Pattern Recognition 180 (2026) 114310

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

# DPIA: Dynamic Periodic Interaction and Adaptation for Multivariate Time Series forecasting

Cheng Tan, Wenbin Zhang, Mingjing Du *

*Jiangsu Key Laboratory of Educational Intelligent Technology, School of Artificial Intelligence and Computer Science, Jiangsu Normal University, Xuzhou, 221116, China*

ARTICLE INFO

*Keywords:*
Multivariate Time Series forecasting
Channel interactions
Mixture of experts
Adaptive period identification

ABSTRACT

Multivariate Time Series (MTS) forecasting is crucial for many critical domains; yet modeling complex temporal patterns while capturing dynamic inter-variable dependencies remains challenging. Current methods face an efficiency-expressivity trade-off: Channel Independence (CI) achieves high efficiency but ignores cross-variable correlations, while attention-based models capture correlations but suffer from quadratic complexity and noise sensitivity. To reconcile these objectives, we propose the Dynamic Periodic Interaction and Adaptation network (DPIA). Anchored in an efficient centralized interaction architecture, DPIA introduces three distinct yet synergistic innovations to address current limitations. First, the Periodical Patch Mixer (PPM) transforms raw, noisy series into structured pattern representations via dynamic period detection, shifting interaction from points to patterns. Second, the Personalized Interaction Module (PIM) employs an adaptive gating mechanism to allow individual channels to selectively integrate global information based on their idiosyncratic characteristics. Third, the Dispatch-Fusion Block (DFB) serves as a vectorized Mixture-of-Experts (MoE) prediction head that dynamically routes heterogeneous features to specialized experts. Extensive experiments on multiple real-world benchmarks demonstrate that DPIA achieves competitive or superior performance across most scenarios while ensuring high computational efficiency. The source code is publicly available at https://github.com/Du-Team/DPIA.

## 1. Introduction

Multivariate Time Series (MTS) forecasting serves as a fundamental component in a wide range of mission-critical applications, including financial analysis, energy management, and intelligent transportation systems. The objective of MTS forecasting is to accurately predict future values of multiple correlated variables over time, which requires models to simultaneously capture complex temporal dynamics and inter-variable dependencies. This task is inherently challenging due to a structural duality in MTS data: meaningful patterns often manifest as long-term periodic or trend-related structures along the temporal dimension, while the relationships among variables are dynamic, heterogeneous, and frequently non-stationary across the channel dimension.

Early deep learning approaches, such as recurrent and convolutional architectures [1,2], provide effective tools for modeling sequential dependencies, while more recent Transformer-based models [3] further enhance the ability to capture global temporal and cross-variable interactions through attention mechanisms. Despite these advances, existing methods continue to face fundamental limitations when balancing modeling capacity, robustness, and computational efficiency in practical forecasting scenarios. Attention-based architectures explicitly model cross-channel correlations and long-range temporal dependencies, but their quadratic computational complexity with respect to sequence length makes them expensive and sensitive to high-frequency noise in long time series. In contrast, lightweight linear or MLP-based models [4] often adopt a Channel Independence (CI) strategy to achieve linear complexity and improved stability, yet this simplification comes at the cost of completely discarding inter-variable interactions that are crucial for accurate multivariate forecasting. As a result, current approaches either suffer from excessive computational overhead or rely on overly restrictive assumptions that limit their representational power.

Recent efforts attempt to alleviate this tension by introducing centralized interaction mechanisms [5] that aggregate information across channels with linear complexity. We argue that this centralized paradigm is intrinsically suitable for multivariate scenarios because

* Corresponding author.
*E-mail addresses:* tancheng@jsnu.edu.cn (C. Tan), zwbwen@jsnu.edu.cn (W. Zhang), dumj@jsnu.edu.cn (M. Du).

https://doi.org/10.1016/j.patcog.2026.114310
Received 4 February 2026; Received in revised form 29 April 2026; Accepted 21 June 2026
Available online 26 June 2026

it strikes a superior balance between modeling capacity and efficiency. Unlike channel-independent methods that completely ignore inter-series correlations, or vanilla attention mechanisms that suffer from quadratic complexity and noise sensitivity, the centralized approach aggregates diverse channel information into a global representation. This structure effectively filters local noise while capturing essential global dependencies, providing a more robust foundation for high-dimensional series interaction. Although such paradigms improve efficiency, they still exhibit notable shortcomings. First, interactions are typically performed directly on point-wise time series, which exposes the global representation to stochastic high-frequency fluctuations and hinders the extraction of robust semantic structures. Second, centralized representations are usually broadcast uniformly to all channels, overlooking the inherent heterogeneity among variables and their differing requirements for global contextual information. Finally, even when rich and diverse features are obtained after interaction, most existing methods rely on static, single-head linear predictors, which lack the adaptability to handle heterogeneous dynamics across channels and thus form a structural bottleneck in the forecasting pipeline.

To effectively mitigate the aforementioned limitations, we propose the Dynamic Periodic Interaction and Adaptation network (DPIA). DPIA establishes a novel framework that harmonizes high-efficiency centralized interaction with tailored cross-variable dependencies. The core contributions of our work, constituting a systematic and end-to-end modeling pipeline, are as follows:

- We introduce the Periodical Patch Mixer (PPM), which transforms raw time series into structured pattern representations via dynamic period detection. By shifting the modeling granularity from points to patterns, the PPM captures semantic temporal structures while effectively filtering out high-frequency noise.
- We design the Personalized Interaction Module (PIM), which replaces uniform aggregation with a tailored gating mechanism. It enables channels to selectively integrate global context based on their idiosyncratic characteristics, significantly enhancing representational flexibility.
- We develop the Dispatch-Fusion Block (DFB), a vectorized Mixture-of-Experts (MoE) prediction head that dynamically routes heterogeneous features to specialized experts. This design breaks the bottleneck of static predictors, accommodating complex dynamics while maintaining high computational efficiency.
- Extensive experiments on eight real-world benchmarks demonstrate that DPIA achieves competitive or superior performance across the majority of scenarios compared to leading baselines.

The remaining parts of the paper are organized as follows: Section 2 discusses the related work, Section 3 details the methodology, Section 4 presents the experimental evaluations, and Section 5 concludes the paper.

## 2. Related work

### 2.1. Time series forecasting

The evolutionary trajectory of time series forecasting has undergone a profound paradigm shift, moving from classical statistical frameworks to sophisticated neural representation learning, and more recently, the proliferation of universal foundation models. Early traditional statistical forecasting methods predominantly relied on models such as the Autoregressive Integrated Moving Average (ARIMA) [6] and exponential smoothing techniques [7]. While effective for stable data, traditional statistical models falter when confronted with the high-dimensional complexity of contemporary time series. This necessitated a shift toward connectionist approaches, where architectures like LSTM [1] and TCN [2] integrate robust mechanisms for temporal dependency modeling. The state-of-the-art has since been redefined by Transformer-based designs [3], which replace recurrent or convolutional priors with self-attention, thereby enabling the capture of comprehensive global context across extended horizons.

Despite their success, the prohibitive quadratic complexity and substantial memory footprint of standard attention mechanisms have catalyzed a surge in efficiency-oriented architectures. For instance, Informer [8] introduces ProbSparse attention, while Autoformer [9] and FEDformer [10] integrate series decomposition and frequency-domain mappings to capture long-range dependencies more efficiently. Concurrently, a resurgence of MLP-based frameworks challenges the Transformer hegemony. Models such as DLinear [4], N-BEATS [11], TSMixer [12], and TiDE [13] demonstrate that parsimonious linear mappings can achieve superior performance with a significantly reduced computational footprint. Entering the current research frontier, the paradigm has pivoted decisively toward Time Series Foundation Models (TSFMs). TimeCMA [14] and CALF [15] treat time series as a language, while emerging cross-modality paradigms [16] leverage large-scale pretraining on multiple domains, achieving remarkable zero-shot or few-shot forecasting. Furthermore, generative architectures like Timer [17] and MOMENT [18] continue to extend the boundaries of universal representation learning. Notwithstanding these breakthroughs, a critical semantic gap persists between granular point-wise processing and high-level pattern abstraction. Most contemporary frameworks remain susceptible to high-frequency stochasticity. They lack a mechanism capable of distilling raw signals into structured periodic representations before the interaction stage.

### 2.2. Channel interaction mechanisms in MTS

Effectively modeling inter-variable dependencies constitutes a fundamental pillar of multivariate time series forecasting; however, it presents a persistent trade-off between interaction capacity and robustness. Early strategies frequently implemented inter-channel mixing via rudimentary linear projections prior to temporal modeling, which proved highly sensitive to distribution shifts and inter-channel noise [4]. To mitigate these issues, the CI paradigm has gained considerable traction, spearheaded by PatchTST [19] and MICN [20]. By treating each variable as an isolated univariate entity, CI-based architectures effectively insulate the model from inter-channel noise propagation, thereby bolstering the stability of temporal representations. Nevertheless, the intrinsic limitation of CI lies in its systematic neglect of vital cross-variable correlations—information that is indispensable for capturing the synergistic dynamics inherent in complex multivariate systems.

Recent research seeks to bridge this gap by introducing more sophisticated interaction mechanisms. Crossformer [21] explicitly captures dependencies across both temporal and channel dimensions via a two-stage attention mechanism, while iTransformer [22] adopts a radical paradigm shift—inverting the traditional Transformer structure to treat each variable as a token, thereby capturing global correlations within linear complexity. Centralized interaction frameworks, exemplified by SOFTS [5], further optimize efficiency through global core fusion. Nevertheless, these paradigms typically distribute a homogenized representation to all channels. Such undifferentiated broadcasting fails to account for the inherent heterogeneity of multivariate data. To address this, recent works have explored more granular interaction schemes. For instance, CVACL-MA [23] introduces a multi-adapter framework that performs comprehensive variate analysis and collaborative learning to capture diverse inter-variable dependencies. Similarly, in broader MTS analysis tasks such as anomaly detection, explicitly modeling complex inter-variable correlations has also proven crucial; for example, MST-GAT [24] leverages graph attention networks to explicitly capture intra- and inter-modal spatial–temporal dependencies among heterogeneous sensors. While such approaches mitigate the limitations of uniform broadcasting, effectively balancing global context aggregation with efficient, channel-specific information retrieval remains a critical objective. Consequently, there is a compelling need for an

adaptive interaction paradigm that replaces static broadcasting with a personalized subscription strategy, enabling individual variables to selectively assimilate global information based on their idiosyncratic characteristics.

### 2.3. Periodicity and multi-scale pattern modeling

Capturing robust periodicity is essential for deciphering the underlying structure of temporal data across multiple scales. Classical decomposition methods such as STL [25] separate series into trend and seasonal components, a philosophy integrated into deep architectures like Autoformer [9] through autocorrelation mechanisms. To further address multi-scale variations, TimesNet [26] transforms 1D sequences into 2D tensors based on identified periods to model both intra-period and inter-period fluctuations simultaneously. Concurrently, FEDformer [10] and FreTS [27] leverage frequency domain analysis, proving that frequency domain MLPs are highly effective learners that can filter out time domain noise. Building on this direction, TFDNet [28] and PDFNet [29] introduce sophisticated time–frequency decomposition and multi-period modeling to enhance feature separability. Furthermore, extending to unsupervised representation learning, FCACC [30] proposes a fuzzy cluster-aware contrastive framework to effectively capture diverse temporal patterns from unlabeled data. Beyond continuous numerical time series, the necessity of modeling cyclical and recurrent historical patterns is equally critical in discrete temporal reasoning tasks; for instance, TiRGN [31] employs a time-guided recurrent graph network to seamlessly integrate sequential, repetitive, and periodic historical dynamics for temporal knowledge graphs. However, while these methods successfully extract rich temporal features, they typically focus on global or instance-level representations, often lacking a mechanism to align heterogeneous multi-scale patterns into a unified semantic space for fine-grained interaction.

In addition to frequency-domain and attention-based modeling strategies, ModernTCN [32] re-engineers pure convolutional architectures to enhance their versatility across general time series tasks. Meanwhile, Path-Former [33] and TimeMixer [34] introduce adaptive routing pathways and decomposable multi-scale mixing mechanisms, respectively, to effectively capture the hierarchical granularity of complex temporal patterns. Despite these advances, most contemporary methods operate directly on point-wise observations or static-scale representations. High-frequency noise in such inputs frequently compromises feature extraction. There is therefore a clear need for a dynamic pre-processor that can identify multi-scale periodicities and align heterogeneous temporal patterns into a unified semantic space. This would shift the interaction granularity from unstable raw points to robust semantic patterns, improving modeling stability across diverse temporal horizons.

### 2.4. Adaptive prediction and mixture of experts

To accommodate the heterogeneity and complex temporal dynamics of multivariate channels, the field increasingly pivots toward adaptive architectures and sparse modeling. The Mixture-of-Experts paradigm allows for scaling model capacity without a proportional increase in inference costs by activating only a subset of specialized predictors [35, 36]. Within the time series domain, the AMD framework [37] explores adaptive multi-scale decomposition using experts, while Time-MoE [38] exemplifies a recent milestone in scaling foundation models to the billion-parameter regime.

Simultaneously, the resurgence of State Space Models (SSMs) [39] provides powerful linear time alternatives to Transformers for long sequence modeling. Similarly, recent breakthroughs like xLSTM [40] revitalize traditional recurrent frameworks by modernizing gating mechanisms, thereby achieving superior predictive efficacy in increasingly complex forecasting scenarios. A critical bottleneck persists in current architectures: most models apply a single static prediction head uniformly across all variables. This implicitly assumes that post-interaction features share a uniform mapping space. The assumption is increasingly restrictive. It cannot accommodate the heterogeneous feature distributions that arise from personalized interactions, where different channels exhibit divergent latent dynamics. To surmount this bottleneck, it is imperative to orchestrate a vectorized routing mechanism that dynamically dispatches diverse feature instances to specialized expert kernels based on their idiosyncratic temporal dynamics. Such an architecture effectively transcends the constraints of homogeneous predictors and ameliorates representation conflicts, thereby ensuring that divergent feature distributions are mapped with high fidelity and facilitating a more nuanced and robust forecasting output than traditional monolithic heads.

## 3. DPIA

The goal of multivariate time series forecasting is to predict a future sequence $\boldsymbol{Y} \in \mathbb{R}^{C\times T}$ over a horizon of length $T$, given an input lookback window $\boldsymbol{X} \in \mathbb{R}^{C\times L}$, where $C$ and $L$ denote the number of channels and the window size, respectively. To effectively model these complex dependencies, we propose the DPIA framework, as illustrated in Fig. 1. The framework operates through a progressive processing pipeline that transitions from granular temporal points to structured semantic patterns and, finally, to adaptive feature fusion. The model initiates with PPM, acting as a semantic pattern extractor that transforms raw, entangled time points into structured, multi-scale pattern representations via dynamic period detection. These purified patterns serve a dual purpose: they are primarily fed into PIM to generate refined multivariate features via channel-adaptive interaction, while simultaneously acting as the routing context for DFB. Following PPM, PIM executes adaptive cross-channel coupling to generate refined multivariate features. Finally, DFB synthesizes inputs from both preceding stages. It receives the PIM-enhanced features as content and employs a vectorized MoE head to dynamically route them to specialized predictors based on the extracted pattern context. By organically integrating pattern extraction, personalized interaction, and adaptive routing, DPIA effectively captures intricate dependencies while maintaining high computational efficiency.

### 3.1. Periodical patch mixer

In real-world scenarios, raw time series are frequently corrupted by stochastic fluctuations, rendering direct point-wise interaction suboptimal due to low signal-to-noise ratios. To address this, PPM functions as a robust, data-driven pre-processor designed to distill the raw 1D sequence into structured, salient pattern representations. By shifting the interaction granularity from volatile temporal points to robust semantic motifs, PPM ensures stable feature extraction. This pattern-centric transformation is materialized through a specialized three-stage pipeline.

#### 3.1.1. Dynamic period identification

To capture the inherent multi-scale periodicity, we perform spectral decomposition using the Fast Fourier Transform (FFT). To identify the most salient temporal cycles, we extract the top-K frequency components based on their amplitudes. Specifically, we aggregate the frequency magnitudes by averaging them across all channels to derive a globally shared period set, denoted as $\boldsymbol{P} = \{P_1, \ldots, P_K\}$. This consensus-based approach effectively filters out idiosyncratic, channel-specific noise, ensuring that the identified periods represent the intrinsic structural oscillations of the multivariate system. To optimize deployment, we incorporate a period-caching mechanism that bypasses redundant spectral computations during the inference phase, thereby enhancing throughput.

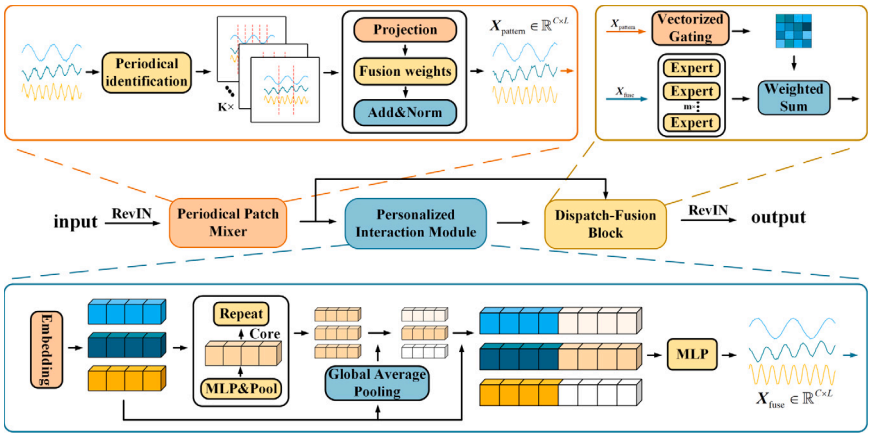


**Fig. 1.** The overall architecture of DPIA.

*3.1.2. Vectorized patching with multi-scale alignment*

A fundamental challenge in multi-period modeling arises from the temporal asynchrony of diverse periodicities (e.g., diurnal vs. weekly cycles), where varying physical lengths preclude efficient parallelization. To resolve this, we propose a Multi-scale Alignment strategy to map heterogeneous temporal motifs into a commensurate representation space.

Specifically, for each identified period $P_k \in \boldsymbol{P}$, the input sequence is segmented into non-overlapping patches via a sliding window. To enable parameter sharing across different temporal scales, we align these varying-length patches into a unified feature space via linear interpolation. This process can be formulated as:

$$\boldsymbol{X}_{\text{pth}}^{(k)} = \tau\Big(\text{Segment}(\boldsymbol{X}, P_k)\Big), \tag{1}$$

where $\text{Segment}(\cdot)$ denotes the segmentation operation, and $\tau(\cdot)$ is a deterministic interpolation operator defined to resample variable-length patches to a fixed semantic length $L_{\text{tsl}}$.

It is worth noting that, $\tau(\cdot)$ is implemented as 1D linear interpolation along the temporal dimension. This choice is deliberate: since the PPM targets dominant periodic structures rather than point-wise high-frequency detail, linear interpolation provides appropriate low-pass smoothing that suppresses stochastic noise during scale alignment. More sophisticated interpolation methods that preserve high-frequency content would conflict with this noise-suppression objective while adding non-trivial computational overhead. Consequently, $\boldsymbol{X}_{\text{pth}}^{(k)} \in \mathbb{R}^{C\times N_k\times L_{\text{tsl}}}$ represents the aligned multi-scale patterns. This alignment ensures that patterns extracted from different periodic scales are mapped into a scale-aligned representation space, enabling a single shared MLP mixer to process them simultaneously in a parameter-efficient and scale-invariant manner. Furthermore, fine-grained signal information is preserved through a residual connection in the subsequent multi-scale pattern fusion stage. By carrying forward the original signal, this structure ensures that high-frequency content is still available to downstream components in the prediction pipeline.

*3.1.3. Multi-scale pattern fusion*

By mapping heterogeneous temporal patterns into a unified representation space with aligned scales, the model effectively overcomes the computational barriers posed by varying physical lengths, thereby enabling deep feature mining within a unified semantic dimension. To extract scale-invariant features, we feed the aligned patterns $\boldsymbol{X}_{\text{pth}}^{(k)}$ into a shared MLP mixer. Subsequently, the processed features are restored to their original temporal scale via an inverse alignment operation:

$$\hat{\boldsymbol{X}}^{(k)} = \rho\Big(\text{MLP}_{\text{mixer}}(\boldsymbol{X}_{\text{pth}}^{(k)})\Big), \tag{2}$$

where $\text{MLP}_{\text{mixer}}(\cdot)$ denotes the shared feature extractor, and $\rho(\cdot)$ represents the inverse alignment operator that reshapes the processed features back to the original sequence format $\hat{\boldsymbol{X}}^{(k)} \in \mathbb{R}^{C\times L}$.

To integrate information from different periodic scales, we employ a learnable weighted fusion followed by a residual connection:

$$\boldsymbol{X}_{\text{pattern}} = \boldsymbol{X} + \phi\left(\sum_{k=1}^{K} w_k \cdot \hat{\boldsymbol{X}}^{(k)}\right), \tag{3}$$

where $\hat{\boldsymbol{X}}^{(k)}$ has the same dimensions as the input $\boldsymbol{X}$, and $w_k$ denotes learnable coefficients that adaptively weight the contributions of various periodic patterns. We employ a learnable projection function $\phi(\cdot)$ to fuse these components. By incorporating $\boldsymbol{X}$ as a residual path, the model effectively preserves the intrinsic signal characteristics.

*3.2. Personalized interaction module*

The PIM operates as the centralized interaction engine of DPIA, engineered to facilitate adaptive cross-channel information exchange while maintaining high computational efficiency. Although existing centralized mechanisms [5] achieve linear complexity, they often encounter the issue of representation homogeneity, where aggregated global information is broadcast uniformly across all channels. Such an approach overlooks the intrinsic heterogeneity within multivariate time series, failing to account for the fact that different variables typically exhibit distinct dependencies on underlying global trends.

To overcome this limitation, PIM introduces a personalized adaptive gating mechanism, effectively transforming static information broadcasting into dynamic, personalized information subscription. The module processes the pattern representations $\boldsymbol{X}_{\text{pattern}} \in \mathbb{R}^{C\times L}$ from the PPM through three synergistic stages: aggregation, gating, and redistribution.

First, in the aggregation stage, the module distills the representations of all channels into a global core representation while maintaining linear computational complexity. To mitigate overfitting and bolster model robustness, consistent with the architectural principles of SOFTS [5], we employ a stochastic pooling strategy during the training phase. This strategy randomly samples channel features to form the core representation $\boldsymbol{o}_{\text{core}} \in \mathbb{R}^{1\times d'}$:

$$\boldsymbol{o}_{\text{core}} = \psi\Big(\text{MLP}_{\text{agg}}(\boldsymbol{X}_{\text{pattern}})\Big), \tag{4}$$

where $\text{MLP}_{\text{agg}}(\cdot)$ projects the input features into a latent core space, and $\psi(\cdot)$ denotes the stochastic pooling operation. Unlike element-wise dropout, which regularizes individual feature dimensions after aggregation has already occurred, stochastic pooling intervenes directly at the channel-selection stage. In centralized aggregation, a small number of channels with the strongest or most regular signals tend to dominate the formation of $\boldsymbol{o}_{\text{core}}$, producing a globally biased representation that causes remaining channels to over-rely on a skewed context. By randomly varying which channels participate in forming the core at

each training iteration, stochastic pooling compels the model to learn a core representation that is invariant across different channel subsets—a form of regularization that targets the specific failure mode of centralized aggregation and cannot be replicated by applying dropout to the already-aggregated output.

Next, the Gating stage directly generates the channel-wise gating tensor $\boldsymbol{g} \in \mathbb{R}^{C\times 1}$ to determine the information demand for all channels simultaneously. To maintain lightweight efficiency, the gating decision is derived from high-level statistical features rather than the full feature map. Specifically, we apply global average pooling to the input features along the temporal dimension and pass the aggregated statistics through a lightweight gating network:

$$\boldsymbol{g} = \sigma(\mathrm{MLP}_{\mathrm{gate}}(\mathrm{Mean}(\boldsymbol{X}_{\mathrm{pattern}}))), \tag{5}$$

where $\mathrm{Mean}(\cdot)$ denotes the temporal averaging operation, $\mathrm{MLP}_{\mathrm{gate}}$ is a channel-agnostic projection network, and $\sigma$ is the Sigmoid activation function that constrains the gate values to $(0, 1)$. This vectorized formulation allows the model to autonomously learn the sensitivity of each channel to global information in parallel.

Finally, in the Redistribution stage, the global core representation $\boldsymbol{o}_{\mathrm{core}}$ is dynamically modulated by the generated gating tensor $\boldsymbol{g}$. This mechanism regulates the dissemination of global context via broadcasted element-wise multiplication, ensuring that global information is integrated according to each channel's specific requirements. The fused output $\boldsymbol{X}_{\mathrm{fuse}} \in \mathbb{R}^{C\times L}$ is then constructed by concatenating the original local features with the gated global information, followed by a linear fusion projection to unify the feature space:

$$\boldsymbol{X}_{\mathrm{fuse}} = \mathrm{MLP}_{\mathrm{fuse}}(\mathrm{Concat}(\boldsymbol{X}_{\mathrm{pattern}}, \boldsymbol{g} \odot \boldsymbol{o}_{\mathrm{core}})), \tag{6}$$

where $\mathrm{Concat}(\cdot)$ denotes the concatenation operation along the feature dimension, $\mathrm{MLP}_{\mathrm{fuse}}$ is a learnable projection network responsible for integrating the hybrid features, and $\odot$ denotes element-wise multiplication with broadcasting. The resulting $\boldsymbol{X}_{\mathrm{fuse}}$ constitutes a refined feature space augmented by deep personalized interactions. It effectively harmonizes the preservation of channel-specific idiosyncrasies with the integration of a comprehensive global context.

### 3.3. Dispatch-fusion block

DFB functions as the adaptive prediction head of DPIA. It receives two inputs directly from the preceding stages: the pattern representation $\boldsymbol{X}_{\mathrm{pattern}} \in \mathbb{R}^{B\times C\times L}$ output by PPM, which serves as the routing context, and the fused representation $\boldsymbol{X}_{\mathrm{fuse}} \in \mathbb{R}^{B\times C\times L}$ output by PIM, which provides the semantic content. These two tensors are the exact same feature tensors produced by their respective modules—no intermediate transformation is applied before they enter DFB. Distinct from the PIM, which prioritizes cross-channel interactions, DFB is explicitly engineered to mitigate negative transfer, a common bottleneck in multivariate time series forecasting. Given that real-world datasets often comprise variables with heterogeneous temporal dynamics (e.g., contrasting smooth seasonality with stochastic fluctuations), conventional channel-shared projections impose an overly rigid mapping that often leads to performance degradation. To circumvent this, we propose a vectorized MoE architecture, which facilitates the dynamic routing of multivariate features to specialized predictors tailored for diverse patterns.

Conventional MoE implementations often suffer from computational inefficiency when handling high-dimensional multivariate data due to channel-wise serial dependencies. To overcome this scalability constraint, we introduce a vectorized execution strategy that reformulates the channel-wise routing as a super-batch operation. By explicitly decoupling routing decisions from the channel dimension, this design facilitates maximal hardware parallelization. Under this paradigm, the batch dimension $B$ is seamlessly integrated into the execution flow. The module accepts two high-dimensional tensors: the pattern representation $\boldsymbol{X}_{\mathrm{pattern}} \in \mathbb{R}^{B\times C\times L}$, which serves as the routing context, and the fused representation $\boldsymbol{X}_{\mathrm{fuse}} \in \mathbb{R}^{B\times C\times L}$, which provides the semantic content.

Prior to computation, we flatten both tensors into a super-batch format $\boldsymbol{X}'_{\mathrm{pattern}}, \boldsymbol{X}'_{\mathrm{fuse}} \in \mathbb{R}^{(B\cdot C)\times L}$. This transformation allows the module to treat every channel feature as an independent sample, facilitating parallelized processing. We then employ a routing network to generate the decision matrix $\boldsymbol{G}_{\mathrm{route}} \in \mathbb{R}^{(B\cdot C)\times N_e}$. Following the adaptive synthesis paradigm proposed in AMD [37], we adopt its differentiable top-K routing mechanism to address the inherent non-differentiability of discrete expert selection. This mechanism provides a smooth approximation to hard top-K routing, enabling input-adaptive expert activation while preserving gradient flow during training.

Concretely, we first compute an initial expert probability distribution $\boldsymbol{p}$ by introducing stochastic gating noise to encourage expert exploration and mitigate premature expert collapse:

$$\boldsymbol{p} = \mathrm{Softmax}(\mathrm{MLP}_{\mathrm{route}}(\boldsymbol{X}'_{\mathrm{pattern}}) + \boldsymbol{\epsilon}), \tag{7}$$

where $\mathrm{MLP}_{\mathrm{route}}$ maps aligned features to the expert decision space and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$ denotes Gaussian noise. A differentiable top-K transformation is then applied to modulate the routing distribution:

$$\mathrm{TopK}(p_j, k) = \begin{cases} \alpha \cdot (\exp(p_j) - 1), & \text{if } p_j \geq v_k \\ \alpha \cdot \ln(p_j + 1), & \text{if } p_j < v_k, \end{cases} \tag{8}$$

where $p_j$ denotes the $j$th element of the expert probability vector $\boldsymbol{p}$, $v_k$ is the $k$th largest value in $\boldsymbol{p}$, and $\alpha$ is a scaling factor that follows the same setting as in the AMD framework. This design choice ensures consistency with prior differentiable top-K routing mechanisms and avoids introducing additional tunable hyperparameters. The exponential mapping amplifies the contributions of the top-K experts, while the logarithmic mapping suppresses non-top-K candidates without breaking gradient flow. Subsequently, the transformed logits are normalized to obtain the routing weights:

$$\boldsymbol{G}_{route} = \mathrm{Softmax}(\mathrm{TopK}(\boldsymbol{p}, k)). \tag{9}$$

This formulation yields a dense yet strictly prioritized routing distribution, in which dominant experts are exponentially emphasized while less relevant ones are smoothly suppressed, enabling sparse, specialized expert execution at inference time while maintaining stable gradient propagation during training.

Finally, guided by the routing weights $\boldsymbol{G}_{\mathrm{route}}$, $N_e$ expert networks are executed in parallel to process the fused content tensor $\boldsymbol{X}'_{\mathrm{fuse}}$. The outputs of individual experts are then aggregated through a weighted combination:

$$\boldsymbol{Y}'_{\mathrm{pred}} = \sum_{j=1}^{N_e} \left(\boldsymbol{G}_{\mathrm{route}}[:, j] \odot \boldsymbol{E}_j(\boldsymbol{X}'_{\mathrm{fuse}})\right), \tag{10}$$

where $\boldsymbol{E}_j$ denotes the $j$th expert function and $\odot$ represents element-wise multiplication. This formulation enables input-adaptive expert dispatching, allowing different experts to specialize in modeling distinct temporal characteristics across variables. The aggregated representations are subsequently reshaped back to $\mathbb{R}^{B\times C\times T}$ to produce the final prediction $\hat{\boldsymbol{Y}}$.

### 3.4. Loss function

To optimize the proposed DPIA framework, we define a joint learning objective designed to simultaneously minimize forecasting divergence and promote equitable expert utilization. The comprehensive loss function, denoted as $\mathcal{L}_{\mathrm{total}}$, is formulated as a weighted sum of the prediction loss and an auxiliary regularization term:

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{pred}} + \lambda \cdot \mathcal{L}_{\mathrm{balance}}. \tag{11}$$

where the hyperparameter $\lambda$ serves to modulate the intrinsic trade-off between predictive fidelity and expert diversity. The prediction loss $\mathcal{L}_{\mathrm{pred}}$ employs the standard Mean Squared Error (MSE) to quantify

**Table 1**
Detailed dataset descriptions.

| Dataset | Dim | Dataset size | Frequency | Information |
|---|---|---|---|---|
| ETTh1 | 7 | (8545,2881,2881) | Hourly | Electricity |
| ETTh2 | 7 | (8545,2881,2881) | Hourly | Electricity |
| ETTm1 | 7 | (34465,11521,11521) | 15 min | Electricity |
| ETTm2 | 7 | (34465,11521,11521) | 15 min | Electricity |
| Weather | 21 | (36792,5271,10540) | 10 min | Weather |
| ECL | 321 | (18317,2633,5261) | Hourly | Electricity |
| Traffic | 862 | (12185,1757,3509) | Hourly | Transportation |
| Solar | 137 | (36601,5161,10417) | 10 min | Energy |

the discrepancy between the predicted sequence $\hat{\boldsymbol{Y}}$ and the ground truth $\boldsymbol{Y}$:

$$\mathcal{L}_{\text{pred}} = \frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{Y}_{:,:,t} - \hat{\boldsymbol{Y}}_{:,:,t}\|_F^2. \tag{12}$$

To fully harness the representational capacity of the MoE head, we incorporate an auxiliary load-balancing $\mathcal{L}_{\text{balance}}$, following the design of the selector loss $\mathcal{L}_{\text{selector}}$ introduced in the AMD framework. Both losses share the same mathematical formulation and are motivated by the common objective of preventing expert collapse. In complex multivariate forecasting scenarios, routing mechanisms tend to over-activate a small subset of dominant experts, leading to inefficient parameter utilization. By penalizing the distributional imbalance of the importance weights $\boldsymbol{w}_i$, the load-balancing term encourages a more equitable allocation of training signals across the expert pool. This, in turn, facilitates stable training of DFB and improves its ability to model diverse temporal regimes. Formally, this auxiliary loss is defined as follows:

$$\mathcal{L}_{\text{balance}} = \sum_{i=1}^{C} \frac{\text{Var}(\boldsymbol{w}_i)}{(\text{Mean}(\boldsymbol{w}_i))^2 + \delta}, \tag{13}$$

where $\boldsymbol{w}_i \in \mathbb{R}^{N_e}$ denotes the accumulated routing weights assigned to the $i$th channel across all experts, aggregated over the training batch, and $\delta$ is a small constant for numerical stability. This term optimizes the gating mechanism by penalizing the variance in expert assignment, ensuring a balanced workload distribution.

## 4. Experiments

### 4.1. Experimental settings

#### 4.1.1. Datasets

To verify the effectiveness of DPIA, we conduct extensive experiments on eight widely recognized real-world benchmarks, comprising Weather, Solar-Energy, Electricity, Traffic, and the full suite of ETT datasets (ETTh1, ETTh2, ETTm1, and ETTm2). These datasets represent diverse temporal dynamics and are standard for evaluating multivariate time series forecasting models. Comprehensive statistics for each dataset are summarized in Table 1.

#### 4.1.2. Baselines

To establish a robust and comprehensive benchmark, we evaluate DPIA against an array of state-of-the-art baselines categorized into three representative paradigms: (1) MLP-based architectures, including TimeMixer [34], SOFTS [5], and DLinear [4]; (2) Transformer-based models, such as iTransformer [22], PatchTST [19], and TQNet [41]; and (3) CNN-based approaches, represented by TimesNet [26] and CASA [42].

#### 4.1.3. Metrics

To guarantee a rigorous and fair comparison, the input lookback window $L$ is fixed at 512 for all competitive models, with performance assessed across four diverse forecasting horizons $T \in \{96, 192, 336, 720\}$. To mitigate the impact of stochasticity and ensure the statistical significance of our findings, all reported results are aggregated as the average of five independent trials. Following established evaluation protocols in time series forecasting, we adopt MSE and Mean Absolute Error (MAE) as the primary performance metrics.

### 4.2. Comparison experiments

Table 2 summarizes the multivariate long-term forecasting performance. DPIA consistently outperforms the competitive baselines across the majority of datasets. This superiority is primarily attributed to its capacity to accurately capture the nuanced characteristics of heterogeneous time series. In datasets such as ETT and Weather, where variables exhibit distinct physical semantics and high channel heterogeneity, the PIM module leverages its adaptive gating mechanism to derive variable-specific interaction patterns. Consequently, DPIA significantly surpasses the uniform interaction paradigm of SOFTS. Furthermore, DPIA achieves a clear margin over PatchTST on the Solar-Energy dataset. This advantage stems from the PPM, which distills dominant periodicities from the frequency domain, providing a more robust structural representation than PatchTST's raw point-wise processing. Simultaneously, DPIA effectively models intricate inter-channel dependencies, which are inherently neglected by the Channel-Independent strategy of PatchTST.

Quantitatively, across all 40 evaluation settings consisting of 4 prediction horizons and 1 average result for each of the 8 datasets, DPIA achieves the lowest MSE and MAE in 27 settings, representing a win rate of 67.5%. These results indicate that the proposed framework achieves competitive performance compared to existing state-of-the-art baselines across the majority of evaluated scenarios.

### 4.3. Ablation studies

To rigorously quantify the individual contribution of each core component within the DPIA framework, we conduct a series of ablation experiments by systematically substituting key modules with standardized counterparts. The quantitative performance of these variants is summarized in Table 3. Specifically, we evaluate three degraded versions: (1) w/o PPM, where the Periodical Patch Mixer is replaced by a vanilla MLP; (2) w/o PIM, where the Personalized Interaction Module is replaced by the uniform STAR interaction paradigm; and (3) w/o DFB, where the Dispatch-Fusion Block is replaced by a standard linear projection.

The ablation results reveal distinct performance patterns that validate our architectural design. Notably, the w/o PPM variant exhibits a precipitous performance degradation, particularly on high-dimensional datasets such as Traffic and ECL. This decline underscores the indispensable role of the PPM in transforming raw, stochastic time series into structured periodic representations. Unlike a generic MLP, which lacks spectral sensitivity, the PPM effectively filters input-level noise and prevents it from propagating to downstream components, thereby ensuring the integrity of the latent features.

Regarding the w/o PIM variant, the performance disparity is highly contingent upon dataset characteristics. On datasets characterized by pronounced channel heterogeneity — such as ECL and Solar — the forecasting error escalates significantly. This is because the uniform interaction paradigm of the STAR module is insufficient to accommodate the divergent dependency patterns inherent in different variables. Conversely, on relatively homogeneous datasets like the ETT series, the performance gap is notably constricted. This contrast confirms that our personalized mechanism is explicitly engineered to reconcile the complex, heterogeneous inter-variable dependencies that conventional uniform methods typically overlook.

The w/o DFB variant yields consistently higher errors across all benchmarks, indicating that the post-interaction features are highly

**Table 2**
Multivariate long-term forecasting results. The best result is marked in bold and the second best in underline. Avg means the average results from all four prediction lengths.

| Models | | DPIA(Ours) | | TQNet | | CASA | | TimeMixer | | SOFTS | | iTransformer | | PatchTST | | DLinear | | TimesNet | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | 96 | **0.376** | **0.404** | 0.377 | 0.405 | 0.389 | 0.418 | 0.379 | 0.406 | 0.382 | 0.409 | 0.395 | 0.420 | 0.388 | 0.414 | 0.380 | 0.407 | 0.461 | 0.465 |
| | 192 | **0.407** | **0.424** | 0.408 | 0.425 | 0.430 | 0.445 | 0.421 | 0.436 | 0.416 | 0.432 | 0.431 | 0.445 | 0.475 | 0.473 | 0.408 | **0.424** | 0.478 | 0.480 |
| | 336 | **0.425** | **0.437** | 0.435 | 0.447 | 0.446 | 0.461 | 0.446 | 0.457 | 0.442 | 0.450 | 0.448 | 0.459 | 0.473 | 0.471 | 0.437 | 0.446 | 0.497 | 0.494 |
| | 720 | **0.461** | **0.474** | 0.507 | 0.500 | 0.496 | 0.503 | 0.480 | 0.487 | 0.462 | 0.483 | 0.523 | 0.522 | 0.489 | 0.491 | 0.485 | 0.503 | 0.557 | 0.531 |
| | Avg | **0.417** | **0.435** | 0.432 | 0.444 | 0.440 | 0.457 | 0.431 | 0.446 | 0.425 | 0.444 | 0.449 | 0.461 | 0.456 | 0.462 | 0.428 | 0.445 | 0.498 | 0.493 |
| ETTh2 | 96 | **0.285** | **0.347** | 0.289 | 0.350 | 0.295 | 0.356 | 0.287 | 0.355 | 0.292 | 0.356 | 0.314 | 0.365 | 0.320 | 0.378 | 0.295 | 0.362 | 0.384 | 0.418 |
| | 192 | **0.363** | **0.396** | 0.364 | 0.398 | 0.364 | 0.397 | 0.378 | 0.413 | 0.369 | 0.399 | 0.379 | 0.407 | 0.417 | 0.433 | 0.378 | 0.416 | 0.405 | 0.433 |
| | 336 | **0.380** | **0.414** | 0.384 | 0.416 | 0.391 | 0.422 | 0.382 | 0.420 | 0.394 | 0.422 | 0.421 | 0.437 | 0.441 | 0.453 | 0.468 | 0.477 | 0.405 | 0.440 |
| | 720 | **0.417** | **0.446** | 0.420 | 0.451 | 0.430 | 0.458 | 0.432 | 0.459 | 0.420 | 0.452 | 0.439 | 0.461 | 0.475 | 0.485 | 0.609 | 0.561 | 0.464 | 0.477 |
| | Avg | **0.361** | **0.401** | 0.364 | 0.404 | 0.370 | 0.408 | 0.370 | 0.412 | 0.369 | 0.407 | 0.388 | 0.418 | 0.413 | 0.437 | 0.437 | 0.454 | 0.414 | 0.442 |
| ETTm1 | 96 | **0.296** | 0.350 | 0.299 | 0.351 | 0.300 | 0.355 | 0.306 | 0.356 | 0.301 | 0.355 | 0.311 | 0.365 | 0.298 | 0.352 | 0.304 | **0.348** | 0.331 | 0.378 |
| | 192 | **0.340** | 0.376 | 0.341 | 0.377 | 0.341 | 0.384 | 0.346 | 0.381 | 0.344 | 0.382 | 0.348 | 0.385 | 0.359 | 0.395 | **0.340** | **0.372** | 0.430 | 0.423 |
| | 336 | **0.373** | **0.393** | **0.373** | 0.396 | 0.374 | 0.401 | 0.375 | 0.400 | 0.376 | 0.401 | 0.377 | 0.403 | 0.385 | 0.409 | 0.374 | 0.395 | 0.407 | 0.425 |
| | 720 | 0.428 | **0.421** | 0.432 | 0.426 | 0.429 | 0.432 | 0.466 | 0.446 | 0.456 | 0.448 | 0.436 | 0.437 | 0.442 | 0.445 | **0.427** | 0.425 | 0.472 | 0.462 |
| | Avg | **0.359** | **0.385** | 0.361 | 0.388 | 0.361 | 0.393 | 0.373 | 0.396 | 0.369 | 0.397 | 0.368 | 0.398 | 0.371 | 0.401 | 0.361 | **0.385** | 0.410 | 0.422 |
| ETTm2 | 96 | 0.172 | 0.261 | 0.174 | **0.260** | 0.182 | 0.267 | 0.176 | 0.268 | 0.182 | 0.267 | 0.179 | 0.271 | 0.185 | 0.277 | **0.167** | **0.260** | 0.207 | 0.290 |
| | 192 | 0.231 | **0.304** | 0.238 | 0.306 | 0.235 | 0.305 | 0.231 | 0.305 | 0.250 | 0.312 | 0.243 | 0.313 | 0.239 | 0.316 | **0.229** | 0.309 | 0.255 | 0.323 |
| | 336 | **0.276** | **0.334** | 0.283 | 0.336 | 0.288 | 0.340 | 0.288 | 0.341 | 0.294 | 0.341 | 0.291 | 0.346 | 0.297 | 0.348 | 0.290 | 0.352 | 0.319 | 0.362 |
| | 720 | **0.360** | **0.385** | 0.372 | 0.393 | 0.364 | 0.389 | 0.376 | 0.392 | 0.378 | 0.396 | 0.378 | 0.398 | 0.384 | 0.406 | 0.409 | 0.430 | 0.418 | 0.420 |
| | Avg | **0.260** | **0.321** | 0.267 | 0.324 | 0.267 | 0.325 | 0.268 | 0.326 | 0.276 | 0.329 | 0.273 | 0.332 | 0.277 | 0.337 | 0.274 | 0.338 | 0.299 | 0.349 |
| Weather | 96 | **0.147** | **0.199** | 0.156 | 0.207 | 0.157 | 0.209 | **0.147** | **0.199** | 0.156 | 0.207 | 0.164 | 0.215 | 0.151 | 0.202 | 0.171 | 0.231 | 0.168 | 0.225 |
| | 192 | **0.191** | **0.241** | 0.204 | 0.253 | 0.205 | 0.255 | 0.193 | 0.242 | 0.212 | 0.258 | 0.211 | 0.256 | 0.204 | 0.256 | 0.214 | 0.271 | 0.221 | 0.271 |
| | 336 | **0.240** | **0.279** | 0.257 | 0.293 | 0.259 | 0.294 | 0.242 | 0.281 | 0.265 | 0.299 | 0.266 | 0.296 | 0.257 | 0.294 | 0.258 | 0.309 | 0.285 | 0.317 |
| | 720 | **0.319** | **0.332** | 0.324 | 0.338 | 0.341 | 0.351 | **0.319** | 0.334 | 0.354 | 0.356 | 0.341 | 0.345 | 0.341 | 0.352 | 0.320 | 0.358 | 0.352 | 0.360 |
| | Avg | **0.224** | **0.263** | 0.235 | 0.273 | 0.241 | 0.277 | 0.225 | 0.264 | 0.247 | 0.280 | 0.245 | 0.278 | 0.238 | 0.276 | 0.241 | 0.292 | 0.256 | 0.293 |
| ECL | 96 | 0.134 | 0.237 | 0.134 | 0.231 | **0.131** | 0.227 | 0.146 | 0.246 | **0.131** | **0.226** | 0.132 | 0.228 | 0.135 | 0.236 | 0.139 | 0.238 | 0.184 | 0.287 |
| | 192 | 0.154 | 0.253 | 0.155 | 0.250 | 0.156 | 0.251 | 0.156 | 0.252 | 0.155 | 0.248 | 0.154 | 0.249 | **0.150** | **0.247** | 0.154 | 0.254 | 0.190 | 0.292 |
| | 336 | 0.172 | 0.272 | 0.174 | 0.271 | 0.169 | 0.265 | 0.178 | 0.275 | 0.169 | 0.266 | 0.169 | 0.265 | **0.167** | **0.264** | **0.167** | 0.268 | 0.207 | 0.307 |
| | 720 | 0.207 | 0.301 | 0.210 | 0.300 | 0.198 | 0.293 | 0.255 | 0.341 | 0.202 | 0.294 | **0.194** | **0.288** | 0.203 | 0.295 | 0.204 | 0.304 | 0.263 | 0.348 |
| | Avg | 0.167 | 0.266 | 0.168 | 0.263 | 0.164 | 0.259 | 0.184 | 0.278 | 0.164 | 0.258 | **0.162** | **0.257** | 0.163 | 0.260 | 0.166 | 0.266 | 0.211 | 0.308 |
| Solar | 96 | **0.180** | **0.230** | 0.183 | 0.247 | 0.189 | 0.238 | 0.209 | 0.257 | 0.198 | 0.244 | 0.214 | 0.259 | 0.188 | 0.256 | 0.207 | 0.273 | 0.201 | 0.270 |
| | 192 | **0.202** | **0.254** | 0.203 | 0.261 | **0.202** | 0.256 | 0.223 | 0.270 | 0.206 | 0.265 | 0.227 | 0.281 | 0.204 | 0.271 | 0.226 | 0.288 | 0.232 | 0.292 |
| | 336 | **0.203** | **0.253** | 0.208 | 0.267 | 0.205 | 0.257 | 0.228 | 0.278 | 0.235 | 0.278 | 0.228 | 0.287 | 0.207 | 0.270 | 0.240 | 0.299 | 0.242 | 0.287 |
| | 720 | **0.216** | **0.268** | 0.222 | 0.274 | **0.216** | 0.270 | 0.229 | 0.277 | 0.217 | 0.277 | 0.223 | 0.284 | 0.218 | 0.282 | 0.249 | 0.306 | 0.240 | 0.295 |
| | Avg | **0.200** | **0.251** | 0.204 | 0.262 | 0.203 | 0.255 | 0.222 | 0.270 | 0.214 | 0.266 | 0.223 | 0.278 | 0.204 | 0.270 | 0.230 | 0.292 | 0.229 | 0.286 |
| Traffic | 96 | 0.401 | 0.298 | 0.401 | 0.295 | 0.368 | 0.267 | 0.372 | 0.260 | **0.351** | **0.250** | 0.352 | 0.258 | 0.367 | 0.259 | 0.415 | 0.302 | 0.603 | 0.325 |
| | 192 | 0.417 | 0.305 | 0.407 | 0.292 | 0.382 | 0.271 | 0.386 | 0.267 | **0.371** | **0.259** | **0.371** | 0.267 | 0.385 | 0.267 | 0.422 | 0.298 | 0.616 | 0.330 |
| | 336 | 0.429 | 0.305 | 0.421 | 0.300 | 0.390 | 0.275 | 0.395 | 0.273 | **0.385** | **0.267** | 0.387 | 0.275 | 0.395 | 0.272 | 0.431 | 0.304 | 0.626 | 0.334 |
| | 720 | 0.465 | 0.322 | 0.433 | 0.295 | 0.435 | 0.299 | 0.438 | 0.295 | **0.422** | **0.285** | 0.424 | 0.294 | 0.434 | 0.293 | 0.468 | 0.325 | 0.669 | 0.354 |
| | Avg | 0.428 | 0.308 | 0.416 | 0.296 | 0.394 | 0.278 | 0.398 | 0.274 | **0.382** | **0.265** | 0.384 | 0.274 | 0.395 | 0.273 | 0.434 | 0.307 | 0.628 | 0.336 |
| 1st Count | | **27** | **27** | 1 | 1 | 3 | 0 | 2 | 1 | 6 | 6 | 3 | 2 | 2 | 2 | 5 | 5 | 0 | 0 |

**Table 3**
Ablations on DPIA. Results are averaged from all four prediction lengths.

| Models | DPIA | | w/o PPM | | w/o PIM | | w/o DFB | |
|---|---|---|---|---|---|---|---|---|
| Metric | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | **0.417** | **0.435** | 0.423 | 0.439 | 0.421 | 0.438 | 0.424 | 0.440 |
| ETTh2 | **0.361** | **0.401** | 0.366 | 0.404 | 0.365 | 0.405 | 0.365 | 0.406 |
| ETTm1 | **0.359** | **0.385** | 0.362 | 0.386 | 0.360 | 0.386 | 0.361 | 0.388 |
| ETTm2 | **0.260** | **0.321** | 0.262 | 0.323 | 0.261 | 0.322 | 0.261 | 0.323 |
| Weather | **0.224** | **0.263** | 0.226 | 0.265 | 0.227 | 0.265 | 0.227 | 0.265 |
| ECL | **0.167** | **0.266** | 0.204 | 0.309 | 0.176 | 0.274 | 0.182 | 0.284 |
| Traffic | **0.428** | **0.308** | 0.559 | 0.366 | 0.449 | 0.319 | 0.439 | 0.316 |
| Solar | **0.200** | **0.251** | 0.205 | 0.260 | 0.207 | 0.261 | 0.201 | 0.257 |

heterogeneous in nature. A static MLP head lacks the representational capacity to effectively accommodate such multi-modal diversity. The DFB resolves this bottleneck by employing a vectorized MoE architecture, which dynamically routes varied feature instances to specialized experts. This mechanism ensures that each latent temporal pattern is processed by the optimally suited predictor, thereby enhancing the overall model expressivity.

### 4.4. Parameter sensitivity

We conduct an extensive sensitivity analysis to evaluate the robustness of the proposed DPIA framework against variations in critical hyperparameters. Specifically, we examine the model's stability relative to fluctuations in input sequence lengths and module-specific architectural configurations, providing an empirical basis for our optimal parameter selection.

The input sequence length $L$ is pivotal for the PPM to capture robust periodic patterns. We empirically evaluate the impact of $L$ across the set $\{96, 192, 336, 512, 720\}$. As illustrated in Fig. 2, increasing $L$ initially leads to a significant reduction in forecasting error, particularly on datasets with rich temporal dynamics like ETTm2 and Weather. This enhancement is primarily attributed to the PPM's capacity to distill more reliable dominant periodicities and global dependencies from longer historical contexts. Meanwhile, we observe that increasing the input sequence length from $L = 336$ to $L = 512$ consistently leads

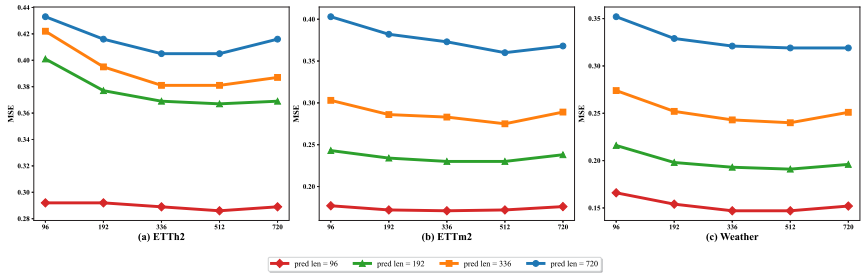


**Fig. 2.** Hyper-parameter sensitivity with respect to the input sequence length. The results are recorded with the all four predict length.

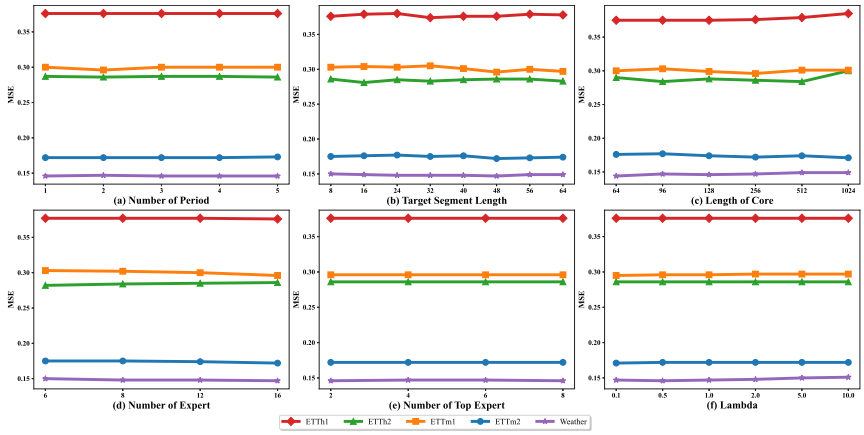


**Fig. 3.** Hyper-parameter sensitivity with respect to six critical hyperparameters. The results are recorded with all five benchmark datasets.

to modest yet stable performance improvements across most datasets, whereas further extending the length to $L = 720$ provides limited additional gains. Although a shorter input such as $L = 336$ already achieves competitive performance, we adopt $L = 512$ as the default setting since it offers more consistent accuracy improvements across diverse benchmarks. This behavior suggests that a moderately longer temporal context helps the model capture more complete periodic structures and long-range dependencies, which is particularly beneficial for multivariate time series with heterogeneous temporal dynamics. Consequently, $L = 512$ is selected as a practical default to favor robustness and predictive accuracy.

To ensure robustness and optimal convergence, we investigate the sensitivity of six pivotal hyperparameters, with results summarized in Fig. 3. For the PPM module, the number of dominant periods is set to $K = 2$, as the results indicate this effectively captures primary cyclic dynamics without introducing high-frequency noise. Simultaneously, the target patch length is set to 48, which provides the optimal granularity for temporal embedding, avoiding the representation bias associated with overly coarse or fine-grained patches. In the PIM module, the dimension of the global core representation is set to $d' = 256$. The analysis reveals a clear U-shaped performance curve, indicating that 256 provides a sufficient information bottleneck to filter noise while retaining essential global trends, whereas smaller dimensions lead to information loss and larger dimensions increase redundancy. Regarding the DFB module, we set the total number of experts to $N_e = 16$ and the number of active experts to $k = 4$. This configuration ensures sufficient capacity to model heterogeneous temporal dynamics while maintaining sparsity for inference efficiency. Finally, the balancing coefficient $\lambda$ for the auxiliary load-balancing loss is set to 1.0. As shown in Fig. 3(f), this value achieves a stable equilibrium, effectively minimizing prediction error while preventing expert collapse.

### 4.5. Efficiency analysis

To assess the practical viability of DPIA for large-scale deployments, we conduct a comparative analysis on the Weather dataset, focusing on computational throughput (training speed), memory occupancy, and forecasting precision. With a fixed lookback window of 512 and a prediction horizon of 96, the empirical results — as illustrated in Fig. 4 — demonstrate that DPIA achieves an optimal trade-off between resource expenditure and predictive fidelity.

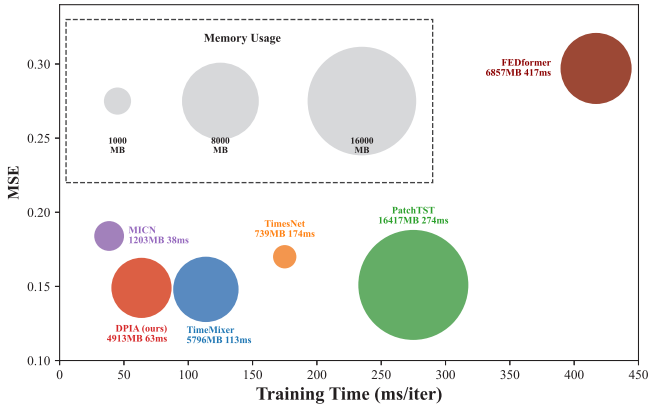


**Fig. 4.** Memory usage (MB), training time (ms/iter), and MSE comparisons on the Weather dataset. Experiments are conducted with an input length of 512 and a prediction horizon of 96.

In terms of hardware resource utilization, DPIA demonstrates superior efficiency with a memory footprint of 4913 MB, whereas PatchTST demands a significantly higher 16,417 MB. This marked disparity underscores the efficiency of our pattern-based interaction and vectorized execution paradigm, which effectively circumvents the quadratic memory complexity inherent in the vanilla attention mechanisms of Transformer [3] and PatchTST [19]. Furthermore, in terms of computational throughput, DPIA records a training latency of 63 ms per iteration. This is significantly faster than other sophisticated architectures, such as

FEDformer (417 ms) and PatchTST (274 ms), validating the operational efficiency of our framework for large-scale time series forecasting.

The trade-off analysis further elucidates the architectural advantages of the proposed framework. While lightweight models such as MICN exhibit a marginally lower latency (38 ms), their predictive fidelity is substantially compromised, yielding an MSE of 0.172 compared to 0.147 for DPIA. Similarly, although TimesNet maintains a compact memory footprint (739 MB), it suffers from a diminished forecasting accuracy (0.168 MSE) and a prohibitive training latency (174 ms). Most notably, when benchmarked against the highly competitive TimeMixer — which matches our accuracy with an MSE of 0.147 — DPIA delivers a nearly 1.8× speedup and an approximately 15% reduction in memory occupancy. Specifically, DPIA achieves its performance with 63 ms and 4913 MB, whereas TimeMixer necessitates 113 ms and 5796 MB. Collectively, these results underscore that DPIA reconciles high-precision forecasting with the computational economy required for large-scale multivariate deployments.

## 5. Conclusion

In this paper, we propose the DPIA network, a novel framework designed to address the inherent challenges of noise interference and channel heterogeneity in multivariate time series forecasting. Adopting a progressive processing strategy, DPIA seamlessly integrates three core components: the periodical patch mixer for multi-scale pattern extraction, the personalized interaction module for adaptive cross-channel information subscription, and the dispatch-fusion block for dynamic expert routing. Extensive experiments on multiple real-world benchmarks demonstrate that DPIA achieves competitive or superior performance across most scenarios while ensuring high computational efficiency. From a broader perspective, our results suggest that explicitly decoupling semantic pattern discovery and resolving channel heterogeneity are critical strategies for advancing robust multivariate time series analysis. However, the current reliance on FFT-based global spectral analysis introduces a stationarity assumption, which limits the model's applicability to series with abrupt regime transitions or gradually evolving seasonality. Future work will explore adaptive time–frequency representations — such as short-time Fourier transforms and learnable wavelet decompositions — to handle such non-stationary periodic signals.

## CRediT authorship contribution statement

**Cheng Tan:** Writing – original draft, Software, Methodology. **Wenbin Zhang:** Writing – original draft, Validation. **Mingjing Du:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work is supported by the Qinglan Project of Jiangsu Province of China, the National Natural Science Foundation of China (No. 62006104), Jiangsu Normal University Faculty Research Excellence Development Program (No. JSNUGZL2026032), Postgraduate Research & Practice Innovation Program of Jiangsu Normal University (No. 2025XKT1457).

## Data availability

Data will be made available on request.

## References

[1] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[2] G. Lai, W. Chang, Y. Yang, H. Liu, Modeling long- and short-term temporal patterns with deep neural networks, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 95–104.
[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[4] A. Zeng, M. Chen, L. Zhang, Q. Xu, Are transformers effective for time series forecasting? in: Proceedings of the AAAI Conference on Artificial Intelligence, 2023, pp. 11121–11128.
[5] L. Han, X.-Y. Chen, H.-J. Ye, D.-C. Zhan, Softs: Efficient multivariate time series forecasting with series-core fusion, in: Advances in Neural Information Processing Systems, 2024, pp. 64145–64175.
[6] P.C. Young, S. Shellswell, Time series analysis, forecasting and control, IEEE Trans. Autom. Control 17 (1972) 281–283.
[7] C.C. Holt, Forecasting seasonals and trends by exponentially weighted moving averages, Int. J. Forecast. 20 (1) (2004) 5–10.
[8] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, W. Zhang, Informer: Beyond efficient transformer for long sequence time-series forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2021, pp. 11106–11115.
[9] H. Wu, J. Xu, J. Wang, M. Long, Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, in: Advances in Neural Information Processing Systems, 2021, pp. 22419–22430.
[10] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, R. Jin, Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting, in: International Conference on Machine Learning, 2022, pp. 27268–27286.
[11] B.N. Oreshkin, D. Carpov, N. Chapados, Y. Bengio, N-BEATS: Neural basis expansion analysis for interpretable time series forecasting, in: International Conference on Learning Representations, 2020.
[12] V. Ekambaram, A. Jati, N. Nguyen, P. Sinthong, J. Kalagnanam, Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 459–469.
[13] I.O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, et al., Mlp-mixer: An all-mlp architecture for vision, in: Advances in Neural Information Processing Systems, 2021, pp. 24261–24272.
[14] C. Liu, Q. Xu, H. Miao, S. Yang, L. Zhang, C. Long, Z. Li, R. Zhao, Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2025, pp. 18780–18788.
[15] P. Liu, H. Guo, T. Dai, N. Li, J. Bao, X. Ren, Y. Jiang, S.-T. Xia, Calf: Aligning llms for time series forecasting via cross-modal fine-tuning, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2025, pp. 18915–18923.
[16] C. Liu, S. Zhou, Q. Xu, H. Miao, C. Long, Z. Li, R. Zhao, Towards cross-modality modeling for time series analytics: A survey in the LLM era, in: International Joint Conference on Artificial Intelligence, 2024, pp. 10564–10572.
[17] Y. Liu, H. Zhang, L. Li, et al., Timer: Generative pre-trained transformers are large time series models, in: International Conference on Machine Learning, 2024, pp. 32369–32399.
[18] M. Goswami, K. Sanyal, et al., MOMENT: A family of open time-series foundation models, in: International Conference on Machine Learning, 2024, pp. 16115–16152.
[19] Y. Nie, N.H. Nguyen, P. Sinthong, J. Kalagnanam, A time series is worth 64 words: Long-term forecasting with transformers, in: International Conference on Learning Representations, 2023.
[20] H. Wang, J. Peng, F. Huang, J. Wang, J. Chen, Y. Xiao, Micn: Multi-scale local and global context modeling for long-term series forecasting, in: International Conference on Learning Representations, 2023.
[21] Y. Zhang, J. Yan, Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting, in: International Conference on Learning Representations, 2023.
[22] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, M. Long, Itransformer: Inverted transformers are effective for time series forecasting, in: International Conference on Learning Representations, 2024.
[23] J. Ye, C. Zhou, X. Zhou, Y. Huang, C. Zhao, CVACL-MA: Comprehensive variate analysis and collaborative learning with multi-adapter for multivariate time series forecasting, Pattern Recognit. 169 (2026) 111936.
[24] C. Ding, S. Sun, J. Zhao, MST-GAT: A multimodal spatial–temporal graph attention network for time series anomaly detection, Inf. Fusion 89 (2023) 527–536.
[25] R.B. Cleveland, W.S. Cleveland, J.E. McRae, I. Terpenning, et al., STL: A seasonal-trend decomposition, J. Off. Stat. 6 (1) (1990) 3–73.
[26] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, M. Long, TimesNet: Temporal 2D-variation modeling for general time series analysis, in: International Conference on Learning Representations, 2023.

[27] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, N. An, D. Lian, L. Cao, Z. Niu, Frequency-domain MLPs are more effective learners in time series forecasting, in: Advances in Neural Information Processing Systems, 2023, pp. 76656–76679.
[28] Y. Luo, S. Zhang, Z. Lyu, Y. Hu, TFDNet: Time–frequency enhanced decomposed network for long-term time series forecasting, Pattern Recognit. 162 (2025) 111412.
[29] G. Tan, Y. Wang, Z. Xiao, D. He, G. Sa, From a multi-period perspective: A periodic dynamics forecasting network for multivariate time series forecasting, Pattern Recognit. 167 (2025) 111760.
[30] C. Wang, M. Du, X. Jiang, Y. Dong, Fuzzy cluster-aware contrastive clustering for time series, Pattern Recognit. 173 (2026) 112899.
[31] Y. Li, S. Sun, J. Zhao, TiRGN: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning, in: International Joint Conference on Artificial Intelligence, 2022, pp. 2152–2158.
[32] D. Luo, J. Wang, ModernTCN: A modern pure convolution structure for general time series analysis, in: International Conference on Learning Representations, 2024.
[33] P. Chen, et al., Path-former: Multi-scale transformers with adaptive pathways, in: International Conference on Learning Representations, 2024.
[34] S. Wang, H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J.Y. Zhang, J. Zhou, TimeMixer: Decomposable multiscale mixing for time series forecasting, in: International Conference on Learning Representations, 2024.
[35] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, J. Dean, Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, in: International Conference on Learning Representations, 2017.
[36] W. Fedus, B. Zoph, N. Shazeer, Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, J. Mach. Learn. Res. 23 (120) (2022) 1–39.
[37] Y. Hu, P. Liu, P. Zhu, D. Cheng, T. Dai, Adaptive multi-scale decomposition framework for time series forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2025, pp. 17359–17367.
[38] X. Shi, S. Wang, Y. Nie, D. Li, Z. Ye, Q. Wen, M. Jin, Time-MoE: Billion-scale time series foundation models with mixture of experts, in: International Conference on Learning Representations, 2025.
[39] A. Gu, K. Goel, C. Ré, Efficiently modeling long sequences with structured state spaces, in: International Conference on Learning Representations, 2022.
[40] M. Beck, K. Pöppel, M. Spanring, A. Auer, O. Prudnikova, M. Kopp, G. Klambauer, J. Brandstetter, S. Hochreiter, xLSTM: Extended long short-term memory, in: Advances in Neural Information Processing Systems, 2024, pp. 107547–107603.
[41] S. Lin, H. Chen, H. Wu, C. Qiu, W. Lin, Temporal query network for efficient multivariate time series forecasting, in: International Conference on Machine Learning, 2025.
[42] M. Lee, H. Yoon, M. Kang, CASA: CNN autoencoder-based score attention for efficient multivariate long-term time-series forecasting, in: International Joint Conference on Artificial Intelligence, 2025, pp. 5563–5570.