noise satisfying

$$\mathbb{E}_{P_g}[\varepsilon_g \mid \xi] = 0. \tag{A7}$$

For a fixed input ξ, define the average predictor induced by strategy $\mathcal{S}$ over possible training datasets as

$$\bar{h}^{\mathcal{S}}_g(\xi) = \mathbb{E}_{\mathcal{D}}\left[\hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)\right]. \tag{A8}$$

Then the prediction error can be decomposed as

$$y - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi) = \left(y - h^*_g(\xi)\right) + \left(h^*_g(\xi) - \bar{h}^{\mathcal{S}}_g(\xi)\right) + \left(\bar{h}^{\mathcal{S}}_g(\xi) - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)\right). \tag{A9}$$

For simplicity, define

$$a = y - h^*_g(\xi), \qquad b = h^*_g(\xi) - \bar{h}^{\mathcal{S}}_g(\xi), \qquad c = \bar{h}^{\mathcal{S}}_g(\xi) - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi).$$

Then

$$\left(y - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)\right)^2 = (a + b + c)^2. \tag{A10}$$

Expanding the square gives

$$(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2ac + 2bc. \tag{A11}$$

We take expectations over y and $\mathcal{D}$ conditional on ξ. The cross terms vanish. First, because $h^*_g(\xi) = \mathbb{E}_{P_g}[y \mid \xi]$, we have

$$\mathbb{E}_{P_g}[a \mid \xi] = \mathbb{E}_{P_g}[y - h^*_g(\xi) \mid \xi] = 0. \tag{A12}$$

Therefore,

$$\mathbb{E}[2ab \mid \xi] = 2b\mathbb{E}_{P_g}[a \mid \xi] = 0. \tag{A13}$$

Second, by the definition of $\bar{h}^{\mathcal{S}}_g(\xi)$, we have

$$\mathbb{E}_{\mathcal{D}}[c \mid \xi] = \mathbb{E}_{\mathcal{D}}[\bar{h}^{\mathcal{S}}_g(\xi) - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)] = 0. \tag{A14}$$

Thus,

$$\mathbb{E}[2bc \mid \xi] = 2b\mathbb{E}_{\mathcal{D}}[c \mid \xi] = 0. \tag{A15}$$

Finally, since the evaluation noise is independent of the training process, and since both $\mathbb{E}_{P_g}[a \mid \xi] = 0$ and $\mathbb{E}_{\mathcal{D}}[c \mid \xi] = 0$, we have

$$\mathbb{E}[2ac \mid \xi] = 0. \tag{A16}$$

Combining Equations (A11)–(A16), for a fixed input ξ we obtain

$$\begin{aligned}&\mathbb{E}_{y,\mathcal{D}}\left[\left(y - \hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)\right)^2 \mid \xi\right] \\ &= \mathbb{E}_y\left[\left(y - h^*_g(\xi)\right)^2 \mid \xi\right] + \left(h^*_g(\xi) - \bar{h}^{\mathcal{S}}_g(\xi)\right)^2 + \mathbb{E}_{\mathcal{D}}\left[\left(\hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi) - \bar{h}^{\mathcal{S}}_g(\xi)\right)^2\right].\end{aligned} \tag{A17}$$

Taking expectation over $\xi \sim P_g$ yields

$$\mathbb{E}_{\mathcal{D}}\left[\mathcal{R}_g(\hat{h}^{\mathcal{S}}_{\mathcal{D}})\right] = \sigma_g^2 + B_g^2(\hat{h}^{\mathcal{S}}) + V_g(\hat{h}^{\mathcal{S}}), \tag{A18}$$

where

$$\sigma_g^2 = \mathbb{E}_{(\xi,y)\sim P_g}\left[\left(y - h^*_g(\xi)\right)^2\right] \tag{A19}$$

is the irreducible error in segment g,

$$B_g^2(\hat{h}^{\mathcal{S}}) = \mathbb{E}_{\xi\sim P_g}\left[\left(\mathbb{E}_{\mathcal{D}}[\hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)] - h^*_g(\xi)\right)^2\right] \tag{A20}$$

is the squared bias of the effective prediction system induced by strategy $\mathcal{S}$ in segment g, and

$$V_g(\hat{h}^{\mathcal{S}}) = \mathbb{E}_{\xi\sim P_g}\left[\mathrm{Var}_{\mathcal{D}}\left(\hat{h}^{\mathcal{S}}_{\mathcal{D}}(\xi)\right)\right] \tag{A21}$$

is the variance induced by the training process.

Appendix A.3. Overall Decomposition

Substituting the segment-level decomposition in Equation (A18) into the weighted segment risk in Equation (A3), we obtain

$$\begin{aligned}\mathcal{R}(\hat{h}^{\mathcal{S}}) &= \sum_{g\in\mathcal{G}} p_g \mathbb{E}_{\mathcal{D}}\left[\mathcal{R}_g(\hat{h}^{\mathcal{S}}_{\mathcal{D}})\right] \\ &= \sum_{g\in\mathcal{G}} p_g \sigma_g^2 + \sum_{g\in\mathcal{G}} p_g B_g^2(\hat{h}^{\mathcal{S}}) + \sum_{g\in\mathcal{G}} p_g V_g(\hat{h}^{\mathcal{S}}).\end{aligned} \tag{A22}$$

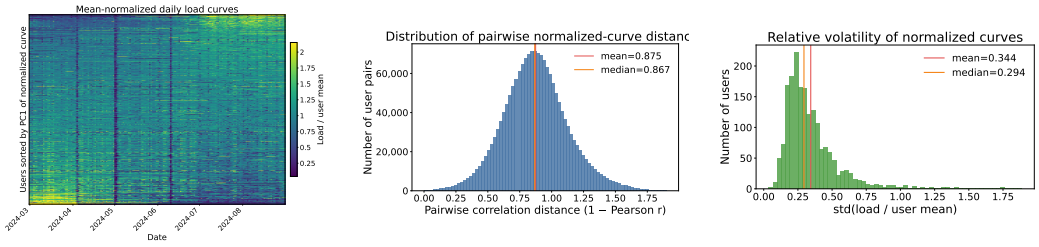
Equation (A22) shows that, under squared loss, the overall expected prediction risk can be decomposed into three parts: the weighted irreducible error, the weighted squared bias, and the weighted variance. The irreducible error is determined by the data-generating process and is independent of the forecasting strategy. Different forecasting strategies affect the overall risk mainly by changing the effective prediction system $\hat{h}^{\mathcal{S}}$, and therefore by changing the weighted bias and variance terms.

Appendix B. Inter-User Heterogeneity and Temporal Dynamics

This appendix provides additional analyses of Dataset D1 to examine the two key characteristics considered in this study, namely inter-user heterogeneity and temporal dynamics, and thereby provide empirical support for our research motivation.

Appendix B.1. Inter-User Heterogeneity

To examine inter-user heterogeneity in Dataset D1, we analyze differences in load patterns and variability across users. Figure A1 summarizes the results from three complementary perspectives.



(**a**) Normalized daily load profiles (**b**) Pairwise correlation distances (**c**) User-level relative volatility

Figure A1. Analysis of inter-user heterogeneity in Dataset D1. Panel (**a**) visualizes normalized user load series, Panel (**b**) quantifies pairwise differences in load shapes, and Panel (**c**) shows differences in consumption variability across users.

Figure A1a provides a direct visualization of the differences in daily load patterns across users. For each user i, we normalize the daily load series by the user's mean load over the observation period:

$$\tilde{y}_{i,t} = \frac{y_{i,t}}{\bar{y}_i}, \qquad \bar{y}_i = \frac{1}{T_i} \sum_{t=1}^{T_i} y_{i,t}.$$

Each row of the heatmap represents one user, each column represents one day, and the color indicates the normalized load level. A value greater than one means that the user's load on that day is higher than its own average level, whereas a value below one indicates a lower-than-average load. This normalization removes the large differences in absolute consumption scale and allows the comparison to focus on the relative load patterns of different users. As shown in the figure, the color sequences differ substantially across rows: some users exhibit relatively stable and regular patterns, whereas others show stronger fluctuations, abrupt changes, or different periods of high and low consumption. Therefore, users display clearly different temporal load patterns, providing direct visual evidence of inter-user heterogeneity.

Figure A1b quantitatively measures the differences in load-pattern shapes across users. Using the normalized load series introduced above, we calculate the Pearson correlation coefficient for every pair of users and define their correlation distance as

$$d_{ij}^{\mathrm{corr}} = 1 - \mathrm{corr}(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j),$$

where a smaller value indicates more similar temporal variation, while a larger value indicates weaker or even opposite co-movement between the two users. The figure presents the distribution of correlation distances over all user pairs. The distribution is widely dispersed, with a mean of 0.875 and a median of 0.867; its 5th and 95th percentiles are 0.472 and 1.311, respectively. Thus, only a limited proportion of user pairs exhibit highly similar load patterns, whereas many pairs show substantial differences in their temporal variation. These results quantitatively confirm that the normalized load profiles are heterogeneous across users, rather than being explained solely by differences in consumption scale.

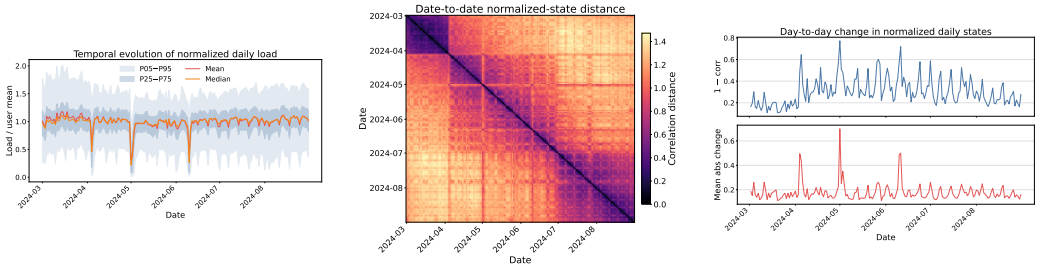
Figure A1c examines differences in load variability across users. For each user i, we calculate the relative volatility as

$$v_i = \frac{\mathrm{std}(\mathbf{y}_i)}{\mathrm{mean}(\mathbf{y}_i)},$$

where a larger value indicates that the user's load fluctuates more strongly relative to its average consumption level. Using this scale-normalized measure allows users with different absolute load levels to be compared on a consistent basis. As shown in the figure, the distribution is widely spread, with a median of 0.294 and 5th and 95th percentiles of 0.146 and 0.706, respectively. This means that some users exhibit relatively stable electricity consumption, whereas others experience fluctuations that are much larger relative to their average load. The result therefore provides additional evidence of inter-user heterogeneity, showing that users differ not only in the shape of their load patterns but also in the degree of load variability.

Appendix B.2. Temporal Dynamics

To examine temporal dynamics in Dataset D1, we analyze how the cross-user load state changes over time. Figure A2 presents the results from three complementary perspectives.



(**a**) Normalized load level over time (**b**) Correlation distances across dates (**c**) Adjacent-day changes over time

Figure A2. Analysis of temporal dynamics in Dataset D1. Panel (**a**) shows the evolution of the average normalized load level over time, Panel (**b**) visualizes the correlation-distance structure between different dates, and Panel (**c**) reports the magnitude of changes between consecutive dates.

Figure A2a shows the temporal evolution of the normalized load level. For each user, the daily load is divided by the user's mean load over the observation period. For each date t, we then calculate the average normalized load across all users:

$$\bar{y}_t^{\mathrm{norm}} = \frac{1}{N}\sum_{i=1}^{N}\frac{y_{i,t}}{\bar{y}_i}.$$

A value above one indicates that the overall load level on that date is higher than the users' average levels, whereas a value below one indicates a relatively low-load date. As shown in the figure, the normalized load level varies continuously over time, with the 5th and 95th percentiles equal to 0.877 and 1.098, respectively. This indicates that the overall consumption state is not constant across the observation period.

Figure A2b further examines whether the cross-user load configuration changes across dates. For each date t, we construct a normalized load-state vector across all users:

$$\mathbf{z}_t = \left(\frac{y_{1,t}}{\bar{y}_1}, \frac{y_{2,t}}{\bar{y}_2}, \ldots, \frac{y_{N,t}}{\bar{y}_N}\right).$$

We then calculate the correlation distance between every pair of dates:

$$d^{\text{corr}}_{t,t'} = 1 - \text{corr}(\mathbf{z}_t, \mathbf{z}_{t'}).$$

A smaller value indicates that users exhibit similar relative load states on the two dates, while a larger value indicates a stronger change in the cross-user load configuration. The heatmap shows substantial variation in the similarities among different dates. The pairwise distances have a mean of 1.003 and a median of 1.067, with the 5th and 95th percentiles equal to 0.444 and 1.351, respectively. These results suggest that different dates can correspond to substantially different load states rather than following one fixed temporal pattern.

Figure A2c focuses on short-term changes by comparing consecutive dates. For each adjacent date pair $(t-1, t)$, we calculate the correlation distance between their normalized load-state vectors:

$$\Delta^{\text{corr}}_t = 1 - \text{corr}(\mathbf{z}_{t-1}, \mathbf{z}_t).$$

A larger value represents a stronger change in the relative load configuration from one day to the next. The adjacent-day distances have a mean of 0.303 and a median of 0.277, with the 5th and 95th percentiles equal to 0.149 and 0.556, respectively. The magnitude of change also varies noticeably over time, indicating that some periods are relatively stable whereas others experience more pronounced transitions.

Overall, these results show that Dataset D1 exhibits temporal dynamics at both aggregate and cross-user distributional levels. The overall load level changes over time, the relative load configuration differs across dates, and the strength of short-term transitions is itself time-varying. This provides empirical support for treating different forecasting windows as potentially distinct temporal load states.

Appendix C. Implementation Details

All hyperparameters are selected using the validation set, and the test set is not involved in model selection. For each dataset and forecasting strategy, grid search is conducted separately within a predefined search space, and the configuration with the lowest validation MSE is used for final evaluation. When a strategy trains forecasting models on specialized subsets, such as user clusters or window clusters, the corresponding model hyperparameters are reselected using the validation samples associated with that training scope. For neural forecasting models and MLP-based combination modules, early stopping is applied according to validation performance. The number of clusters used in the static user-clustering and dynamic window-clustering procedures is determined through elbow analysis based only on the training data. The key model and strategy configurations used for Dataset D1 are reported in Tables A1 and A2.

Table A1. Key configurations of the candidate forecasting models for Dataset D1.

Model	Selected Configuration
LightGBM	`num_leaves` = 64, `learning_rate` = 0.05, `n_estimators` = 2000, `feature_fraction` = 0.8, `bagging_fraction` = 0.8, `bagging_freq` = 3, `lambda_l1` = 1.0, `lambda_l2` = 1.0, `early_stopping_rounds` = 100
XGBoost	`max_depth` = 12, `learning_rate` = 0.02, `n_estimators` = 1600, `subsample` = 1.0, `colsample_bytree` = 0.8, `min_child_weight` = 8, `reg_lambda` = 3.0, `max_bin` = 512
N-BEATS	input length = 14, forecast length = 1, hidden units = 512, two blocks per stack, stack types = trend, seasonality, and generic, `thetas_dim` = [4, 8, 16], batch size = 2048, maximum epochs = 180, patience = 18

Table A1. *Cont.*

Model	Selected Configuration
LSTM	input length = 14, hidden units = 128, two recurrent layers, linear prediction head, batch size = 2048, maximum epochs = 220, patience = 20
Transformer	input length = 14, $d_{\text{model}} = 128$, four attention heads, two encoder layers, dropout = 0.1, last-step pooling, batch size = 2048, maximum epochs = 180, patience = 16
PatchTST	input length = 14, forecast length = 1, three encoder layers, four attention heads, $d_{\text{model}} = 128$, $d_{\text{ff}} = 256$, patch length = 14, stride = 7, RevIN enabled, batch size = 2048, maximum epochs = 80, patience = 8

Table A2. Key configurations of the strategy-level components for Dataset D1.

Component	Selected Configuration
Static user clustering	Monday–Sunday mean load profile constructed from the training period; user-level z-normalization; DTW distance; k-medoids; four clusters selected through elbow analysis
Dynamic window clustering	Z-normalized 14-day historical windows; approximate DTW-based k-medoids; four clusters selected through elbow analysis; `sample_size` = 30,000, `medoid_candidates` = 256, `max_iters` = 8
Window-level global combination	One shared MLP: $\text{Linear}(d_{\text{in}}, 16)$, LayerNorm, SiLU, $\text{Linear}(16, 6)$, Softmax
Window-cluster-level global combination	One shared cluster-aware MLP: $\text{Linear}(d_{\text{in}}, 32)$, LayerNorm, SiLU, $\text{Linear}(32, 6)$, Softmax
Window-cluster-specific local combination	One MLP for each window cluster: $\text{Linear}(d_{\text{in}}, 12)$, LayerNorm, SiLU, $\text{Linear}(12, 6)$, Softmax
Window-level dynamic global–local hybrid	One shared MLP: $\text{Linear}(d_{\text{in}}, 80)$, LayerNorm, SiLU, $\text{Linear}(80, 12)$, Softmax
Window-cluster-level dynamic global–local hybrid	One MLP for each window cluster: $\text{Linear}(d_{\text{in}}, 16)$, LayerNorm, SiLU, $\text{Linear}(16, 12)$, Softmax

Appendix D. Computational Efficiency Analysis

This appendix provides a concise analysis of the computational requirements of the compared forecasting approaches. We first compare their structural requirements in terms of the number of forecasting experts, clustering operations, and model aggregation. We then report the measured offline construction time of the ten forecasting strategies on Dataset D1 to provide a practical reference for their implementation costs.

The experiments were conducted on a workstation equipped with an Intel Core Ultra 9 285K CPU, an NVIDIA GeForce RTX 5090 GPU with 32 GB of GPU memory, and 93 GB of system memory. The software environment included Ubuntu 24.04.2, Python 3.9.23, PyTorch 2.8.0, LightGBM 4.6.0, XGBoost 2.1.4, and CUDA 12.8. The expert models were trained serially in the current implementation. The reported expert-construction time includes model fitting and the generation of validation and test predictions, and should therefore be interpreted as an offline construction cost rather than pure training time or online inference latency.

Table A3 shows that a single global model has the lowest computational requirement, whereas global-only combination strategies require K candidate experts and introduce an additional aggregation step. Cluster-specific strategies increase the number of trained and maintained experts linearly with the number of user or window clusters, resulting in KC or KS experts. The global–local hybrid strategies have the highest structural requirements because they maintain both global and local expert pools, requiring $K(1 + C)$ or $K(1 + S)$

experts. However, each forecasting instance is routed to only one relevant local cluster. Consequently, the model-aggregation complexity remains $\mathcal{O}(K)$ for global-only or local-only strategies and $\mathcal{O}(2K)$ for global–local hybrids. The main increase in computational requirements therefore arises from constructing and maintaining additional local expert pools rather than from the aggregation operation itself.

Table A4 provides a practical comparison of the offline construction costs on Dataset D1. The global-only combination strategies incur little additional cost because they reuse the same six global experts, and their total construction time remains close to the reference cost. The user-cluster-specific and window-cluster-specific strategies require additional local expert pools, increasing the offline cost to approximately 1.62 and 1.88 times the reference, respectively. The global–local hybrid strategies have the highest offline costs because they construct and maintain both global and local experts, resulting in approximately 2.62–2.74 times the reference cost.

Table A3. Structural computational requirements of the compared forecasting approaches.

Approach	Forecasting Experts	Clustering	Objects Used for Clustering	Model Aggregation
Single Global Model	1	×	–	–
Global Training, User-Level Combination	K	×	–	N user-specific weight vectors over K experts; $\mathcal{O}(K)$ per forecast
Global Training, Cluster-Level Combination	K	✓ (user)	N user profiles	C cluster-level weight vectors over K experts; $\mathcal{O}(K)$ per forecast
Cluster-Specific Training & Cluster-Level Combination	KC	✓ (user)	N user profiles	C cluster-level weight vectors over K local experts; $\mathcal{O}(K)$ per forecast
User-Level Static Global–Local Hybrid	$K(1+C)$	✓ (user)	N user profiles	N user-specific aggregators over $2K$ experts; $\mathcal{O}(2K)$ per forecast
Cluster-Level Static Global–Local Hybrid	$K(1+C)$	✓ (user)	N user profiles	C cluster-level aggregators over $2K$ experts; $\mathcal{O}(2K)$ per forecast
Global Training, Window-Level Combination	K	×	–	One shared MLP generating weights over K experts; $\mathcal{O}(K)$ per forecast
Global Training, Window-Cluster-Level Combination	K	✓ (window)	M_w historical windows	One cluster-aware MLP generating weights over K experts; $\mathcal{O}(K)$ per forecast
Window-Cluster-Specific Training & Combination	KS	✓ (window)	M_w historical windows	S cluster-specific MLPs over K local experts; $\mathcal{O}(K)$ per forecast
Window-Level Dynamic Global–Local Hybrid	$K(1+S)$	✓ (window)	M_w historical windows	One shared MLP generating weights over $2K$ experts; $\mathcal{O}(2K)$ per forecast
Window-Cluster-Level Dynamic Global–Local Hybrid	$K(1+S)$	✓ (window)	M_w historical windows	S cluster-specific MLPs over $2K$ experts; $\mathcal{O}(2K)$ per forecast

Notes: K denotes the number of candidate forecasting models; N denotes the number of users; C and S denote the numbers of static user clusters and dynamic window clusters, respectively; T_{tr} denotes the length of the training period; and L denotes the historical lookback length. The number of available training windows is approximately $N(T_{\text{tr}} - L)$, while $M_w \leq N(T_{\text{tr}} - L)$ denotes the number of windows used to fit the approximate dynamic-clustering model. For D1, $K = 6$, $N = 1855$, $C = S = 4$, and $M_w = 30{,}000$. Each static user-clustering sample is a user profile constructed from the user's training-period observations. The aggregation complexity describes the combination of expert outputs after the required expert predictions have been generated.

The time required for model aggregation is small relative to expert construction across all strategies. Even the most computationally demanding strategy completes its offline construction in less than 18 min under the current environment. Therefore, although local and global–local strategies introduce additional computation, their offline costs remain manageable for day-ahead load forecasting applications in which model construction and updating can be completed before operational deployment.

Table A4. Measured offline construction costs of the compared forecasting strategies on Dataset D1.

Strategy	Expert Time (s)	Combination Time (s)	Total Time (min)	Relative Cost
Global Training, User-Level Combination	380.979	4.836	6.43	1.013×
Global Training, Cluster-Level Combination	380.979	0.073	6.35	1.000×
Cluster-Specific Training & Cluster-Level Combination	617.028	0.076	10.29	1.620×
User-Level Static Global–Local Hybrid	998.007	0.656	16.64	2.621×
Cluster-Level Static Global–Local Hybrid	998.007	0.025	16.63	2.620×
Global Training, Window-Level Combination	380.979	2.628	6.39	1.007×
Global Training, Window-Cluster-Level Combination	380.979	0.534	6.36	1.001×
Window-Cluster-Specific Training & Combination	713.260	1.736	11.92	1.877×
Window-Level Dynamic Global–Local Hybrid	1043.107	0.568	17.39	2.739×
Window-Cluster-Level Dynamic Global–Local Hybrid	1043.107	1.809	17.42	2.743×

Notes: Expert time includes expert-model fitting and validation/test prediction generation. Combination time is measured using cached expert predictions and includes construction and evaluation of the corresponding aggregation module. The six-global-expert pool is used as the reference when calculating relative cost. A standalone global model is not included because its construction time depends on the selected model architecture.

References

1. Haqqi, M.; Benmamoun, Z.; Hachimi, H.; Raouf, Y.; Jebbor, I.; Akikiz, M. Renewable and Sustainable Energy: Solar Energy and Electrical System Design. In *Proceedings of the 2023 9th International Conference on Optimization and Applications (ICOA)*; IEEE: New York, NY, USA, 2023; pp. 1–6.
2. Haben, S.; Arora, S.; Giasemidis, G.; Voss, M.; Greetham, D.V. Review of low voltage load forecasting: Methods, applications, and recommendations. *Appl. Energy* **2021**, *304*, 117798. [CrossRef]
3. Wang, X.; Wang, H.; Bhandari, B.; Cheng, L. AI-empowered methods for smart energy consumption: A review of load forecasting, anomaly detection and demand response. *Int. J. Precis. Eng. Manuf.-Green Technol.* **2024**, *11*, 963–993.
4. Darwazeh, D.; Duquette, J.; Gunay, B.; Wilton, I.; Shillinglaw, S. Review of peak load management strategies in commercial buildings. *Sustain. Cities Soc.* **2022**, *77*, 103493. [CrossRef]
5. Hobbs, B.F.; Jitprapaikulsarn, S.; Konda, S.; Chankong, V.; Loparo, K.A.; Maratukulam, D.J. Analysis of the Value for Unit Commitment of Improved Load Forecasts. *IEEE Trans. Power Syst.* **1999**, *14*, 1342–1348. [CrossRef]
6. Zhang, J.; Wang, Y.; Hug, G. Cost-Oriented Load Forecasting. *Electr. Power Syst. Res.* **2022**, *205*, 107723. [CrossRef]
7. Malehmirchegini, L.; Farzaneh, H. Demand Response Modeling in a Day-Ahead Wholesale Electricity Market in Japan, Considering the Impact of Customer Risk Aversion and Dynamic Price Elasticity of Demand. *Energy Rep.* **2022**, *8*, 11910–11926. [CrossRef]
8. Alanazi, M.D.; Saeed, A.; Islam, M.; Habib, S.; Sherazi, H.I.; Khan, S.; Shees, M.M. Enhancing Short-Term Electrical Load Forecasting for Sustainable Energy Management in Low-Carbon Buildings. *Sustainability* **2023**, *15*, 16885. [CrossRef]
9. Ansari, G.; Keypour, R. Optimizing the Performance of Commercial Demand Response Aggregator Using the Risk-Averse Function of Information-Gap Decision Theory. *Sustainability* **2023**, *15*, 6243. [CrossRef]
10. Ramu, P.; Gangatharan, S.; Rangasamy, S.; Mihet-Popa, L. Categorization of Loads in Educational Institutions to Effectively Manage Peak Demand and Minimize Energy Cost Using an Intelligent Load Management Technique. *Sustainability* **2023**, *15*, 12209. [CrossRef]
11. Almohaimeed, S.A.; Suryanarayanan, S.; O'Neill, P. Simulation Studies to Quantify the Impact of Demand Side Management on Environmental Footprint. *Sustainability* **2021**, *13*, 9504. [CrossRef]
12. Eren, Y.; Küçükdemiral, İ. A comprehensive review on deep learning approaches for short-term load forecasting. *Renew. Sustain. Energy Rev.* **2024**, *189*, 114031. [CrossRef]
13. Li, Y.; Qin, D.; Poor, H.V.; Wang, Y. Introducing edge intelligence to smart meters via federated split learning. *Nat. Commun.* **2024**, *15*, 9044. [CrossRef] [PubMed]
14. Biswal, B.; Deb, S.; Datta, S.; Ustun, T.S.; Cali, U. Review on smart grid load forecasting for smart energy management using machine learning and deep learning techniques. *Energy Rep.* **2024**, *12*, 3654–3670. [CrossRef]
15. Jebbor, I.; Benmamoun, Z.; Hachimi, H. Forecasting supply chain disruptions in the textile industry using machine learning: A case study. *Ain Shams Eng. J.* **2024**, *15*, 103116. [CrossRef]
16. Gambardella, C.; Pahle, M. Time-varying electricity pricing and consumer heterogeneity: Welfare and distributional effects with variable renewable supply. *Energy Econ.* **2018**, *76*, 257–273. [CrossRef]
17. Zhou, Y.; Ma, R.; Su, Y.; Wu, L. Too big to change: How heterogeneous firms respond to time-of-use electricity price. *China Econ. Rev.* **2019**, *58*, 101342. [CrossRef]
18. Fekri, M.N.; Patel, H.; Grolinger, K.; Sharma, V. Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network. *Appl. Energy* **2021**, *282*, 116177. [CrossRef]

19. Botman, L.; Lago, J.; Becker, T.; Vanthournout, K.; De Moor, B. A global probabilistic approach for short-term forecasting of individual households electricity consumption. *Appl. Energy* **2025**, *382*, 125168. [CrossRef]
20. Qin, D.; Wu, X.; Sun, D.; Liang, Z.; Zhang, N. Load forecasting under distribution shift: An online quantile ensembling approach. *Appl. Energy* **2025**, *401*, 126812. [CrossRef]
21. Wu, H.; Elizaveta, D.; Zhadan, A.; Petrosian, O. Forecasting online adaptation methods for energy domain. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106499. [CrossRef]
22. Geman, S.; Bienenstock, E.; Doursat, R. Neural networks and the bias/variance dilemma. *Neural Comput.* **1992**, *4*, 1–58. [CrossRef]
23. Hastie, T. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: New York, NY, USA, 2009.
24. Rodrigues, F.; Cardeira, C.; Calado, J.M.; Melicio, R. Short-term load forecasting of electricity demand for the residential sector based on modelling techniques: A systematic review. *Energies* **2023**, *16*, 4098. [CrossRef]
25. Ibrahim, B.; Rabelo, L.; Gutierrez-Franco, E. Machine learning for short-term load forecasting in smart grids. *Energies* **2022**, *15*, 8079. [CrossRef]
26. Wen, X.; Liao, J.; Niu, Q.; Shen, N.; Bao, Y. Deep learning-driven hybrid model for short-term load forecasting and smart grid information management. *Sci. Rep.* **2024**, *14*, 13272. [CrossRef] [PubMed]
27. Zhao, P.; Hu, W.; Cao, D.; Dong, Z.; Zeng, Y. A review on deep learning-based electrical load forecasting: From perspectives of learning paradigms and foundation models. *J. Mod. Power Syst. Clean Energy* **2026**, *14*, 791–809. [CrossRef]
28. Chen, G.; Lu, S.; Zhou, S.; Tian, Z.; Kim, M.K.; Liu, J.; Liu, X. A systematic review of building energy consumption prediction: From perspectives of load classification, data-driven frameworks, and future directions. *Appl. Sci.* **2025**, *15*, 3086. [CrossRef]
29. Wu, Z.; Qi, M.; Zhang, W.; Zhang, X.; Yang, Q.; Zhao, W. The electricity load prediction model for residential buildings: A critical review of output types, prediction methods and driving factors. *Buildings* **2025**, *15*, 925. [CrossRef]
30. Kallel, S.; Amayri, M.; Bouguila, N. Clustering and interpretability of residential electricity demand profiles. *Sensors* **2025**, *25*, 2026. [CrossRef] [PubMed]
31. Quesada, C.; Montero-Manso, P.; Pflugradt, N. A data-driven methodology for deriving electricity consumption typologies from smart meters. *Energy Rep.* **2025**, *11*, 1050–1065. [CrossRef]
32. Fesseha, S.; Li, B.; Qi, B.; Chen, S.; Gong, F. A comparative study of multi-step forecasting algorithms for real-time load profiling in industrial electricity demand. *IEEE Access* **2026**, *14*, 50074–50088. [CrossRef]
33. Liu, Y.; Yang, M. Ultra-short-term photovoltaic cluster power prediction based on photovoltaic cluster dynamic clustering and spatiotemporal heterogeneous dynamic modeling. *Electronics* **2025**, *14*, 3641. [CrossRef]
34. Dong, X.; Yu, Y.; Jin, H.; Hu, Z.; Bao, J. Short-term load forecasting in price-volatile markets: A pattern-clustering and adaptive modeling approach. *Processes* **2025**, *14*, 5. [CrossRef]
35. Cheng, Y.; Zhang, T.; Lv, S.; Ma, F.; Fan, M. Spatial-temporal electric load portrait method for multi-microgrids based on clustering-granulation-clustering. *Knowl.-Based Syst.* **2025**, *312*, 112345.
36. Abdallah, M.; Rossi, R.A.; Mahadik, K.; Kim, S.; Zhao, H.; Bagchi, S. Evaluation-free time-series forecasting model selection via meta-learning. *Acm Trans. Knowl. Discov. Data* **2025**, *19*, 1–41. [CrossRef]
37. Kuster, C.; Rezgui, Y.; Mourshed, M. Electrical load forecasting models: A critical systematic review. *Sustain. Cities Soc.* **2017**, *35*, 257–270. [CrossRef]
38. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3146–3154.
39. Chen, T.; Guestrin, C. XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
40. Oreshkin, B.N.; Carpov, D.; Chapados, N.; Bengio, Y. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *arXiv* **2019**, arXiv:1905.10437.
41. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
42. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
43. Nie, Y.; H. Nguyen, N.; Sinthong, P.; Kalagnanam, J. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In Proceedings of the International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.