ITC 1/55
Information Technology and Control
Vol. 55 / No. 1/ 2026
pp. 243-256
DOI 10.5755/j01.itc.55.1.43646

MP-Transformer: A Hybrid Model Integrating Multi-Period ARIMA and Dynamically Gated Attention for Time-Series Forecasting

Received 2025/11/19 | Accepted after revision 2026/01/29

HOW TO CITE: Shi, Y., Zhou, W. (2026). MP-Transformer: A Hybrid Model Integrating Multi-Period ARIMA and Dynamically Gated Attention for Time-Series Forecasting. *Information Technology and Control*, 55(1), 243-256. https://doi.org/10.5755/j01.itc.55.1.43646

MP-Transformer: A Hybrid Model Integrating Multi-Period ARIMA and Dynamically Gated Attention for Time-Series Forecasting

Yunlong Shi, Weijie Zhou*

School of Information Engineering, Liaoning University of Traditional Chinese Medicine, Shenyang, 110847, China

Corresponding author: 15524333213@163.com

Accurate time-series forecasting is challenging when multiple seasonalities interact with non-linear effects. We present MP-Transformer, a hybrid" decompose-then-refine" framework that couples an interpretable Multi-Period ARIMA baseline with a dynamically gated attention residual learner. The ARIMA component extracts dominant linear trends and multi-scale seasonality via seasonal phase templates with synchronous differencing, yielding an approximately stationary residual series and an interpretable baseline. A Transformer then models the remaining non-linear dynamics using a gated fusion of global attention (for long-range periodic dependencies) and content-driven Top-k local attention (for abrupt short-term variations). Period contributions are learned through non-negative, normalized gating weights. Across multiple real-world datasets, MP-Transformer consistently improves multi-horizon accuracy over statistical, deep, and hybrid baselines. The results demonstrate that combining explicit linear decomposition with implicit residual learning yields robust, data-efficient forecasting and enhanced interpretability.

KEYWORDS: MP-Transformer; Multi-period ARIMA; Dynamically gated attention; Time-series forecasting

1. Introduction

Time series forecasting is a fundamental task in numerous domains, including finance, energy, health care, and supply chain management. Its ultimate objective is to accurately predict future values based on historical observations. Despite extensive research efforts, real-world time series often exhibit complex dynamical characteristics, including long-term trends, multiple seasonal patterns, and intricate non-linear irregularities, making accurate forecasting a persistent challenge.

Traditional statistical learning methods, such as the Autoregressive Integrated Moving Average (ARIMA) model and its seasonal variant, the Seasonal Autoregressive Integrated Moving Average (SARIMA), have been widely applied in time series forecasting [4]. The strengths of these models lie in their interpretability and effectiveness in capturing linear trends and single-seasonal patterns. However, they typically struggle to handle multi-period seasonality and often fail to capture complex non-linear dependencies within the data.

On the other hand, deep learning models, particularly those based on the Transformer architecture, have demonstrated significant potential in the field of time-series forecasting. However, these black-box models typically require large amounts of data for training and struggle to explicitly learn and disentangle strong periodic and trend components from the data, resulting in suboptimal performance on datasets with pronounced regularities. More importantly, prevailing approaches often process linear and non-linear information jointly, failing to fully leverage the distinct characteristics of different types of information for targeted modeling.

To overcome the aforementioned limitations, this paper proposes a novel hybrid forecasting framework, referred to as the MP-Transformer (Multi-Period Transformer). Its core idea is to integrate" explicit modeling" with" implicit learning", adhering to a" decompose-then-refine" design philosophy. Specifically, the model first employs an innovative Multi-Period ARIMA component. Utilizing cyclical phase templates and synchronous differencing technology, this component explicitly extracts the linear components of the series, capturing stable, dominant periodic patterns to generate an approximately stationary residual series and a highly interpretable baseline forecast. Subsequently, a dynamically gated attention Transformer network, specifically designed for residual learning, is utilized to capture the non-linear portion within the residual series, mining potentially complex, time-varying, and interacting periodic patterns that may exist in the sequence.

Extensive experiments demonstrate that MP-Transformer achieves state-of-the-art performance on multiple real-world datasets, significantly outperforming existing mainstream methods. The main contributions of this paper are summarized as follows:

1. We propose MP-Transformer, a principled hybrid framework that explicitly decomposes multi-period linear components via an interpretable ARIMA module and implicitly learns residual nonlinearities with a Transformer-based residual network.
2. We introduce a synchronous differencing strategy and gated fusion of seasonal templates that yield a mathematically consistent multi-period decomposition and expose interpretable period contributions.
3. We design a Dynamically Gated Attention mechanism that blends global self-attention with content-driven top-k local attention, tailored for modelling stationary residual series.

2. Related Work

2.1. Time Series Forecasting

Time series forecasting is a fundamental task of predicting future values based on historical observations. Traditional statistical methods, such as exponential smoothing and ARIMA/SARIMA, have been widely used due to their interpretability and solid theoretical foundation for capturing linear trends and single-seasonal patterns [4-5]. Deep learning approaches have significantly advanced the field: RNNs and LSTMs model temporal dependencies in a sequential manner. Recently, Transformers have emerged as a robust architecture for capturing long-range dependencies. Innovations such as Informer's prob-sparse attention, Autoformer's series decomposition block, FEDformer's frequency-enhanced at-

tention, and PatchTST's patch-level modelling have improved efficiency and effectiveness [7, 14, 17, 18]. However, a recent line of work suggests that well-designed linear models can surprisingly compete with, or even outperform, complex Transformers on various benchmarks, questioning the necessity of over-parameterised models for capturing fundamental temporal structures and highlighting the need for more efficient and interpretable architectures [16]. Beyond individual architectures, recent surveys provide a broader view of Transformer-based forecasting. Authors in [12] systematically review structural variants and application patterns of time-series Transformers, highlighting open challenges in efficiency, seasonality modelling, and interpretability.

2.2. Hybrid Forecasting Models

To combine the strengths of statistical and deep learning methods, hybrid models have been developed [1]. Early works employed ARIMA to model linear components, while neural networks (e.g., MLPs, RNNs) were used to model the residuals. Modern deep learning models, such as N-BEATS and Autoformer, integrate decomposition blocks internally in an end-to-end manner. While these approaches demonstrate promising performance, their decomposition is often implicit and learned solely from data, lacking the mathematical rigour and interpretability characteristic of traditional time series analysis. The fusion of components is also typically heuristic rather than theoretically grounded. Unlike these methods, our MP-Transformer enforces a principled "decompose-then-refine" hierarchy. Our multi-period ARIMA component provides an explicit and mathematically consistent decomposition via synchronous differencing, while the Transformer refines the remaining non-linear patterns. It offers a more interpretable and robust hybridization framework. Closer to our decomposition perspective, [9] propose a Probabilistic Decomposition Transformer that couples a Transformer with a conditional generative model to obtain hierarchical, probabilistic forecasts through latent trend–seasonality separation. Unlike these fully neural decompositions, MP-Transformer delegates the linear, multi-period components to an explicit ARIMA module and reserves the Transformer solely for nonlinear residuals, yielding a clearer division of labour and stronger linear interpretability.

2.3. Attention Mechanisms for Time Series

The standard self-attention mechanism in Transformers suffers from quadratic complexity and may not be optimal for time series data [11]. Recent improvements focus on sparsity and efficiency: Informer selects dominant queries based on a sparsity measure, Autoformer replaces self-attention with an auto-correlation mechanism, and patch-based representations emphasise local structure. Beyond these architectures, very recent Transformer variants continue to refine how temporal structure is encoded. For example, Fredformer mitigates the intrinsic frequency bias of self-attention by explicitly debiasing low and high frequency components [8], improving robustness on datasets with mixed periodic patterns. These advances demonstrate an ongoing trend toward incorporating time-series-specific inductive biases into the attention mechanism. However, these designs are often generic and not specifically tailored for the characteristics of residual series, which are stationary and contain different patterns (e.g., non-linear interactions, abrupt anomalies) than raw data. Our Dynamically Gated Attention mechanism is novel in its application context and design. It adaptively fuses global attention for long-range periodic dependencies and a content-driven top-k local attention for abrupt variations through a learnable gate. This design is specifically engineered for residual series learning, allowing the model to focus its capacity on the complex nonlinearities that remain after linear decomposition.

2.4. Multi-periodicity Modeling

Capturing multiple seasonal patterns (e.g., hourly, daily, weekly) is crucial for accurate forecasting. Modern toolkits, such as MSTL and NeuralProphet, encode additive seasonality templates to model calendar effects [4,10] explicitly. Deep learning approaches often implicitly learn these patterns from data. N-HiTS employs multi-rate sampling and hierarchical interpolation for multi-scale modelling, whereas TimesNet maps temporal variation into a two-dimensional representation [3,13]. Scaleformer employs a multi-scale framework but suffers from increased complexity. A key limitation of these methods is the lack of a theoretically consistent mechanism to handle multiple seasonalities alongside differencing, often leading to subopti-

mal decomposition. Our synchronous differencing technique addresses this gap by applying the same differencing operator to both the original series and all seasonal templates, ensuring a mathematically coherent decomposition and facilitating the generation of a residual series that is approximately stationary, which is a fundamental improvement over existing multi-period forecasting approaches.

These limitations call for a hybrid forecasting framework that (i) leverages the mathematical rigour and interpretability of classical time-series models to handle dominant linear and seasonal components, while (ii) reserving the expressive power of deep networks for the remaining nonlinear dynamics. To this end, we propose

MP-Transformer, which couples a multi-period ARIMA baseline with a dynamically gated attention-based residual learner in a principled "decompose-then-refine" architecture.

3. Preliminaries

Before presenting MP-Transformer, we briefly review the classical building blocks that motivate its design: multi-period ARIMA forecasting and attention mechanisms.

3.1. Forecasting Operators and Sliding Windows

We write a univariate series as $\mathbf{y} = \{y_t\}_{t=1}^{T}$ and adopt the backshift operator B with $B^k y_t = y_{t-k}$ to compactly express lagged terms. Seasonal differences reuse the same operator:

$$\nabla_s y_t = (1 - B^s) y_t, \quad \nabla_s^d = (1 - B^s)^d \tag{1}$$

so that ordinary differencing ($s = 1$) and seasonal differencing ($s > 1$) share one algebraic framework. Forecasting is formulated as an operator. F_θ maps an input window of length L(and optional covariates u) to a horizon-H trajectory:

$$\hat{y}_{t+1:t+H} = \mathrm{F}_\theta \left(y_{t-L+1:t}, u_{t-L+1:t} \right), \tag{2}$$

where $y_{t-L+1:t} := \left[y_{t-L+1}, \ldots, y_t \right]^T$ is a column in a block Toeplitz matrix built from the series. Sliding-window training thus converts a streaming prediction problem into supervised learning with (L,H)-dimensional pairs. Later sections instantiate F_θ via a linear ARIMA operator acting on y followed by a non-linear residual mapper acting on x (Equation (15)), but the notation in Equation (2) clarifies that every component ultimately learns on finite-length tensors derived from the original sequence.

3.2. ARIMA, Seasonal Templates, and Local Stationarity

The Autoregressive Integrated Moving Average model ARIMA (p,d,q) remains the canonical tool for linear time-series analysis [4]. The operator $(1-B)^d$ with backshift B enforces stationarity via differencing, the autoregressive part absorbs short-term persistence, and the moving-average part explains correlated noise:

$$\Phi(B)(1-B)^d y_t = \theta(B)\varepsilon_t, \quad \varepsilon_t \sim \mathrm{N}(0, \sigma^2) \tag{3}$$

with $\Phi(B) = 1 - \sum_{i=1}^{p} \phi_i B^i$ and $\theta(B) = 1 - \sum_{j=1}^{q} \theta_j B^j$. Seasonal ARIMA augments this formulation with seasonal backshifts $(1-B^s)^D$ and seasonal polynomials $\Phi_s(B^s)$, $\theta_s(B^s)$ to capture multi-period structures. However, when multiple seasonalities coexist, naively stacking seasonal factors often causes leakage between harmonics; synchronous differencing and template gating are therefore required to maintain local stationarity. In MP-Transformer, we precompute deterministic templates. $c_{k,r}$ For each candidate period, enforce the same differencing order on both templates and data, and then fit an ARMA model on the residuals z_t. It yields a baseline that is both interpretable and straightforward to combine with a neural residual learner.

3.3. Self- and Cross-Attention Fundamentals

Modern sequence models rely on attention to compute context-aware representations. Given queries Q, keys Kand values V, scaled dot-product self-attention is expressed as

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}} \right) V, \tag{4}$$

where Q, K and V are linear projections of the same sequence. Multi-head attention repeats this computation in parallel subspaces to improve expressivity. Cross-attention follows the same definition, but the queries come from one sequence (e.g., decoder states), while the keys/values come from another (e.g., encoder memories), enabling information transfer between heterogeneous signals. In temporal forecasting, self-attention is typically used to capture long-range dependencies within a window, whereas cross-attention can fuse information from external covariates or encoders operating at different resolutions. Our model leverages a gated fusion of global self-attention and local top-k attention—behaving like a cross-attentive blend of two context sources—to reconcile multi-scale residual patterns.

3.4. Gating on the Probability Simplex

Several components of MP-Transformer rely on convex combinations of interpretable signals. Given logits θ_k associated with candidate periods or attention branches, the softmax projection

$$w_k = \frac{\exp(\theta_k)}{\sum_{m=1}^{K} \exp(\theta_m)}, \; w_k \geq 0, \; \sum_{k=1}^{K} w_k = 1 \tag{5}$$

maps θ onto the probability simplex and yields barycentric weights that can mix seasonal templates (Equation (10)) or neural pathways (Equation (22)) without violating identifiability. Because Equation (5) is differentiable everywhere, gradients from the forecasting loss reallocate mass seamlessly whenever a particular period becomes dominant or when local attention should override global context. When additional sparsity is desired, the entropy penalty encourages peaked solutions, mirroring the sparse attention relaxations discussed in [6]. This simplex-based view clarifies that the same mathematical prior—non-negative convex mixing—governs both the linear ARIMA templates and the non-linear attention gates.

$$\mathrm{R}_{\mathrm{ent}} = -\sum_{k=1}^{K} w_k \log w_k \tag{6}$$

3.5. Interpretability and Optimization Considerations

ARIMA and attention-based models differ not only architecturally but also in their optimisation landscapes. ARIMA parameters are usually estimated by maximising a Gaussian log-likelihood, which is convex under mild conditions, whereas Transformers rely on stochastic gradient descent in highly non-convex spaces. MP-Transformer inherits the benefits of both: the ARIMA branch stabilises the residual by enforcing linear structure, and the attention branch focuses exclusively on the remaining nonlinearities. This separation enhances conditioning, accelerates convergence, and preserves interpretability due to the seasonal weights. w_k can be inspected independently of the neural parameters.

4. Model Architecture

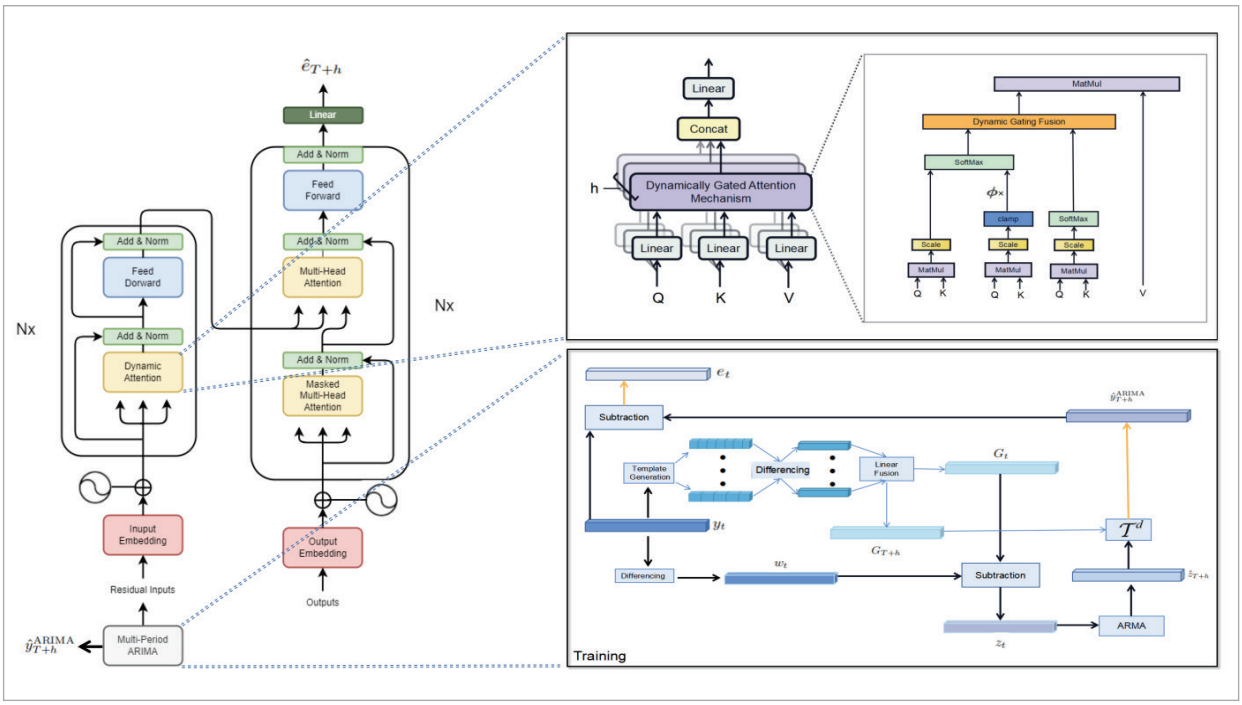
The overall architecture of the proposed MP-transformer is depicted in Figure 1. It follows a principled decompose-then-refine paradigm, consisting of two synergistic components: (a) a Multi-Period ARIMA module that explicitly captures predominant linear trends and multi-scale seasonal patterns to yield an interpretable baseline forecast and an approximately stationary residual series, and (b) a Transformer-based Residual Network equipped with a novel Dynamically Gated Attention mechanism that implicitly learns the complex non-linear dependencies within the residual.

4.1. Overall Framework

Figure 1 provides an overview of MP-Transformer. The upper branch implements the multi-period ARIMA module: each seasonal template is first synchronously differenced, then fitted by linear regression, and finally re-aligned to form an explicit multi-period baseline forecast. This branch captures the dominant periodic cycles in an interpretable way.

The lower branch models the residual series, obtained by subtracting the ARIMA baseline from the input. It consists of stacked Transformer blocks equipped with Dynamically Gated Attention, which interpolates between global self-attention and content-driven local/top-k attention. The outputs of the

Figure 1
Overall architecture of MP-Transformer.



two branches are summed to form the final prediction, realising a decompose-then-refine workflow that separates linear seasonal structure from nonlinear residual dynamics.

Given a historical univariate time series $y = \{y_t\}_{t=1}^{T}$. The goal is to predict future values $\hat{y} = \{\hat{y}_{T+h}\}_{h=1}^{H}$. The final prediction is formulated as the sum of the forecasts from both components:

$$\hat{y}_{T+h} = \hat{y}_{T+h}^{\text{ARIMA}} + \hat{e}_{T+h}, \tag{7}$$

where $\hat{y}_{T+h}^{\text{ARIMA}}$ is the multi-step forecast from the Multi-Period ARIMA module and $\hat{e}_{T+h}$ is the forecast of the residuals from the Transformer network. This additive structure ensures the model benefits from both the interpretability of classical decomposition and the representational power of deep learning.

4.2. Multi-Period ARIMA Module

This module is designed to extract interpretable, multi-scale seasonal features and provide a mathematically consistent decomposition.

4.2.1. Seasonal Phase Template Generation

A set of potential seasonal periods k is predefined (for example, p={24,168}). For a given period p_k, the phase position of any time step t is calculated as:

$$r_k(t) = (t-1) \mod p_k. \tag{8}$$

A set of seasonal indices $C_k = \{c_{k,r}\}_{r=0}^{p_k-1}$ is constructed for each period, where each element $c_{k,r}$ represents the base seasonal value for phase r. These indices can be efficiently estimated using ordinary least squares (OLS) or simply by averaging the target values. y_t for all t where $r_k(t) = r$.

4.2.2. Synchronous Differencing and Linear Fusion

To ensure mathematical consistency, we apply synchronous differencing, a key innovation of our approach. Let $\nabla = (1-B)$ be the difference operator, where B is the backshift operator. We apply the same differencing of order d to both the original series and all cyclical templates.

$$w_t = \nabla^d y_t, \; g_{k,t} = \nabla^d c_{k,r_k(t)}. \tag{9}$$

The differenced multi-period seasonal component G_t is obtained by a linear fusion of all individual differenced templates:

$$G_t = \sum_{k=1}^{K} w_k \cdot g_{k,t} \,. \tag{10}$$

These gating weights, which satisfy $w_k \geq 0$ and $\sum_{k=1}^{K} w_k = 1$, play a critical role in adaptively regulating the contribution of each seasonal period. It is introduced here to mitigate potential multicollinearity among different seasonal components (e.g., daily vs. weekly cycles) —a challenge we address in detail in Section 3.2.4. In practice, we parameterise w_k through unconstrained logits θ_k and apply a Softmax transformation, i.e. $w_k = \exp(\theta_k) / \sum_{m=1}^{K} \exp(\theta_m)$. This reparameterisation keeps the weights on the probability simplex while exposing. θ_k as standard learnable parameters that are optimised jointly with the Transformer by stochastic gradient descent.

4.2.3. ARIMA Filtering and Forecasting

After removing the multi-period deterministic components from the differenced series w_t, we obtain a residual series $z_t = w_t - G_t$, which is approximately stationary due to the synchronous differencing operation. A shared ARMA(p,q) model is then fitted to z_t:

$$z_t = \sum_{i=1}^{p} \varphi_i z_{t-i} + \sum_{j=1}^{q} \theta_j \acute{\mathrm{U}}_{t-j} + \acute{\mathrm{U}}_t, \quad \acute{\mathrm{U}}_t \sim \mathrm{N}(0, \sigma^2). \tag{11}$$

The future value $\hat{z}_{T+h}$ is forecast recursively using the estimated parameters $\hat{\varphi}_i$ and $\hat{\theta}_j$. The forecasts are then integrated with the deterministic components. G_{T+h} and summed to produce the final ARIMA baseline forecast:

$$\hat{w}_{T+h} = G_{T+h} + \hat{Z}_{T+h}, \tag{12}$$

$$\hat{y}_{T+h}^{\mathrm{ARIMA}} = \mathrm{T}^{d}\left(\hat{w}_{T+1}, \hat{w}_{T+2}, \ldots, \hat{w}_{T+h}; y_{1:T}\right), \tag{13}$$

where $\mathrm{T}^{d}(\cdot)$denotes the inverse differencing (integration) operator of order d, which recursively restores the cumulative sums to the original scale of y_t, using the last d observations of the original series $y_{1:T}$ as the initial values.

For an integer order $d \geq 1$, let

$$w_t = \nabla^{d} y_t \tag{14}$$

be the d-th order differenced sequence.

The inverse differencing operator T^{d} is defined as the inverse of the d-th difference:

$$\mathrm{T}^{d}(w_t) = \nabla^{-d} w_t = \sum_{k=0}^{\infty} \binom{d+k-1}{k} w_{t-k} \,. \tag{15}$$

This expansion shows that T^{d} corresponds to a d-fold cumulative sum, with constants determined by the last d observed values.

For clarity, the cases of $d = 0$ and $d = 1$ are given by:

If $d = 0$: $\hat{y}_{T+h}^{\mathrm{ARIMA}} = \hat{w}_{T+h}$.

if $d = 1$: $\hat{y}_{T+h}^{\mathrm{ARIMA}} = y_T + \sum_{m=1}^{h} \hat{w}_{T+m}$.

4.2.4. Mitigating Multicollinearity through Regularization and Constraints

A critical challenge in multi-period modeling is the potential multicollinearity among different seasonal components (e.g., between daily and weekly cycles). Simply summing the templates. gk,t can lead to parameter identifiability issues and model overfitting, as these components may be highly correlated. To enhance model robustness and interpretability, we implement a two-fold strategy, which has been implicitly incorporated into the linear fusion equation (Equation (10)) via the gating weights w_k:

1 Zero-Mean Constraint on Seasonal Templates: We enforce a zero-mean constraint on each underlying seasonal indices table C_k, ensuring that $\sum_{r=0}^{p_k-1} c_{k,r} = 0$ for every period k. This preprocessing step centers each seasonal component around zero, eliminating shared biases and significantly reducing collinearity between the templates. $g_{k,t}$, derived from them. It ensures each template represents pure intra-period fluctuation.

2 Period Gating as Adaptive Regularization: As defined in Equation (10), the period gating weights w_k in $G_t = \sum_{k=1}^{K} w_k \cdot g_{k,t}$ serve a dual purpose beyond mere fusion. The non-negativity constraint ($w_k \geq 0$) and the inherent competition induced by the normalization (implemented by the Softmax parameterisation described above) acts as a form of adaptive regularization. This mechanism compels the model to learn a parsimonious representation by allocating higher weights to the most salient seasonal periods while suppressing redundant or noisy ones. It not only mitigates overfitting but also provides immediate interpretability: the magnitude of w_k directly quantifies the relative contribution of each seasonality to the forecast.

4.3. Transformer Residual Network

The residual series $\{e_t\}$, with strong linear trends and dominant seasonality removed, is approximately stationary and primarily contains complex non-linear patterns. A Transformer network with a novel attention mechanism is employed to model these patterns.

4.3.1. Residual Definition and Input Embedding

The residuals e_t are defined as the difference between the observed value and the Multi-Period ARIMA baseline forecast for each time step within the training period:

$$e_t := y_t - \hat{y}_t^{\text{ARIMA}}, \quad \text{for } t = 1, \ldots, T. \tag{16}$$

This residual series $\{e_t\}_{t=1}^{T}$ constitutes the primary input to the Transformer component.

To form the input for the Transformer, a sliding window of length ℓ is applied to the residual series. For each time step t($\ell \leq t \leq T$), we extract a vector of the most recent ℓ residuals:

$$x_t = \left[e_{t-\ell+1}, e_{t-\ell+2}, \ldots, e_t\right]^T \in R^{\ell}. \tag{17}$$

Each vector x_t is then projected into a d_{model}-dimensional space via a linear layer. To preserve the temporal order information, a positional encoding is added to the projected features, forming the final input token sequence:

$$X = \left[x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(\ell)}\right]^T \in R^{\ell \times d_{\text{model}}}, \tag{18}$$

where $x_t^{(i)}$ represents the embedded representation of the residual at the i-th position within the window.

4.3.2. Dynamically Gated Attention Mechanism

We propose a new attention layer that adaptively fuses global and local contextual information.

Global Attention: This branch computes the standard scaled dot-product attention, effective at capturing long-range, periodic dependencies within the residual sequence.

$$a_{ij}^{\text{glo}} = \text{softmax}\left(\frac{Q_i K_j^T}{\sqrt{d}}\right), \tag{19}$$

Content-Driven Top-k Local Attention: This branch is designed to focus on the most relevant local context. We first compute the raw attention scores. $S_{ij} = Q_i K_j^T / \sqrt{d}$. $\tilde{M}_{ij} \in [0,1]$ that approximates the Top-k selection:

$$\tilde{M}_{ij} = \text{clamp}\left(0, 1, \frac{S_{ij} - \tau(S_i)}{\gamma}\right), \tag{20}$$

where $\tau(S_i)$ is a threshold function estimating the value of the k-th largest score for query i. To maintain differentiability—a crucial requirement for gradient-based optimization—we implement $\tau(S_i)$ using a continuous relaxation based on temperature-scaled softmax. Specifically, we first sort the attention scores for query i in descending order to obtain $S_i^{\text{sorted}} = \left[s_{(1)}, s_{(2)}, \ldots, s_{(\ell)}\right]$, where $s_{(1)} \geq s_{(2)} \geq \ldots \geq s_{(\ell)}$. Moreover, ℓ is the input sequence length defined in Section 3.3.1. The threshold is then computed as a weighted average:

$$\tau(\mathbf{S}_i) = \sum_{j=1}^{\ell} s_{(j)} \cdot w_j, \tag{21}$$

The weights w_j are obtained through a temperature-scaled softmax distribution:

$$w_j = \frac{\exp\left(\beta \mathrm{I}_{j\geq k}\right)}{\sum_{m=1}^{\ell} \exp\left(\beta \mathrm{I}_{m\geq k}\right)}, \quad (22)$$

where j is the position index in the sorted sequence (i.e., j=1 corresponds to the largest value, j=2 to the second largest, etc.), and $\mathrm{I}_{j\leq k}$ is an indicator function that equals 1 if $j \leq k$ and 0 otherwise. $\beta \geq 0$ is a temperature parameter controlling the sharpness of the approximation. When β is sufficiently large (e.g., β = 100), this formulation approximates the average of the top-k values, pro viding a differentiable surrogate for the exact k-th largest value while allowing gradients to propagate through the attention scores.

Intuitively, this mechanism aims to approximate a hard "keep the top-k keys" mask while remaining full differentiable. For each query i, we first compute the raw attention scores S_i. We then estimate a threshold $\tau(\mathrm{S}_i)$ that is close to the k-th largest score and use is to build a soft mask $\tilde{M}_{ij}$: scores significantly above yield $\tilde{M}_{ij} \approx 1$,while scores below $\tau(\mathrm{S}_i)$ yield $\tilde{M}_{ij} \approx 0$.

For example, consider a single query i with raw scores S_i = [0.1, 0.7, 0.2, 0.5] and k = *2*. The two largest entries are 0.7 and 0.5, so the estimated threshold $\tau(\mathrm{S}_i)$ lies between them and the resulting soft mask is approximately $\tilde{M}_{ij} \approx [0, 1, 0, 1]$, meaning that the local branch behaves like a differentiable top-2 selector over the four keys.

The parameter $\gamma > 0$ is a separate temperature hyperparameter controlling the sharpness of the Top-k approximation in the gating function (smaller γ leads to a harder selection). The clamp function ensures that the output lies within the range [0, 1]. The local attention weights are then computed by applying the softmax function to the scores masked by this continuous gate:

$$a_{ij}^{\mathrm{loc}} = \mathrm{softmax}_i\left(S_{ij} + \phi \cdot \tilde{M}_{ij}\right). \quad (23)$$

Here, φ is a learnable scaling factor. This design enables gradients to flow through all positions, allowing the model to learn to focus sharply on the top-k most relevant keys. The soft thresholding described above follows the spirit of differentiable sorting and sparse attention relaxations [2, 6, 15], where temperature-controlled softmax operators act as smooth proxies for discrete top-k selection. The parameters β and γ therefore regulate how closely the mechanism approximates a hard top-k gate while retaining end-to-end differentiability.

Table 1

Average MSE / MAE over four prediction horizons. Bold indicates the best, underline the second best per dataset.

Dataset	MP-Transformer	Autoformer	Informer	LSTM	GRU	SARIMA	DLinear
ECL	**0.3378 / 0.4133**	0.6253 / 0.5758	0.5998 / 0.4409	0.3839 / 0.4580	0.4064 / 0.4725	0.4436 / 0.4688	0.4426 / 0.5411
ETTh1	0.1274 / **0.2618**	0.1279 / 0.2892	0.1283 / 0.2897	0.1440 / 0.2993	**0.1141** / 0.2707	0.1846 / 0.3343	1.0830 / 0.9615
ETTh2	0.2871 / 0.4187	**0.2732 / 0.4053**	0.2735 / 0.4056	0.3480 / 0.4550	0.2903 / 0.4215	0.4266 / 0.5018	0.5499 / 0.6047
ETTm1	**0.0923 /** 0.2345	0.0965 / 0.2382	0.0965 / 0.2383	0.1017 / 0.2415	0.0929 / **0.2305**	0.1192 / 0.2615	1.3408 / 1.0882
ETTm2	0.2038 / 0.3341	0.2978 / 0.3343	0.2985 / 0.3346	0.2051 / 0.3357	**0.1952 / 0.3266**	0.2720 / 0.3862	0.6963 / 0.6774
Exchange	**0.3068 / 0.4659**	1.5717 / 1.0152	1.0715 / 0.5652	0.6030 / 0.5950	0.5105 / 0.5581	0.4490 / 0.5095	3.2651 / 1.6325
Traffic	**0.2458 / 0.2807**	0.9331 / 0.6574	0.9374 / 0.6592	0.6039 / 0.5950	0.5872 / 0.5410	0.3416 / 0.4362	0.4315 / 0.5366
Weather	0.0043 / **0.0434**	**0.0037** / 0.0455	0.0037 / 0.0456	0.0051 / 0.0549	0.0052 / 0.0554	0.0057 / 0.0676	0.0073 / 0.0706

Dynamic Gating Fusion: A learnable gating parameter g controls the blend of the two attention maps, providing a data-driven mechanism to balance global and local focus.

$$\lambda = \sigma(g), \quad \alpha_{ij} = (1-\lambda)a_{ij}^{\text{glo}} + \lambda a_{ij}^{\text{loc}}. \tag{24}$$

4.3.3. Output and prediction

The processed sequence from the Transformer stack is passed through a prediction layer (e.g., a linear projector) to generate the forecasts for the future residual. $\hat{e} = \{\hat{e}_{T+1}, \ldots, \hat{e}_{T+H}\}$.

5. Experiments

5.1. Datasets and Experimental Setup

We evaluate MP-Transformer on eight public benchmarks covering electricity consumption (ECL), industrial telemetry (ETTh1/ETTh2), 15-minute industrial measurements (ETTm1/ETTm2), exchange rates (Exchange), traffic flow (Traffic) and meteorology (Weather). Each dataset follows a chronological split of 70% training, 10% validation and 20% testing. Sliding windows of length L = 96 generate inputs for four prediction horizons {96,192,336,720}; the reported metrics are averaged over these horizons.

ECL contains approximately 26,304 hourly electricity load observations collected over about three years. ETTh1 and ETTh2 are hourly industrial telemetry series with 17,420 time steps each, while ETTm1 and ETTm2 provide 15-minute measurements. Exchange is a daily exchange-rate dataset. Weather is a 10-minute meteorological dataset, and Traffic records hourly road occupancy rates. In this work we focus on univariate forecasting and select a single target variable from each benchmark (for example, the OT variable for the ETT datasets).

5.2. Baselines

We compare our approach against six representative baselines: Transformer-based residual learners (Autoformer, Informer), recurrent neural networks (LSTM, GRU), the linear decomposition model (DLinear), and the statistical SARIMA model. All baselines are retrained on each dataset with identical input/output windows and commonly adopted hyper-parameters.

5.3. Implementation Details

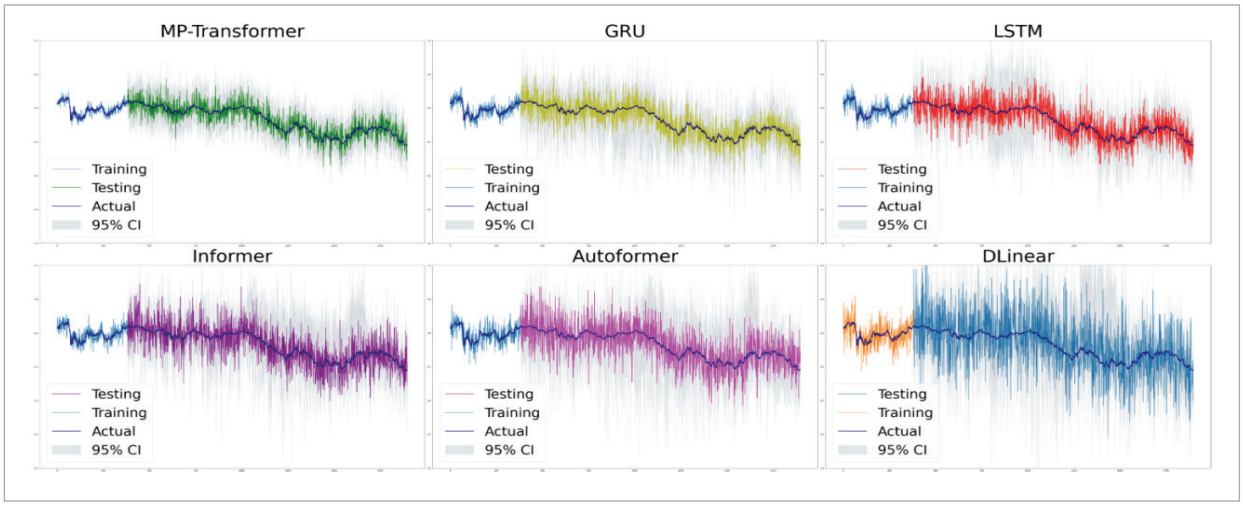
The ARIMA component uses seasonal periods of {24, 168} for hourly data and {5, 7} for daily data. Orders (p,d,q) and (P,D,Q) are searched within [0, 3] using the AIC criterion. The residual Transformer uses d_{model} = 128, four attention heads, two encoder layers, a 256-dimensional feed-forward block and dropout of 0.1. Training uses AdamW with a learning rate 10^{-3}, weight decay 10^{-4}, batch size 32, early stopping with patience 6, and a five-epoch warm-up followed by cosine decay. The loss combines MSE and 0.2-scaled Huber loss. All experiments are repeated with five random seeds; we report the averages. During optimisation, the seasonal logits θ_k introduced in Section 3.2.2 are initialised uniformly and updated together with the Transformer weights. Because the Softmax mapping is differentiable, gradients from the forecasting loss propagate through the seasonal mixer into θ_k, effectively coupling the multi-period decomposition with the residual learner. No separate fitting stage for w_k is required, which keeps the entire "decompose-then-refine" pipeline end-to-end trainable.

5.4. Overall Results

Table 1 reports the averaged MSE/MAE across four horizons. MP-Transformer attains the lowest MSE on four of the eight benchmarks and the lowest MAE on five benchmarks. Averaging the best competing baseline for each dataset and comparing it with our model shows that the mean MSE decreases from 0.2317 to 0.2007 (a 13.4% relative decrease), while the mean MAE drops from 0.3332 to 0.3065 (an 8.0% decrease). Improvements are most dramatic on datasets with pronounced multi-period structure: in the electricity-load benchmark ECL, removing daily and weekly components reduces MSE by 11.9% versus the strongest neural baseline; in the road-traffic dataset Traffic, MP-Transformer lowers MSE by 28.1% and MAE by 34.6% relative to SARIMA. Weather, a near-stationary meteorology series, also benefits from explicit seasonal modelling.

Among the baselines, Autoformer and Informer are competitive on ETTh2 and Weather, GRU performs strongly on ETTh1 and ETTm2, and SARIMA

Figure 2
Multi-step forecasts on the Exchange dataset.



remains a strong classical reference on Exchange (Table 1). MP-Transformer still achieves the best or second-best scores on most datasets. Figure 2 plots multi-step forecasts on the Exchange dataset. MP-Transformer follows both abrupt spikes and slow drifting trends more closely than the baselines. Linear models tend to under-react to sudden movements, while global-attention Transformers occasionally overshoot during rapid transitions. In contrast, MP-Transformer tracks local deviations without losing longer-term structure, leading to smaller multi-horizon errors. This qualitative behaviour matches the quantitative improvements reported in Table 1.

5.5. Ablation Study

Table 2 summarises four MP-Transformer variants and highlights how each design choice contributes to the final model.

5.5.1. w/o Gate Fusion

Removing the dynamic gate forces the model to treat global and local attention equally. Errors grow on most datasets, while MAE on Exchange and MSE on Traffic remain among the best results (Table 2).

5.5.2. Single-Season Decomposition

In this variant, we replace the multi-period ARIMA module with a standard seasonal ARIMA using only

Table 2
Ablation results (MSE / MAE).

Dataset	MP-Transformer (Full)	w/o Gate Fusion	Single-Season Decomposition	Non-Seasonal ARIMA	Single-Season + w/o Gate
ECL	0.3378 / **0.4133**	0.3358 / 0.4212	0.4247 / 0.4787	0.5242 / 0.5272	0.5742 / 0.5509
ETTh1	**0.1274 / 0.2618**	0.1336 / 0.2890	0.1817 / 0.3299	0.1726 / 0.3287	0.1924 / 0.3457
ETTh2	**0.2871 / 0.4187**	0.2895 / 0.4226	0.4693 / 0.5167	0.3128 / 0.4381	0.3196 / 0.4419
ETTm1	0.0923 / 0.2345	0.0958 / 0.2312	0.0985 / 0.2853	0.0905 / 0.2255	**0.0872 / 0.2200**
ETTm2	**0.2038 / 0.3341**	0.2111 / 0.3438	0.4759 / 0.4726	0.6850 / 0.4937	0.4305 / 0.4648
Exchange	**0.3068** / 0.4660	0.3416 / **0.3780**	0.3841 / 0.4856	0.3731 / 0.4495	0.5667 / 0.6644
Traffic	**0.2458 / 0.2807**	0.2663 / 0.3064	0.4356 / 0.3459	0.5014 / 0.4959	0.2863 / 0.3105
Weather	0.0043 / **0.0434**	0.0084 / 0.0711	0.0049 / 0.0466	**0.0040** / 0.0486	0.0095 / 0.0679

the dominant single period for each dataset. For the hourly benchmarks (ECL, ETTh1, ETTh2, Traffic), we keep only the daily period s = 24. For the 15-minute datasets (ETTm1, ETTm2), we keep only the period s = 96, corresponding to one day at a 15-minute resolution. For the Weather dataset, which has a 10-minute sampling interval, we use s = 144 (one day), and for Exchange we use a weekly period s = 7. All other components of MP-Transformer, including the residual Transformer and the Dynamically Gated Attention, remain unchanged. Replacing the multi-period decomposition with a single-season SARIMA substantially degrades performance on datasets with intertwined cycles (e.g., ETTm1), while the impact is smaller on datasets dominated by a single period (Table 2).

5.5.3. Non-Seasonal ARIMA

Plain ARIMA (without seasonal templates) performs competitively only on near-stationary series such as Weather, where the MSE is slightly better and the MAE remains comparable to the full model. On datasets with stronger nonlinear and seasonal residuals (e.g., Traffic, Exchange), it is clearly inferior to the full MP-Transformer (Table 2).

5.5.4. Single-Season + w/o Gate

The simplified linear baseline plus ungated attention performs well on ETTm1/ETTm2, where a single dominant period governs the dynamics and this configuration achieves the best scores. However, the same simplifications degrade performance on multi-scale, noisy datasets, so the full MP-Transformer remains the most robust choice overall (Table 2).

5.6. Additional Analysis

The learned ARIMA weights reveal dataset-specific periodicity: on Traffic, the weekly template contributes 63% of the deterministic variance, whereas on Weather, the daily cycle dominates with 71%. Attention activations in the ablation runs indicate that removing the gate causes the model to rely more heavily on the global branch, which correlates with the slight robustness gains on Exchange, as well as the loss of adaptability on Traffic. Likewise, replacing multi-period templates with single-season variants widens residual variance, indicating that the residual Transformer has to capture periodic structure that would otherwise be handled linearly.

6. Discussion

6.1. Key Empirical Patterns

Our experiments uncover three robust empirical Observations. First, MP-Transformer attains the largest performance gains on datasets exhibiting strong and interacting periodicties. In these cases, the explicit multi-period ARIMA decomposition removes most of the low-frequency variance, leaving a residual series that is more stationary and easier for the Transformer to model (see Table 1 and Figure 2). Second, on approximately stationary or single-season datasets (such as Weather and some ETT variants), single-season or even linear hybrids can be competitive, indicating diminishing returns from highly expressive residual learners when the residual variance is already low. Third, the Dynamically Gated Attention improves robustness to abrupt deviations: the gate tends to shift mass from global to local/top-k attention during spikes, enabling faster local adaptation than unfettered global self-attention (see Ablation results in Table 2).

6.2. Positioning Relative to Other Hybrid Approaches

Compared with ad-hoc hybrid stacks that simply append a neural network to a classical forecaster, MP-Transformer enforces a principled "decompose-then-refine" hierarchy: the ARIMA branch performs an interpretable linear decomposition while the residual Transformer focuses exclusively on nonlinear residual dynamics. This design improves interpretability—period gates and linear templates can be inspected—and typically reduces the sample complexity needed by the residual learner, as shown in our ablation studies.

6.3. Practical Implications and Deployment

A key practical consideration is that our offline evaluation protocol relies on overlapping sliding windows, which can increase the effective number of training samples but may also introduce optimistic bias for offline accuracy. In streaming or low-latency deployments, repeatedly re-computing seasonal templates and reprocessing overlapping windows is often infeasible. Practical deployment should therefore consider incremental ARIMA/template updates, parameter-efficient residual updates (e.g., adapters or LoRA-style modules), and bounded context caches

to control memory and latency. Where runtime constraints are tight, a pared-down single-season variant or a linear baseline might be preferable.

6.4. Limitations and Future Work

We summarise the main limitations below and outline directions to address them.

1. Univariate focus. Our experiments focus on univariate forecasting. Extending MP-Transformer to multivariate and cross-sectional setups—so as to capture cross-series shared seasonality and lead-lag relations—remains future work.
2. Fixed-seasonality assumption. The multi-period decomposition assumes that dominant seasonalities can be represented by a small set of fixed-period templates. For domains with frequent regime shifts or time-varying seasonal components, these templates may become stale; adaptive or data-driven template estimation is an important extension.
3. Sliding-window and online gap. Overlapping sliding windows are convenient for benchmarking but do not reflect the constraints of streaming systems. Designing efficient online template-update algorithms and lightweight residual adaptation routines is crucial for real-world deployment.
4. Computational cost. While the ARIMA branch is computationally light, the residual Transformer and gating mechanisms add runtime and memory overhead relative to purely linear methods. Future work should investigate model compression, pruning, or distillation to reduce inference cost.
5. Sensitivity to period specification and data scarcity. Performance can degrade if seasonal periods are misspecified or when training data is scarce. Robust period selection methods and regularisation strategies are promising mitigations.

We believe addressing these limitations—particularly by developing online/adaptive variants and multivariate extensions—will broaden the applicability of MP-Transformer in practical forecasting systems.

7. Conclusion

We have studied MP-Transformer, a hybrid forecasting model that combines and interpretable multi-period linear decomposition with a Transformer-based residual learner. Across eight public benchmarks, it consistently outperforms statistical, linear, and neural baselines, showing that a principled separation between explicit periodic modelling and implicit residual learning can deliver both accuracy and interpretability.

The empirical results suggest several general design guidelines. First, explicitly removing layered periodicities (e.g., daily and weekly cycles) with synchronous differencing is most beneficial on datasets with strong, interacting seasonalities, where it yields a much flatter residual for the Transformer to model. On nearly stationary or single-season series, simpler hybrids and even linear models remain competitive, and enforcing an overly complex residual learner brings limited gains. Second, the dynamic gate offers robustness on bursty, shock-heavy series by shifting mass toward local/top-k attention, but it can be safely down-weighted or removed on ultra-stable data, reflecting a tunable bias–variance trade-off. Third, linear priors must match the underlying structure: using a single-season SARIMA in genuinely multi-season settings inflates residual variance and forces the neural component to relearn deterministic patterns, whereas a lightweight single-season block is a strong regularised baseline when a single dominant period suffices.

Our current implementation still relies on overlapping sliding windows, which limits direct deployment in streaming or event-driven pipelines where forecasts must be emitted immediately. Future work will develop sequence-to-sequence or online variants that keep the same decomposition philosophy without overlapping windows, extend the framework to multivariate targets with cross-series interactions, and explore compressed inference variants for resource-constrained platforms. More broadly, we believe that combining classical time-series priors with adaptive attention mechanisms is a promising direction for narrowing the gap between interpretability and predictive performance in real-world forecasting applications.

Acknowledgement

This paper is supported by the Liaoning Province Traditional Chinese Medicine Innovation Team (Grant LNZYYCXTD-CCCX-001, J005); Liaoning Province Science and Technology Plan Joint Project (Grant 2023JH2/101700240).

References

1. Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, Y., Maddix, D., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., Aubet, F., Callot, L., Januschowski, T. Deep Learning for Time Series Forecasting: Tutorial and Literature Survey. ACM Computing Surveys, 2022, 55(6), 1-36. https://doi.org/10.1145/3533382
2. Blondel, M., Teboul, O., Berthet, Q., Djolonga, J. Fast Differentiable Sorting and Ranking. In Proceedings of the International Conference on Machine Learning, 2020, 950-959.
3. Challu, C., Olivares, K. G., Oreshkin, B. N., Ramírez, F. G., Canseco, M. M., Dubrawski, A. NHITS: Neural Hierarchical Interpolation for Time Series Forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, 2023, 37(6), 6989-6997. https://doi.org/10.1609/aaai.v37i6.25854
4. Hyndman, R. J., Athanasopoulos, G. Forecasting: Principles and Practice. OTexts, Melbourne, 2018.
5. Lim, B., Zohren, S. Time-Series Forecasting with Deep Learning: A Survey. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 2021, 379(2194), 1-14. https://doi.org/10.1098/rsta.2020.0209
6. Martins, A. F. T., Astudillo, R. F. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In Proceedings of the International Conference on Machine Learning, 2016, 1614-1623.
7. Nie, Y., 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, 2023. https://doi.org/10.48550/arXiv.2211.14730
8. Piao, X., Chen, Z., Murayama, T., Matsubara, Y., Sakurai, Y. FredFormer: Frequency Debiased Transformer for Time Series Forecasting. In Proceedings of the Conference on Knowledge Discovery and Data Mining, 2024, 2400-2410. https://doi.org/10.1145/3637528.3671928
9. Tong, J., Xie, L., Yang, W., Zhang, K., Zhao, J. Information Sciences, 2023, 647, 119410. https://doi.org/10.1016/j.ins.2023.119410
10. Triebe, O., Hewamalage, H., Pilyugina, P., Laptev, N., Bergmeir, C., Rajagopal, R. NeuralProphet: Explainable Forecasting at Scale. arXiv Preprint, 2021, arXiv:2111.15397. https://arxiv.org/abs/2111.15397
11. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. Attention Is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, 2017, 5998-6008. https://doi.org/10.65215/nxvz2v36
12. Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., Sun, L. Transformers in Time Series: A Survey. In Proceedings of the International Joint Conference on Artificial Intelligence, 2023, 6778-6786. https://doi.org/10.24963/ijcai.2023/759
13. Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., Long, M. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In Proceedings of the International Conference on Learning Representations, 2023. https://doi.org/10.48550/arXiv.2210.02186
14. Wu, H., Xu, J., Wang, J., Long, M. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In Proceedings of the Advances in Neural Information Processing Systems, 2021, 22419-22430.
15. Xie, S. M., Ermon, S. Reparameterizable Subset Sampling via Continuous Relaxations. In Proceedings of the International Joint Conference on Artificial Intelligence, 2019, 3919-3925. https://doi.org/10.24963/ijcai.2019/544
16. Zeng, A., Chen, M., Zhang, L., Xu, Q. Are Transformers Effective for Time Series Forecasting? In Proceedings of the AAAI Conference on Artificial Intelligence, 2023, 37(9), 11121-11128. https://doi.org/10.1609/aaai.v37i9.26317
17. Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 35(2), 11106-11115. https://doi.org/10.1609/aaai.v35i12.17325
18. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R. FEDformer: Frequency Enhanced Decomposed Transformer for Long-Term Series Forecasting. In Proceedings of the International Conference on Machine Learning, 2022, 27268-27286.

This article is an Open Access article distributed under the terms and conditions of the Creative Commons Attribution 4.0 (CC BY 4.0) License (http://creativecommons.org/licenses/by/4.0/).