GraFT: Infusing Pre-trained Transformers with Relational Structure for Time Series Forecasting

Yuqi Yuan[1,2,3], **Xiong Luo**[1,2,3*], **Qiaojuan Peng**[1,2,3], **Wenbing Zhao**[4]
[1]School of Computer and Communication Engineering, University of Science and Technology Beijing
[2]Shunde Innovation School, University of Science and Technology Beijing
[3]Beijing Key Laboratory of Knowledge Engineering for Materials Science
[4]Department of Electrical Engineering and Computer Science, Cleveland State University
{D202310417, xluo, D202210397}@xs.ustb.edu.cn, w.zhao1@csuohio.edu

Abstract

Large Language Models (LLMs) have recently emerged as a leading approach for multivariate time series forecasting. However, their effectiveness is hampered by a fundamental architectural mismatch: the permutation-invariant self-attention of Transformers lacks inductive biases for the strict temporal order and complex cross-variable dependencies inherent in time series. Existing methods often sidestep this issue with input-level alignment techniques rather than endowing the model itself with structural awareness. To address this gap, we introduce GraFT (Graph-infused Forecasting Transformer), a framework that systematically embeds relational priors into a pre-trained backbone by constructing a heterogeneous patch relation graph, which represents both universal temporal principles with static edges and instance-specific patterns with dynamic adaptive edges. To process this multi-relational structure, a relational graph convolutional network generates structure-aware representations, which are infused into the patch embeddings to provide explicit structural guidance to the Transformer's attention mechanism. Extensive experiments show that GraFT achieves state-of-the-art performance on long-term forecasting and zero-shot learning, outperforming leading LLM-based methods on eight standard benchmarks with an average Mean Squared Error (MSE) reduction of 14.4%.

Code — https://github.com/yuanyumi/GraFT

Introduction

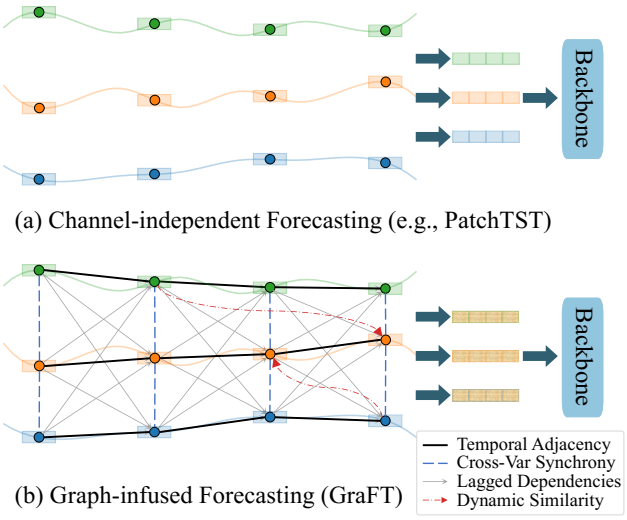
Multivariate time series forecasting (MTSF) is a fundamental task whose importance is demonstrated by its broad applications in domains ranging from weather forecasting (Angryk et al. 2020) and energy prediction (Demirel et al. 2012) to financial modeling (Patton 2013; Niu et al. 2020), a significance amplified by the recent data deluge from burgeoning fields like urban traffic (Xiao et al. 2022; Miao et al. 2024) and environmental sensing (Liu et al. 2025, 2022a).

In the pursuit of higher accuracy, the field has witnessed an evolution of deep learning architectures, progressing from Multilayer Perceptrons (MLPs) (Zeng et al. 2023; Das et al. 2023) and Convolutional Neural Networks (CNNs) (Wu et al. 2023) to more recent State Space Models (SSMs) (Gu and Dao 2024; Wu et al. 2025). Although these approaches yielded significant advancements, they consistently faced a foundational limitation in capturing long-range dependencies, a challenge the Transformer architecture (Vaswani et al. 2017) was specifically engineered to address. By introducing a self-attention mechanism capable of modeling global correlations across the entire sequence, Transformers provided a breakthrough solution, rapidly establishing them as the dominant approach in the field (Zhou et al. 2021; Nie et al. 2023).

However, a fundamental architectural mismatch arises when applying Transformers to MTSF. The model's core mechanism, permutation-invariant self-attention, is inherently at odds with the structured nature of the data, which is characterized by both a strict temporal order and a complex cross-variable topology. To mitigate this mismatch, research has advanced along two primary fronts. To enforce temporal order, efforts have focused on refining the attention mechanism with inductive biases like sparsity or auto-correlation (Zhou et al. 2021; Wu et al. 2021). Concurrently, attempts to model the cross-variable topology have diverged into two main streams of work, with one centering on intricate channel-mixing schemes (Zhang and Yan 2023) and the other exploring channel-independent models (Nie et al. 2023). The very emergence of these specialized and distinct approaches confirms that adapting the Transformer's core to the specific structures of time series constitutes a significant and active research frontier.

The recent advent of Large Language Models (LLMs), representing a significant scaling of the Transformer architecture, has introduced a new paradigm to MTSF that, rather than directly re-engineering the model's core, often circumvents this architectural mismatch through a strategy of peripheral alignment. This strategy aims to render time series data intelligible to a pre-trained model without altering its core architecture, achieved through techniques such as reprogramming numerical sequences into elaborate textual prompts (Jin et al. 2024), aligning patch embeddings with a model's semantic vocabulary (Pan et al. 2024), or using external graph neural networks to process the logical structure of a task prompt (Hu et al. 2025). Although innovative, these techniques all operate at the model's periphery, while the model's core insensitivity to temporal order and cross-variable topology remains.

*Corresponding author



(a) Channel-independent Forecasting (e.g., PatchTST)

(b) Graph-infused Forecasting (GraFT)

Figure 1: Conceptual comparison of forecasting paradigms. (a) The channel-independent approach processes variables in isolation, while (b) our GraFT framework uses a heterogeneous graph to fuse patch representations into a structure-aware state.

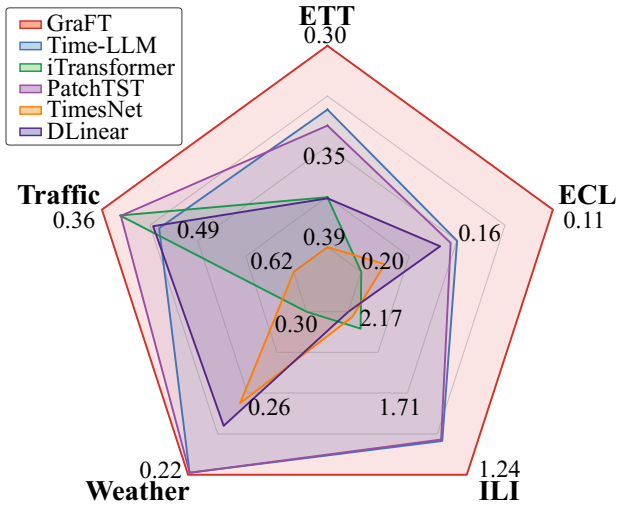


Figure 2: Comparison of model performance in terms of Mean Squared Error (MSE). For visualization, axes are inverted such that a larger enclosed area indicates better performance.

To confront this architectural challenge directly, we introduce **GraFT** (**Gra**ph-infused **F**orecasting **T**ransformer), a framework designed to systematically embed structural priors into a pre-trained Transformer backbone. As conceptualized in Figure 1, in contrast to the channel-independent paradigm that processes variables in isolation, GraFT explicitly models the rich inter-dependencies between patches from different variables and time steps by constructing a **Heterogeneous Patch Relation Graph** (**HPRG**) over the input. This novel data structure is designed to encode a rich spectrum of relational knowledge through a dual-mechanism: static edges represent universal principles like temporal continuity and synchrony, while dynamic edges adaptively capture instance-specific pattern similarity. To leverage this multi-relational structure, a Relational Graph Convolutional Network (R-GCN) (Schlichtkrull et al. 2018) generates structure-aware representations that explicitly guide the Transformer's attention mechanism. The powerful outcome of this structure-aware guidance is visualized in Figure 2, where GraFT demonstrates a consistent and significant performance advantage across multiple benchmarks. Our contributions are as follows:

- We propose a new forecasting paradigm designed to mitigate the core architectural mismatch between the permutation-invariant attention of Transformers and the ordered, multi-variable structure of time series. Our approach systematically infuses a pre-trained backbone with explicit relational priors, representing a fundamental shift from adapting data for the model to adapting the model for structured data.
- We introduce the HPRG to generate structure-aware representations by unifying static edges that encode universal temporal priors with dynamic edges that capture instance-specific patterns.
- Our extensive experiments demonstrate that GraFT achieves state-of-the-art performance on eight standard long-term forecasting benchmarks and exhibits exceptional zero-shot transfer learning capabilities.

Related Work

Multivariate Time Series Forecasting

In MTSF, a primary challenge lies in modeling inter-variable dependencies. One dominant line of work attempts to capture these interactions through channel-mixing. Early attention-based Transformers approached this by employing global queries for cross-channel communication (Zhou et al. 2021; Wu et al. 2021). Subsequent research explored alternative mixing strategies, using MLPs to process both inter-variable and intra-variable patterns (Wang et al. 2024, 2025) or operating in the frequency domain with transformation techniques (Wu et al. 2023). A distinct thread explicitly models the relational structure from the outset using graph neural networks (Gao et al. 2022; Mourya et al. 2024). In stark contrast, an influential alternative, channel-independence, has demonstrated remarkable success by processing each variable as a separate sequence (Nie et al. 2023; Zeng et al. 2023). The success of this simplified paradigm suggests that conventional channel-mixing mechanisms may be insufficient for effective multivariate modeling.

Encoding Temporal Structures for Transformers

The Transformer architecture (Vaswani et al. 2017), despite its power, presents a fundamental challenge with its inherent permutation-invariance, which standard positional encodings only partially address (Wen et al. 2023). Consequently,

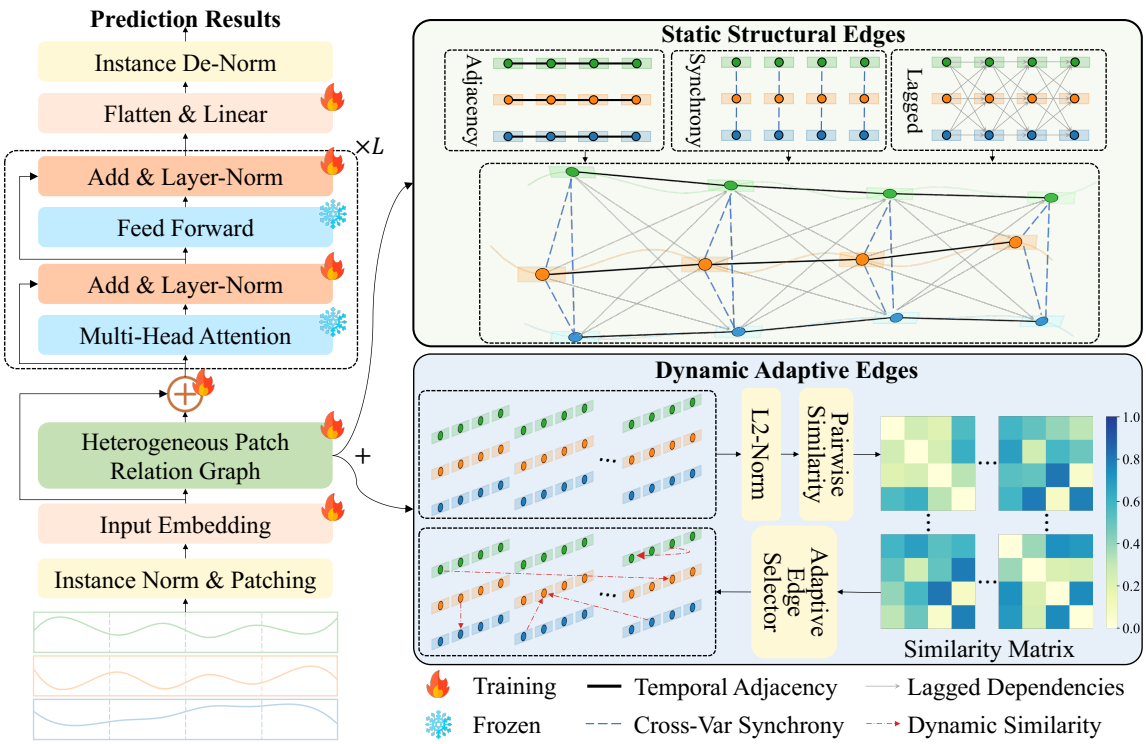


Figure 3: Overall Framework of GraFT.

a significant body of research has focused on embedding stronger temporal inductive biases. One direction refines the attention mechanism itself, either by introducing sparsity to reflect the locality of time series (Zhou et al. 2021), replacing attention with an auto-correlation mechanism (Wu et al. 2021), or performing frequency-domain analysis (Zhou et al. 2022). Another direction imposes explicit architectural priors, designing hierarchical representations to capture multi-scale patterns (Liu et al. 2022b) or disentangling processing into distinct stages (Zhang and Yan 2023). Moving beyond architectural modifications, more foundational approaches tackle core data properties directly, through mechanisms like de-stationary attention (Liu et al. 2022c) or by learning dynamical system operators (Liu et al. 2023). In this work, we propose a unified heterogeneous graph to simultaneously model both inter-variable relationships and the temporal dependencies inherent to time series within a Transformer backbone.

Methodology

Given a multivariate time series $\mathbf{X} \in \mathbb{R}^{L \times M}$ with M variables over a look-back window of length L, our goal is to predict the subsequent H time steps, denoted as $\hat{\mathbf{Y}} \in \mathbb{R}^{H \times M}$. Our proposed GraFT framework addresses the limitations of standard Transformers by systematically injecting structural inductive biases. As illustrated in Figure 3, the framework first tokenizes the input series into patch embeddings. These embeddings serve as nodes in a HPRG, which is composed of both universal static structural edges and dynamic adaptive edges. Static edges encode fundamental temporal principles, while dynamic edges capture instance-specific pattern similarity. An R-GCN then processes this graph to produce a structural representation that is subsequently fused with the patch embeddings. The resulting structurally-informed representation is then infused into a pre-trained Transformer backbone to generate the forecast.

Patch-based Input Representation

Following standard practice, we first apply Reversible Instance Normalization (RevIN) (Kim et al. 2022) to the input $\mathbf{X}$. Each of the M channels is then segmented into $N_p = \lfloor L/P \rfloor$ non-overlapping patches $\mathbf{p}_{i,j} \in \mathbb{R}^P$, where P is the patch length. Each patch is subsequently projected into a d-dimensional embedding space via a shared linear layer and augmented with a sinusoidal positional encoding $\mathbf{PE}_j$ to yield its initial embedding $\mathbf{h}_{i,j}^{(0)}$,

$$\mathbf{h}_{i,j}^{(0)} = \text{Linear}(\mathbf{p}_{i,j}) + \mathbf{PE}_j, \tag{1}$$

and this process is applied to all patches, which are then collected to form the initial feature matrix $\mathbf{H}^{(0)} \in \mathbb{R}^{(M \cdot N_p) \times d}$. These initial embeddings capture local patterns but remain unaware of the relational structure between patches, motivating our subsequent graph-based encoding.

Graph-based Structural Encoding

The graph-based encoding module defines the relational structure of the time series data via the HPRG. This graph

is then processed by an R-GCN to learn structure-infused representations.

HPRG Construction The HPRG is a heterogeneous graph defined as $\mathcal{G} = (\mathcal{V}, \{\mathcal{E}_r\}_{r\in\mathcal{R}})$, where $\mathcal{V} = \{v_{i,j}\}$ is the vertex set, with each vertex $v_{i,j}$ corresponding to the patch $\mathbf{p}_{i,j}$, and $\mathcal{R}$ is a set of relation types. The HPRG is composed of two main categories of edges, each with specific directionality reflecting its underlying assumption.

Static Structural Edges. These universally applicable edges encode fundamental temporal relationships and are constructed identically for every instance.

Temporal Adjacency ($\mathcal{E}_{\text{adj}}$) establishes bidirectional edges between consecutive patches within each variable. This enforces local continuity, reflecting that adjacent patches are mutually contextual.

$$\mathcal{E}_{\text{adj}} = \{(v_{i,j}, v_{i,j+1}), (v_{i,j+1}, v_{i,j}) \mid 1 \le i \le M, 1 \le j < N_p\}. \tag{2}$$

Cross-Variable Synchrony ($\mathcal{E}_{\text{sync}}$) captures contemporaneous correlations by creating bidirectional edges between patches at the same time index. The bidirectionality models the mutual influence between different variables at a specific moment.

$$\mathcal{E}_{\text{sync}} = \{(v_{i,j}, v_{k,j}), (v_{k,j}, v_{i,j}) \mid 1 \le j \le N_p, 1 \le i < k \le M\}. \tag{3}$$

Lagged Dependencies ($\mathcal{E}_{\text{lag}}$) creates directed edges from each patch at time index j to all patches at $j+1$. This directional design explicitly models the causal flow of time, where past values influence future ones.

$$\mathcal{E}_{\text{lag}} = \{(v_{i,j}, v_{k,j+1}) \mid 1 \le i, k \le M, 1 \le j < N_p\}. \tag{4}$$

Dynamic Adaptive Edges. To capture instance-specific latent patterns, we construct a set of directed edges based on pattern similarity. A single edge type, *Pattern Similarity* ($\mathcal{E}_{\text{sim}}$), connects each node to its top-K most similar peers, where K is a hyperparameter. Formally, for any given node $v_a \in \mathcal{V}$, its neighborhood $\mathcal{N}_{\text{sim}}(v_a)$ is determined by identifying the subset of vertices $\mathcal{S} \subseteq \mathcal{V} \setminus \{v_a\}$ of size K that maximizes the sum of cosine similarities:

$$\mathcal{N}_{\text{sim}}(v_a) = \underset{\mathcal{S}\subseteq\mathcal{V}\setminus\{v_a\}, |\mathcal{S}|=K}{\text{argmax}} \sum_{v_b\in\mathcal{S}} \frac{\mathbf{h}_{v_a}^{(0)\top}\mathbf{h}_{v_b}^{(0)}}{\|\mathbf{h}_{v_a}^{(0)}\|\|\mathbf{h}_{v_b}^{(0)}\|}. \tag{5}$$

The resulting edges are directed from v_a to the nodes in its identified neighborhood, as similarity is not necessarily symmetric. The complete edge set is then constructed as:

$$\mathcal{E}_{\text{sim}} = \{(v_a, v_b) \mid v_b \in \mathcal{N}_{\text{sim}}(v_a), \forall v_a \in \mathcal{V}\}. \tag{6}$$

Structure-Aware Representation Learning We employ an L_G-layer R-GCN to learn structure-aware representations, where L_G is a hyperparameter defining the depth of graph convolutions. For each layer $l \in \{0, \dots, L_G - 1\}$, the R-GCN updates the embedding of each node v by aggregating messages from its neighbors u across all relation types $r \in \mathcal{R}$:

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\mathbf{W}_{\text{self}}\mathbf{h}_v^{(l)} + \sum_{r\in\mathcal{R}}\sum_{u\in\mathcal{N}_r(v)} \frac{1}{|\mathcal{N}_r(v)|}\mathbf{W}_r\mathbf{h}_u^{(l)}\right), \tag{7}$$

where the process is initialized with the previously defined patch embeddings, i.e., $\mathbf{h}_v^{(0)} = \mathbf{h}_{i,j}^{(0)}$ for a node v corresponding to patch (i, j). Furthermore, $\mathcal{N}_r(v)$ is the set of neighbors of node v under relation r, $\mathbf{W}_r$ and $\mathbf{W}_{\text{self}}$ are learnable weight matrices, and σ is a non-linear activation function. The output of the final layer, $\mathbf{h}_v^{(L_G)}$, serves as the final graph-aware representation, which we denote by $\mathbf{h}_v^{(G)}$, where the superscript (G) signifies that it is derived from the graph structure. This vector representation is then fused with the initial embedding via a gated mechanism:

$$\mathbf{h}'_v = \alpha\mathbf{h}_v^{(G)} + (1-\alpha)\mathbf{h}_v^{(0)}, \tag{8}$$

where $\alpha = \text{sigmoid}(g)$ is a learnable gating coefficient, and g is a trainable scalar parameter. This fusion is performed for all nodes, yielding a final set of structurally-informed embeddings for the forecasting stage.

The module's impact is evident within the self-attention mechanism. Let $\mathbf{q}(\cdot)$ and $\mathbf{k}(\cdot)$ be the query and key projections. The new attention score $s'(v_a, v_b) \propto \mathbf{q}(\mathbf{h}'_a)^\top\mathbf{k}(\mathbf{h}'_b)$ can be decomposed as:

$$s'(v_a, v_b) \propto (1-\alpha)^2 \cdot \left(\mathbf{q}(\mathbf{h}_a^{(0)})^\top\mathbf{k}(\mathbf{h}_b^{(0)})\right) + \Delta_{\mathcal{G}}(v_a, v_b), \tag{9}$$

where the first term is the original content-based attention. The second term, $\Delta_{\mathcal{G}}(v_a, v_b)$, is a structural attention bias induced by the graph $\mathcal{G}$:

$$\begin{aligned}\Delta_{\mathcal{G}}(v_a, v_b) = {} & \alpha(1-\alpha)\mathbf{q}(\mathbf{h}_a^{(0)})^\top\mathbf{k}(\mathbf{h}_b^{(G)}) \\ & + \alpha(1-\alpha)\mathbf{q}(\mathbf{h}_a^{(G)})^\top\mathbf{k}(\mathbf{h}_b^{(0)}) \\ & + \alpha^2\mathbf{q}(\mathbf{h}_a^{(G)})^\top\mathbf{k}(\mathbf{h}_b^{(G)}).\end{aligned} \tag{10}$$

Since $\mathbf{h}_v^{(G)}$ is a function of its L_G-hop neighborhood, the structural bias $\Delta_{\mathcal{G}}(v_a, v_b)$ explicitly encodes the topological proximity between nodes. This decomposition reveals how our method elevates self-attention from simple feature matching to a structure-aware process, guiding the attention mechanism along the relational pathways defined by the HPRG.

Forecasting with Graph-Infused Representations

The structurally-informed embeddings from the previous stage are collected into a sequence matrix, denoted as $\mathbf{H}' \in \mathbb{R}^{(M\cdot N_p)\times d}$, and fed into the pre-trained GPT-2 backbone. To adapt the model for forecasting, we employ a custom parameter-efficient fine-tuning (PEFT) strategy. This strategy involves freezing the core computational blocks of the pre-trained backbone, namely the attention and feed-forward layers, while exclusively training a minimal set of parameters essential for the forecasting task. Specifically, the trainable parameters comprise the R-GCN module for structural

encoding, the input patch embedding and output projection layers that serve as task-specific interfaces, and the backbone's original positional embedding and normalization layers, which are fine-tuned to adapt to the distinct characteristics of time series data. The final head maps the output representations to the prediction horizon H. The model is trained end-to-end by minimizing the MSE loss, denoted as $\mathcal{L}_{\text{MSE}}$:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{H \cdot M} \|\mathbf{Y} - \hat{\mathbf{Y}}\|_F^2, \tag{11}$$

where $\hat{\mathbf{Y}}$ is the model's forecast and $\mathbf{Y} \in \mathbb{R}^{H \times M}$ represents the ground truth future values.

Experiments

To demonstrate the effectiveness of our proposed GraFT, we conduct extensive experiments on time series forecasting tasks including long-term forecasting and zero-shot learning. Besides, we further validate our model through ablation studies, efficiency analysis, and architectural comparisons.

Baselines. We compare our model against a comprehensive set of competitive baselines, with results cited from their original publications where available. These baselines are categorized as follows: (1) LLM-based models: time series-specific approaches like GPT4TS (Zhou et al. 2023) and S^2IP-LLM (Pan et al. 2024), and prompt-based methods such as Time-LLM (Jin et al. 2024) and FSCA (Hu et al. 2025); (2) Transformer-based models: PatchTST (Nie et al. 2023) and iTransformer (Liu et al. 2024); (3) CNN-based model: TimesNet (Wu et al. 2023); (4) Linear-based model: DLinear (Zeng et al. 2023).

Implementation Details. We adopt MSE and Mean Absolute Error (MAE) as evaluation metrics for all models. Following Hu et al. (2025), we employ a pre-trained GPT-2 model (Radford et al. 2019), utilizing its first 4 Transformer layers as the backbone. The model is optimized using the Adam optimizer (Kingma and Ba 2015) with a cosine annealing learning rate scheduler. For our HPRG module, we utilize a 2-layer R-GCN with its fusion gate parameter g initialized to 0.5. The number of neighbors for dynamic similarity edges, K, is set to 5. To ensure a fair comparison with recent LLM-based methods (Zhou et al. 2023; Pan et al. 2024; Jin et al. 2024; Hu et al. 2025), we maintain a consistent input sequence length of 512. For other baselines, we report their best-performing results as published in their respective papers. All experiments are conducted on a single NVIDIA A800 GPU, and the reported results are averaged over three independent runs with different random seeds for reproducibility.

Long-term Forecasting

Setups. We conduct experiments on eight widely-used benchmark datasets for long-term time series forecasting: ETTh1, ETTh2, ETTm1, ETTm2 (Zhou et al. 2021), Weather, ECL, Traffic, and ILI (Wu et al. 2023). Performance is evaluated over four prediction horizons: {96, 192, 336, 720} for all datasets except ILI, which uses horizons of {24, 36, 48, 60}.

Results. Table 1 summarizes the long-term forecasting results. Compared to existing LLM-based baselines, GraFT consistently outperforms other methods, achieving average MSE reductions of 23.8% over FSCA, 14.6% over S^2IP-LLM, 8.6% over FSCA, and 10.6% over the LLaMA-7B-based Time-LLM. Notably, on the challenging ECL and ILI datasets, our model surpasses the second-best method with significant MSE reductions of 28.5% and 9.9%, respectively. Moreover, GraFT achieves state-of-the-art results in over 70% of all experimental cases. These results highlight that explicitly encoding relational priors is critical for capturing the intricate dynamics that underpin superior forecasting accuracy.

Zero-shot Learning

Setups. To evaluate GraFT's generalization ability, we adopt the zero-shot transfer protocol from prior work (Jin et al. 2024), using the same eight transfer pairs. In this setup, the model is trained on a source ETT dataset and then directly evaluated on a distinct target dataset under its long-term forecasting horizons, simulating realistic distribution shifts.

Results. As summarized in Table 2, GraFT demonstrates superior generalization, achieving the lowest average MSE and MAE across all transfer tasks. This robust performance is attributed to the principled structural priors encoded by the HPRG. We attribute this robust performance to the HPRG's structural priors, which foster the learning of fundamental, transferable representations of time series dynamics, enabling effective knowledge transfer across disparate data distributions.

Ablation Study

Table 3 presents the ablation study of the HPRG's relational components, defined by the edge set $\mathcal{E} \in \{\mathcal{E}_{\text{adj}}, \mathcal{E}_{\text{sync}}, \mathcal{E}_{\text{lag}}, \mathcal{E}_{\text{sim}}\}$. The results underscore the necessity of the graph-based priors, as the model's performance considerably degrades upon their complete removal (w/o $\mathcal{E}$) or when using any single edge set in isolation. Furthermore, the relative contribution of each edge type varies across datasets. For example, removing similarity edges (w/o $\mathcal{E}_{\text{sim}}$) is most detrimental on ECL, while removing lagged edges (w/o $\mathcal{E}_{\text{lag}}$) induces the largest performance drop on ILI. This validates our heterogeneous design, demonstrating that the model's robust performance arises from the synergistic integration of diverse relational cues.

Efficiency Analysis

Figure 4 visualizes the trade-off between forecasting performance and training time. This efficiency stems from GraFT's prompt-free and structurally-intrinsic design, allowing it to achieve state-of-the-art performance at a fraction of the computational cost of other LLM-based methods. Specifically, approaches like FSCA and Time-LLM rely on elaborate prompting schemes and larger model backbones, which incurs significant computational overhead. While lightweight models such as DLinear and PatchTST are faster, GraFT provides a substantial improvement in

Models		GraFT		FSCA		S^{2}IP-LLM		Time-LLM		GPT4TS		iTransformer		PatchTST		TimesNet		DLinear	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.315**	**0.373**	0.349	0.389	0.366	0.396	0.362	0.392	0.376	0.397	0.395	0.420	0.370	0.399	0.384	0.402	0.375	0.399
	192	**0.371**	**0.407**	0.390	0.415	0.401	0.420	0.398	0.418	0.416	0.418	0.427	0.441	0.413	0.421	0.436	0.429	0.405	0.416
	336	**0.388**	**0.425**	0.402	0.432	0.412	0.431	0.430	0.427	0.442	0.433	0.445	0.457	0.422	0.436	0.491	0.469	0.439	0.448
	720	**0.425**	**0.456**	0.433	0.460	0.440	0.458	0.442	0.457	0.477	**0.456**	0.537	0.530	0.447	0.466	0.521	0.500	0.472	0.490
	Avg	**0.375**	**0.415**	0.394	0.424	0.406	0.427	0.408	0.423	0.427	0.426	0.451	0.462	0.413	0.430	0.458	0.450	0.422	0.437
ETTh2	96	**0.239**	**0.315**	0.256	0.328	0.278	0.340	0.268	0.328	0.285	0.342	0.304	0.360	0.274	0.336	0.340	0.374	0.289	0.353
	192	**0.268**	**0.339**	0.311	0.372	0.346	0.385	0.329	0.375	0.354	0.389	0.377	0.403	0.339	0.379	0.402	0.414	0.383	0.418
	336	**0.304**	**0.369**	0.308	0.372	0.367	0.406	0.368	0.409	0.373	0.407	0.405	0.429	0.329	0.380	0.452	0.452	0.448	0.465
	720	0.378	0.423	0.390	0.428	0.400	0.436	**0.372**	**0.420**	0.406	0.441	0.443	0.464	0.379	0.422	0.462	0.468	0.605	0.551
	Avg	**0.298**	**0.362**	0.316	0.375	0.347	0.391	0.334	0.383	0.354	0.394	0.382	0.414	0.330	0.379	0.414	0.427	0.431	0.446
ETTm1	96	**0.159**	**0.256**	0.282	0.343	0.288	0.346	0.272	0.334	0.292	0.346	0.312	0.366	0.290	0.342	0.338	0.375	0.299	0.343
	192	**0.267**	**0.331**	0.324	0.369	0.323	0.365	0.310	0.358	0.332	0.372	0.347	0.385	0.332	0.369	0.374	0.387	0.335	0.365
	336	**0.345**	0.390	0.356	0.386	0.359	0.390	0.352	**0.384**	0.366	0.394	0.379	0.404	0.366	0.392	0.410	0.411	0.369	0.386
	720	0.413	0.412	0.405	0.417	0.403	0.418	**0.383**	**0.411**	0.417	0.421	0.441	0.442	0.416	0.420	0.478	0.450	0.425	0.421
	Avg	**0.296**	**0.347**	0.342	0.378	0.343	0.379	0.329	0.372	0.352	0.383	0.370	0.399	0.351	0.380	0.400	0.406	0.357	0.378
ETTm2	96	**0.136**	**0.236**	0.164	0.254	0.165	0.257	0.161	0.253	0.173	0.262	0.179	0.271	0.165	0.255	0.187	0.267	0.167	0.269
	192	0.226	0.300	0.222	0.296	0.222	0.299	**0.219**	0.293	0.229	0.301	0.242	0.313	0.220	**0.292**	0.249	0.309	0.224	0.303
	336	**0.267**	**0.326**	0.269	**0.326**	0.277	0.330	0.271	0.329	0.286	0.341	0.288	0.344	0.274	0.329	0.321	0.351	0.281	0.342
	720	**0.341**	**0.372**	0.346	0.381	0.363	0.390	0.352	0.379	0.378	0.401	0.378	0.397	0.362	0.385	0.408	0.403	0.397	0.421
	Avg	**0.243**	**0.309**	0.250	0.314	0.257	0.319	0.251	0.313	0.266	0.326	0.272	0.331	0.255	0.315	0.291	0.333	0.267	0.333
Weather	96	0.146	0.200	0.146	0.196	**0.145**	**0.195**	0.147	0.201	0.162	0.212	0.253	0.304	0.149	0.198	0.172	0.220	0.176	0.237
	192	**0.189**	0.241	0.193	0.241	0.190	0.235	**0.189**	**0.234**	0.204	0.248	0.280	0.319	0.194	0.241	0.219	0.261	0.220	0.282
	336	0.244	0.283	0.244	**0.279**	**0.243**	0.280	0.262	**0.279**	0.254	0.286	0.321	0.344	0.245	0.282	0.280	0.306	0.265	0.319
	720	0.316	0.336	0.314	0.333	0.312	0.326	**0.304**	**0.316**	0.326	0.337	0.364	0.374	0.314	0.334	0.365	0.359	0.333	0.362
	Avg	0.224	0.265	0.224	0.262	**0.222**	0.259	0.225	**0.257**	0.237	0.270	0.304	0.335	0.225	0.264	0.259	0.287	0.248	0.300
ECL	96	**0.064**	**0.157**	0.128	0.222	0.135	0.230	0.131	0.224	0.139	0.238	0.147	0.248	0.129	0.222	0.168	0.272	0.140	0.237
	192	**0.095**	**0.193**	0.146	0.239	0.149	0.247	0.152	0.241	0.153	0.251	0.165	0.267	0.157	0.240	0.184	0.289	0.153	0.249
	336	**0.123**	**0.224**	0.163	0.258	0.167	0.266	0.160	0.248	0.169	0.266	0.178	0.279	0.163	0.259	0.198	0.300	0.169	0.267
	720	**0.170**	**0.267**	0.199	0.287	0.200	0.287	0.192	0.298	0.206	0.297	0.322	0.398	0.197	0.290	0.220	0.320	0.203	0.301
	Avg	**0.113**	**0.210**	0.159	0.252	0.161	0.257	0.158	0.252	0.167	0.263	0.203	0.298	0.161	0.252	0.192	0.295	0.166	0.263
Traffic	96	**0.300**	0.262	0.355	**0.246**	0.379	0.274	0.362	0.248	0.388	0.282	0.367	0.288	0.360	0.249	0.593	0.321	0.410	0.282
	192	**0.349**	0.282	0.377	0.255	0.397	0.282	0.374	**0.247**	0.407	0.290	0.378	0.293	0.379	0.256	0.617	0.336	0.423	0.287
	336	**0.385**	0.283	0.387	0.265	0.407	0.289	**0.385**	0.271	0.412	0.294	0.389	0.294	0.392	**0.264**	0.629	0.336	0.436	0.296
	720	0.424	0.313	0.425	0.287	0.440	0.301	0.430	0.288	0.450	0.312	**0.401**	0.304	0.432	**0.286**	0.640	0.350	0.466	0.315
	Avg	**0.365**	0.285	0.386	**0.263**	0.405	0.286	0.388	0.264	0.414	0.294	0.389	0.295	0.390	**0.263**	0.620	0.336	0.433	0.295
ILI	24	**1.177**	**0.652**	1.206	0.728	1.467	0.778	1.285	0.727	2.063	0.881	1.694	0.874	1.319	0.754	2.317	0.934	2.215	1.081
	36	**1.183**	**0.681**	1.251	0.750	1.534	0.841	1.404	0.814	1.868	0.892	2.229	0.983	1.430	0.834	1.972	0.920	1.963	0.963
	48	**1.195**	**0.702**	1.566	0.818	1.608	0.836	1.523	0.807	1.790	0.884	2.382	0.995	1.553	0.815	2.238	0.940	2.130	1.021
	60	**1.417**	**0.770**	1.495	0.833	1.597	0.849	1.531	0.854	1.979	0.957	1.988	0.913	1.470	0.788	2.027	0.928	2.368	1.092
	Avg	**1.243**	**0.701**	1.380	0.783	1.552	0.826	1.435	0.801	1.925	0.903	2.073	0.941	1.443	0.797	2.139	0.931	2.169	1.041
1st Count		**58**		4		4		14		1		1		4		0		0	

Table 1: Long-term forecasting results. We set the forecasting horizons $H \in \{24, 36, 48, 60\}$ for ILI and {96, 192, 336, 720} for the others. A lower value indicates better performance. **Bold**: the best, Underline: the second best.

forecasting performance for a modest increase in training time. This demonstrates that our graph-infusion approach strikes a highly favorable balance between predictive power and computational efficiency.

Architectural Analysis

We further compare our R-GCN encoder against several alternatives, with results presented in Table 4. The superiority of R-GCN over other graph-based variants indicates that GCN's isotropic processing is insufficient for the HPRG's diverse edge types, while the additional parameterization of the Relational Graph Attention Network (R-GAT) offers no advantage. While non-graph baselines like MLP, CNN, and Attention demonstrate some predictive capability, their performance is inherently limited as they lack the architectural mechanisms to process the explicit relational priors. These

Models	GraFT		FSCA		S^{2}IP-LLM		Time-LLM		GPT4TS		iTransformer		PatchTST		TimesNet		DLinear	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
h1 → h2	**0.287**	**0.351**	0.313	0.369	0.403	0.417	0.353	0.387	0.406	0.422	0.457	0.455	0.380	0.488	0.421	0.431	0.493	0.488
h1 → m2	0.281	0.341	0.290	0.348	0.325	0.360	**0.273**	**0.340**	0.325	0.363	0.360	0.390	0.314	0.360	0.327	0.361	0.415	0.452
h2 → h1	0.546	0.519	0.527	0.507	0.669	0.560	**0.479**	**0.474**	0.757	0.578	0.868	0.625	0.565	0.513	0.865	0.621	0.703	0.574
h2 → m2	0.281	0.345	0.288	0.347	0.327	0.363	**0.272**	**0.341**	0.335	0.370	0.335	0.382	0.325	0.365	0.342	0.376	0.328	0.386
m1 → h2	**0.317**	**0.373**	0.353	0.398	0.442	0.439	0.381	0.412	0.433	0.439	0.455	0.458	0.439	0.438	0.457	0.454	0.464	0.475
m1 → m2	**0.246**	**0.309**	0.264	0.319	0.304	0.347	0.268	0.320	0.313	0.348	0.319	0.363	0.296	0.334	0.322	0.354	0.335	0.389
m2 → h2	**0.337**	**0.389**	0.343	0.393	0.406	0.429	0.354	0.400	0.435	0.443	0.432	0.447	0.409	0.425	0.435	0.443	0.455	0.471
m2 → m1	0.473	0.454	0.480	0.463	0.622	0.532	**0.414**	**0.438**	0.769	0.567	0.706	0.572	0.568	0.492	0.769	0.567	0.649	0.537
Avg	**0.346**	**0.385**	0.357	0.393	0.437	0.431	0.349	0.389	0.472	0.441	0.492	0.462	0.412	0.427	0.492	0.451	0.480	0.472

Table 2: Zero-shot forecasting results on ETT datasets, averaged across four forecasting horizons: $H \in \{96, 192, 336, 720\}$. **Bold**: the best, Underline: the second best.

Variant	ECL (Avg)		ILI (Avg)	
	MSE	MAE	MSE	MAE
Full	**0.113**	**0.210**	**1.243**	**0.701**
w/o $\mathcal{E}_{\text{adj}}$	0.155	0.262	1.297	0.732
w/o $\mathcal{E}_{\text{sync}}$	0.154	0.257	1.452	0.751
w/o $\mathcal{E}_{\text{lag}}$	0.141	0.246	1.607	0.850
w/o $\mathcal{E}_{\text{sim}}$	0.244	0.356	1.506	0.832
$\mathcal{E}_{\text{adj}}$	0.255	0.367	2.049	0.981
$\mathcal{E}_{\text{sync}}$	0.319	0.423	1.783	0.900
$\mathcal{E}_{\text{lag}}$	0.277	0.384	2.337	1.015
$\mathcal{E}_{\text{sim}}$	0.187	0.295	1.882	0.940
w/o $\mathcal{E}$	0.319	0.423	2.722	1.138

Table 3: Ablation study of HPRG components on ECL and ILI. All metrics are averaged over four prediction horizons. A variant listed by an edge type alone (e.g., $\mathcal{E}_{\text{adj}}$) uses only that single edge type.

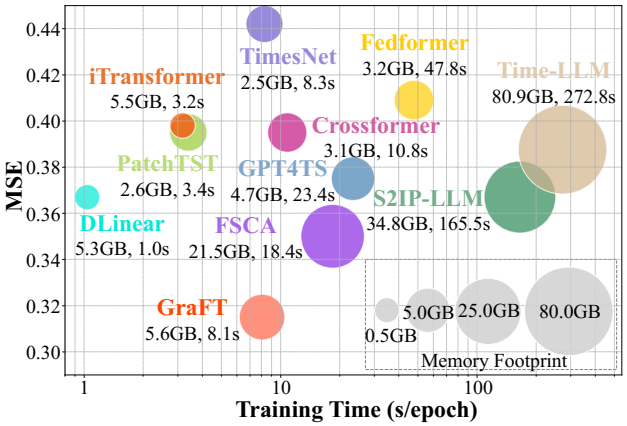


Figure 4: Model efficiency comparison under input-512-predict-96 of ETTh1. The y-axis shows MSE, while the x-axis (log scale) represents training time per epoch.

findings indicate that a relation-aware graph encoder is essential for capitalizing on the explicitly encoded structural priors.

Conclusion

This paper presents GraFT, a framework that represents a paradigm shift in applying LLMs to time series. Instead of treating LLMs as black boxes requiring complex data alignment, we fundamentally adapt its internal architecture by infusing it with explicit structural knowledge. The cornerstone of this approach is the HPRG, which serves as a structured bridge by translating the complex web of temporal principles and instance-specific patterns into explicit relational priors for the LLM backbone. The effectiveness of this conceptually simple yet powerful approach is demonstrated by GraFT's state-of-the-art results in both long-term forecasting and zero-shot learning, achieved with high computational efficiency. This approach demonstrates the potential

Module	ETTh1	ETTh2	ETTm1	ETTm2
R-GCN	**0.375**	**0.298**	**0.296**	**0.243**
GCN	0.393	0.319	0.353	0.264
R-GAT	0.412	0.348	0.356	0.265
MLP	0.399	0.317	0.328	0.262
CNN	0.406	0.358	0.348	0.259
Attention	0.402	0.360	0.349	0.259

Table 4: Comparison of different architectures on ETT datasets. The values reported are the average MSE for four prediction lengths.

for developing more architecturally-aware LLMs for time series forecasting, establishing direct structural infusion as a more robust and principled strategy than methods reliant on peripheral data alignment.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2024YFC3017004, in part by the Beijing Natural Science Foundation under Grant L211020, and in part by the Innovative Talent Training Fund of University of Science and Technology Beijing.

References

Angryk, R. A.; Martens, P. C.; Aydin, B.; Kempton, D.; Mahajan, S. S.; Basodi, S.; Ahmadzadeh, A.; Cai, X.; Filali Boubrahimi, S.; Hamdi, S. M.; et al. 2020. Multivariate time series dataset for space weather data analytics. *Scientific Data*, 7(1): 227.

Das, A.; Kong, W.; Leach, A.; Mathur, S. K.; Sen, R.; and Yu, R. 2023. Long-term forecasting with TiDE: Time-series dense encoder. *Transactions on Machine Learning Research*.

Demirel, Ö. F.; Zaim, S.; Çalişkan, A.; and Özuyar, P. 2012. Forecasting natural gas consumption in Istanbul using neural networks and multivariate time series methods. *Turkish Journal of Electrical Engineering and Computer Sciences*, 20(5): 695–711.

Gao, J.; Zhang, X.; Tian, L.; Liu, Y.; Wang, J.; Li, Z.; and Hu, X. 2022. MTGNN: Multi-task graph neural network based few-shot learning for disease similarity measurement. *Methods*, 198: 88–95.

Gu, A.; and Dao, T. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*.

Hu, Y.; Li, Q.; Zhang, D.; Yan, J.; and Chen, Y. 2025. Context-alignment: Activating and enhancing LLMs capabilities in time series. In *International Conference on Learning Representations*.

Jin, M.; Wang, S.; Ma, L.; Chu, Z.; Zhang, J. Y.; Shi, X.; Chen, P.; Liang, Y.; Li, Y.; Pan, S.; and Wen, Q. 2024. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations*.

Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.; and Choo, J. 2022. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.

Kingma, D. P.; and Ba, J. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Liu, C.; Xiao, Z.; Long, C.; Wang, D.; Li, T.; and Jiang, H. 2025. MVCAR: Multi-view collaborative graph network for private car carbon emission prediction. *IEEE Transactions on Intelligent Transportation Systems*, 26(1): 472–483.

Liu, C.; Xiao, Z.; Wang, D.; Cheng, M.; Chen, H.; and Cai, J. 2022a. Foreseeing private car transfer between urban regions with multiple graph-based generative adversarial networks. *World Wide Web*, 25(6): 2515–2534.

Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A. X.; and Dustdar, S. 2022b. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*.

Liu, Y.; Li, C.; Wang, J.; and Long, M. 2023. Koopa: Learning non-stationary time series dynamics with Koopman predictors. In *Advances in Neural Information Processing Systems*.

Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022c. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *Advances in Neural Information Processing Systems*.

Miao, H.; Zhao, Y.; Guo, C.; Yang, B.; Kai, Z.; Huang, F.; Xie, J.; and Jensen, C. S. 2024. A unified replay-based continuous learning framework for spatio-temporal prediction on streaming data. In *International Conference on Data Engineering*.

Mourya, S.; Reddy, P.; Amuru, S.; and Kuchi, K. K. 2024. Spectral temporal graph neural network for massive MIMO CSI prediction. *IEEE Wireless Communications Letters*, 13(5): 1399–1403.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*.

Niu, T.; Wang, J.; Lu, H.; Yang, W.; and Du, P. 2020. Developing a deep learning framework with two-stage feature selection for multivariate financial time series forecasting. *Expert Systems with Applications*, 148: 113237.

Pan, Z.; Jiang, Y.; Garg, S.; Schneider, A.; Nevmyvaka, Y.; and Song, D. 2024. S2IP-LLM: Semantic space informed prompt learning with LLM for time series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*. PMLR.

Patton, A. 2013. Copula methods for forecasting multivariate time series. *Handbook of Economic Forecasting*, 2: 899–960.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners. OpenAI Blog.

Schlichtkrull, M. S.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th European Semantic Web Conference*, volume 10843 of *Lecture Notes in Computer Science*, 593–607. Springer.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 5998–6008.

Wang, S.; Li, J.; Shi, X.; Ye, Z.; Mo, B.; Lin, W.; Ju, S.; Chu, Z.; and Jin, M. 2025. TimeMixer++: A general time series pattern machine for universal predictive analysis. In *International Conference on Learning Representations*.

Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and Zhou, J. 2024. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*.

Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; and Sun, L. 2023. Transformers in time series: A survey. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, IJCAI '23, 6895–6903.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, 22419–22430.

Wu, Y.; Meng, X.; Hu, H.; Zhang, J.; Dong, Y.; and Lu, D. 2025. Affirm: Interactive mamba with adaptive fourier filters for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 21599–21607.

Xiao, J.; Xiao, Z.; Wang, D.; Havyarimana, V.; Liu, C.; Zou, C.; and Wu, D. 2022. Vehicle trajectory interpolation based on ensemble transfer regression. *IEEE Transactions on Intelligent Transportation Systems*, 23(7): 7680–7691.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, 11121–11128.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 11106–11115.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 27268–27286. PMLR.

Zhou, T.; Niu, P.; Wang, X.; Sun, L.; and Jin, R. 2023. One fits all: Power general time series analysis by pretrained LM. In *Advances in Neural Information Processing Systems*, volume 36.