StreamTimer: Efficient Inference for Long-Context Time Series Transformers

Xiyu Meng[1,*], **Yuhan Wu**[1,*], **Canran Xiao**[2], **Yabo Dong**[1,†] and **Duanqing Xu**[1]
[1]Zhejiang University
[2]Sun Yat-sen University
{mengxiyu, wuyuhan, dongyb, xdq}@zju.edu.cn, xiaocanran999@gmail.com

Abstract

Time series forecasting (TSF) plays a vital role across various domains such as finance, energy, healthcare, and meteorology. Currently, most deep learning based TSF methods typically operate with a fixed lookback window. This approach comes from the high compute and memory costs of long contexts, as well as the standard practice of using sliding windows. This creates a trade-off. Making the window larger reduces the number of training samples, which can harm stability and generalization. However, keeping the window small prevents the model from using long history during inference. We propose an inference-only streaming autoregressive framework that replaces repeated full-context recomputation with a one-time context warmup and incremental decoding, enabling efficient long-history forecasting without retraining. While straightforward caching attentions is brittle for time series due to distribution shifts and noisy or redundant histories, we address these issues with cache-consistent normalization and selective memory under a fixed cache budget. Across diverse benchmarks, our approach substantially reduces inference latency with no or marginal accuracy loss, and often improves performance when longer lookbacks are beneficial.

1 Introduction

Long-term time series forecasting has evolved rapidly over the past decades, witnessing a paradigm shift from traditional statistical methods to deep learning architectures [Jin *et al.*, 2024; Wang *et al.*, 2024b]. While these substantial architectural innovations, their fundamental operational protocol remains surprisingly static, characterized by a reliance on a fixed lookback window [Qiu *et al.*, 2024]. Specifically, models are constrained to a pre-determined number of historical time steps (e.g., 96 days or 720 hours) to forecast future values [Liu *et al.*, 2024b]. This rigid sliding window approach persists even as underlying architectures grow increasingly complex, creating a misalignment between the model's intrinsic capacity and its effective utilization of context.

This operational rigidity introduces a paradox in modern TSF. That is, longer historical contexts theoretically harbor richer temporal information, but extending the window to capture it is computationally expensive. Given the quadratic complexity of self-attention, lengthening the window substantially increases memory and compute costs [Zhou *et al.*, 2021]. Besides, empirical evidence suggests that simply enlarging the window often degrades performance due to attention dispersion, insufficient training samples, and overfitting to irrelevant noise [Nie *et al.*, 2023; Liu *et al.*, 2024d]. Consequently, practitioners are trapped in a dilemma: they must either truncate history or risk computational intractability and model instability. This suggests that the fixed-window paradigm is inherently suboptimal for capturing the continuous and unrestricted dynamics of time series data.

To resolve these structural limitations, we propose a transition from the repeated recomputation of fixed windows to an Inference-Only Streaming Autoregressive Framework. Inspired by the efficient caching mechanisms in Large Language Models (LLMs), our approach views forecasting as a continuous, stateful process. By replacing repeated full-context recomputation with a one-time context warmup followed by incremental decoding, we reduce the complexity of extending context from quadratic to linear. This allows the model to efficiently leverage extensive historical data without the latency penalties of massive fixed windows. Moreover, as an inference-time optimization, this framework requires no retraining, allowing the trained model to achieve superior long-context performance directly

However, the direct adaptation of NLP-style Key-Value (KV) caching [Xiao *et al.*, 2024] to time series is brittle due to two domain challenges. First, time series data are non-stationary. Current baselines typically employ Reversible Instance Normalization (RevIN) [Kim *et al.*, 2021; Qiu *et al.*, 2024] to mitigate statistical drifts; however, this creates distribution cache mismatches, meaning that the key vectors stored at time t effectively reside in a different feature space than Query vectors generated at $t + 1$. Second, unlike semantically dense text, time series history is replete with redundancy, where indiscriminately caching noise diminishes the efficacy of the attention mechanism.

To overcome these issues, we introduce StreamTimer, a robust streaming framework tailored for TSF. We propose Cache-Consistent Normalization to dynamically align cached

representations with the current statistical regime, and incorporate window attention with attention sinks to maintain temporal coherence while retaining only high-value historical tokens. Our contributions lie in three aspects:

- We propose the first inference-only streaming framework with Timer backbones for TSF that effectively utilizes the lookback window, decoupling context length from computational cost.
- We identify and resolve the normalization mismatch problem in generative trained large time series models, ensuring mathematically consistent distribution during streaming inference.
- Experiments validate that StreamTimer achieves significant speedup while outperforming SOTA baselines, particularly on ultra-long sequences.

2 Related Work

2.1 Time Series Forecasting

Contemporary deep learning architectures for time series forecasting predominantly focus on modeling complex global temporal dependencies and multivariate interactions from historical data. Early studies built on Recurrent Neural Networks [Lin *et al.*, 2023] and Temporal Convolutional Networks [Luo and Wang, 2024] to capture local patterns through gated recurrences and dilated convolutions. More recently, Transformers have emerged as the mainstream for long-horizon forecasting due to their capability to capture long-range dependencies. Specifically, Informer [Zhou *et al.*, 2021] introduces sparse attention tailored to long sequences, PatchTST [Nie *et al.*, 2023] and iTransformer [Liu *et al.*, 2024b] leverage patch- and channel-centric designs, and TimeMixer [Wang *et al.*, 2024a] and TimeMoE [Shi *et al.*, 2025] enhance the exploitation of multivariate structures and multi-scale temporal patterns. While these backbones provide robust representation capabilities, they typically rely on a fixed-lookback-fixed-forecast recomputation paradigm.

2.2 Generative Pre-training for Large Time Series Models

The field of TSF has transitioned from small-scale, task-specific deep learning models [Wu *et al.*, 2025; Lin *et al.*, 2024; Wu *et al.*, 2024; Zeng *et al.*, 2023] to Large Time Series Models (LTSMs) [Zhou *et al.*, 2023; Jin *et al.*, 2024; Liu *et al.*, 2024a; Woo *et al.*, 2024; Ansari and Stella, 2024] powered by unsupervised pre-training on massive datasets. Inspired by the success of LLMs, recent works have explored generative pre-training paradigms to build foundation models for time series.

Early efforts, such as Time-LLM [Jin *et al.*, 2024] and LLM4TS [Chang *et al.*, 2024], repurposed pre-trained LLMs (e.g., LLaMA, GPT-2) by aligning time series modalities with textual tokens. However, these methods often suffer from high computational overhead due to cross-modality alignment. More recently, native LTSMs like Timer [Liu *et al.*, 2024d] have emerged, employing a decoder-only Transformer architecture trained via next-token prediction on large-scale heterogeneous datasets. Timer unifies various forecasting tasks into a single generative framework, demonstrating strong zero-shot generalization capabilities. Building on this, Timer-XL [Liu *et al.*, 2024c] addresses the context bottleneck by introducing *TimeAttention* to capture fine-grained intra- and inter-series dependencies, expanding the context window to thousands of tokens.

Despite these architectural advancements, both Timer and Timer-XL adhere to a recomputation-based inference protocol. To forecast a future horizon, these models require processing a fixed lookback window from scratch at each step. This rigid mechanism incurs quadratic computational complexity $\mathcal{O}(TL^2)$ with respect to the context length L and the forecast horizon T, severely limiting their efficiency when handling extremely long historical contexts in real-time streaming scenarios.

3 Approach

3.1 Preliminaries

We construct our streaming framework upon the foundation of LTSMs, specifically adopting the decoder-only Transformer architecture employed by Timer [Liu *et al.*, 2024d] and Timer-XL [Liu *et al.*, 2024c]. Unlike traditional deep forecasters that rely on encoder-decoder structures, Timer and Timer-XL treat time series forecasting as a native sequence generation task.

Patch-based Tokenization. Given a raw univariate time series $\mathcal{X} = \{x_1, \ldots, x_L\} \in \mathbb{R}^L$, the model first transforms the continuous data points into a sequence of discrete embeddings. We adopt a patching strategy where non-overlapping windows of size P are aggregated into tokens. The resulting sequence $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_N)$ consists of $N = L/P$ tokens, where each $\mathbf{s}_t \in \mathbb{R}^P$ represents a local temporal segment. These segments are then projected into a high-dimensional latent space $\mathbb{R}^D$ via a learnable linear embedding $\mathbf{W}_{\text{emb}}$:

$$\mathbf{h}_t^{(0)} = \mathbf{s}_t \mathbf{W}_{\text{emb}} + \mathbf{PE}_t, \tag{1}$$

where $\mathbf{PE}_t$ denotes the positional encoding required to preserve temporal ordering.

Causal Contextualization. The core computation occurs within a stack of K decoder-only Transformer blocks. To ensure the autoregressive property, a strict causal mask $\mathbf{M}$ is applied to the self-attention mechanism. For the k-th layer, the representation $\mathbf{h}_t^{(k)}$ is updated by attending solely to historical tokens:

$$\mathbf{H}^{(k)} = \text{FeedForward}\left(\text{MaskedAttn}\left(\mathbf{H}^{(k-1)}, \mathbf{M}\right)\right), \tag{2}$$

where $\mathbf{M}_{ij} = -\infty$ if $j > i$. This mechanism guarantees that the information flow at step t is strictly isolated from future signals.

Generative Objective. The training paradigm mirrors that of LLMs, formulating forecasting as a next token prediction (NTP) problem. The joint probability of the time series is factorized into a product of conditional probabilities:

$$p(\mathbf{S}) = \prod_{t=1}^{N} p(\mathbf{s}_{t+1} | \mathbf{s}_1, \ldots, \mathbf{s}_t). \tag{3}$$

In the continuous domain of time series, this probabilistic objective manifests as minimizing the reconstruction error between the predicted next patch and the ground truth. The model outputs the prediction $\hat{\mathbf{s}}_{t+1}$ via a linear projection head $\mathbf{W}_{\text{head}}$ applied to the final hidden state $\mathbf{h}_t^{(K)}$. The optimization objective is defined as the Mean Squared Error (MSE) over the sequence:

$$\mathcal{L}_{\text{NTP}} = \frac{1}{N}\sum_{t=1}^{N-1} \|\hat{\mathbf{s}}_{t+1} - \mathbf{s}_{t+1}\|_2^2. \tag{4}$$

Through this formulation, the model learns to synthesize future trajectories based on the unified history representations, establishing the theoretical basis for our subsequent streaming optimizations.

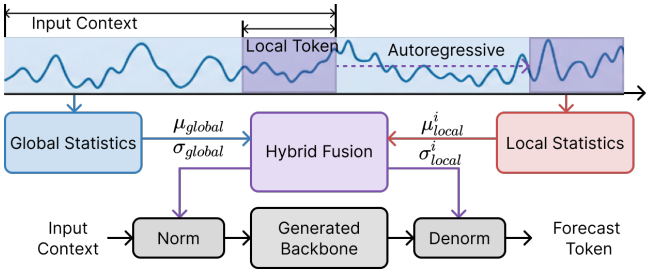

Figure 1: Proposed Hybrid-Scale Normalization Module.

3.2 Hybrid-Scale Normalization for Patch-based Generators

RevIN [Kim *et al.*, 2021] effectively mitigates distribution shifts by normalizing the input using the mean and variance over the entire lookback window. However, for patch-based autoregressive models like Timer, relying solely on global statistics may overlook rapid and high-frequency fluctuations within individual patches. Conversely, normalizing strictly at the patch level risks discarding the global amplitude information necessary for accurate trajectory reconstruction.

To address this dilemma, we introduce a Hybrid-Scale Normalization strategy that dynamically calibrates the distribution by fusing global and local statistics. Let $\mathbf{X} \in \mathbb{R}^{L\times C}$ denote the input lookback window with length L and C variables. We first compute the global statistics across the temporal dimension:

$$\boldsymbol{\mu}_{\text{global}} = \frac{1}{L}\sum_{t=1}^{L}\mathbf{x}_t, \quad \boldsymbol{\sigma}_{\text{global}} = \sqrt{\frac{1}{L}\sum_{t=1}^{L}(\mathbf{x}_t - \boldsymbol{\mu}_{\text{global}})^2 + \epsilon}. \tag{5}$$

Simultaneously, for the i-th patch $\mathbf{s}_i \in \mathbb{R}^{P\times C}$, where P is the patch size, we compute its local statistics $\boldsymbol{\mu}_{\text{local}}^{(i)}$ and $\boldsymbol{\sigma}_{\text{local}}^{(i)}$.

To balance the stability of global trends with the sensitivity to local variations, we define the normalization statistics as a convex combination of both scales, controlled by a mixing coefficient α:

$$\tilde{\boldsymbol{\mu}}^{(i)} = \alpha\cdot\boldsymbol{\mu}_{\text{local}}^{(i)} + (1-\alpha)\cdot\boldsymbol{\mu}_{\text{global}}, \tag{6}$$

$$\tilde{\boldsymbol{\sigma}}^{(i)} = \alpha\cdot\boldsymbol{\sigma}_{\text{local}}^{(i)} + (1-\alpha)\cdot\boldsymbol{\sigma}_{\text{global}}. \tag{7}$$

Each input patch $\mathbf{s}_i$ is then normalized as $\bar{\mathbf{s}}_i = (\mathbf{s}_i - \tilde{\boldsymbol{\mu}}^{(i)})/\tilde{\boldsymbol{\sigma}}^{(i)}$. Crucially, to ensure consistency, the generative output $\hat{\mathbf{s}}_i$ is denormalized using the same hybrid statistics:

$$\hat{\mathbf{x}}_i = \hat{\mathbf{s}}_i\cdot\tilde{\boldsymbol{\sigma}}^{(i)} + \tilde{\boldsymbol{\mu}}^{(i)}. \tag{8}$$

In our implementation, α is a learnable parameter initialized to 0.5. This design allows the model to remain robust to non-stationary drifts while retaining essential global context, thereby enhancing the generation stability of Timer and Timer-XL. We provide detailed empirical and theoretical verification of this choice in Appendix C.

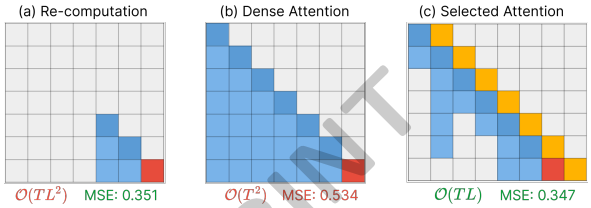

Figure 2: Illustration of proposed KV cache technique.

3.3 Linear-Complexity Streaming via KV Caching

The standard inference paradigm of Transformers typically involves a recomputation strategy, where the entire history is processed repeatedly to generate the next token. For a sequence of length L, this results in a computational complexity of $\mathcal{O}(L^2)$, rendering long-context forecasting prohibitively expensive. To address this, we implement a Streaming Inference Mechanism based on KV Caching, adapted for the patch-based architecture of Timer.

Mechanism. Instead of recalculating the representations of historical patches at every step, we cache the Key $\mathbf{K}$ and Value $\mathbf{V}$ matrices computed in previous iterations. Let $\mathbf{s}_t$ denote the input patch at step t. During the streaming process, the model only computes the Query $\mathbf{q}_t$, Key $\mathbf{k}_t$, and Value $\mathbf{v}_t$ for the current patch. The attention mechanism then attends to the concatenation of the cached states and the current states:

$$\mathbf{K}_{\text{cache}}^{(t)} = [\mathbf{K}_{\text{cache}}^{(t-1)}; \mathbf{k}_t], \tag{9}$$

$$\mathbf{V}_{\text{cache}}^{(t)} = [\mathbf{V}_{\text{cache}}^{(t-1)}; \mathbf{v}_t], \tag{10}$$

$$\text{Attention}(\mathbf{q}_t, \mathbf{K}_{\text{cache}}^{(t)}, \mathbf{V}_{\text{cache}}^{(t)}) = \text{Softmax}\left(\frac{\mathbf{q}_t(\mathbf{K}_{\text{cache}}^{(t)})^\top}{\sqrt{d_k}}\right)\mathbf{V}_{\text{cache}}^{(t)}. \tag{11}$$

Here, $[\cdot;\cdot]$ denotes concatenation along the temporal dimension. By maintaining this persistent memory state, the computation for the current step t becomes independent of the history length $t-1$ in terms of feature extraction, involving only a single forward pass of the Transformer block.

Implementation Protocol. Operationally, we reformulate the inference process as a stateful autoregressive protocol over the discretized time horizon. The procedure commences by initializing an empty Key-Value memory. Unlike standard batch processing, the model subsequently enters a streaming

phase where it ingests exclusively the current patch $\mathbf{s}_t$ at each time step t. During this incremental pass, the transformer blocks query the historical context persisted in the cache to derive the current representation $\mathbf{y}_t$, Crucially, to constrain memory growth, we employ an attention-based eviction policy: post-computation, we retain only the `[REG]` token serving as an attention sink and the subset of historical tokens with the highest attention scores. Low-scoring tokens are discarded from the memory bank. This selective retention strategy allows us to reconstruct the coherent forecast trajectory with a bounded memory overhead per step.

4 Experiments

We evaluate StreamTimer across three dimensions: (1) supervised training as a task-specific forecaster, (2) transfering as a zero-shot forecaster, and (3) the effectiveness of normalization and model efficiency. Since performance saturation on previous benchmarks [Qiu *et al.*, 2024; Wang *et al.*, 2024b] often conceals the challenges of the long-context forecasting paradigm, we establish new long-context forecasting benchmarks by following the experimental settings of Timer-XL [Liu *et al.*, 2024c]. Detailed experimental configurations are provided in Appendix A.

4.1 Univariate Time Series Forecasting

Setups To address the issue of data scarcity often encountered when extending context lengths in univariate time series [Makridakis *et al.*, 2020], we follow the experimental protocol of Timer-XL [Liu *et al.*, 2024c]. Specifically, we leverage the multivariate datasets from [Liu *et al.*, 2024b] and formulate univariate forecasting tasks by applying channel independence. The core objective of our experiments is to evaluate model robustness given long historical inputs, rather than merely pursuing extended forecasting horizons. Consequently, we fix the forecast horizon and focus on expanding the lookback window to monthly and yearly scales. Furthermore, we establish a challenging long-context univariate benchmark named ERA5-S, derived from the ECMWF Reanalysis v5 (ERA5) dataset [Muñoz-Sabater *et al.*, 2021]. Leveraging its 40-year temporal span, this benchmark is specifically designed to assess land-surface temperature forecasting capabilities based on yearly contexts.

Results As illustrated in Figure 3, increasing the context length generally leads to improved prediction accuracy across univariate datasets. While previous state-of-the-art models like Timer-XL show promising results, they tend to encounter performance saturation or even deterioration in excessively long contexts (e.g., monthly and yearly levels), likely due to the accumulation of noise. In contrast, our proposed StreamTimer exhibits superior robustness. As shown in the curves, StreamTimer consistently outperforms Timer-XL and DLinear, particularly achieving lower MSE in the longest lookback settings across all datasets. This indicates that StreamTimer effectively mitigates the impact of irrelevant history while capturing critical long-term dependencies. A similar conclusion can be drawn from the ERA5-S benchmark (Table 1), where StreamTimer surpasses both PatchTST and Timer-XL

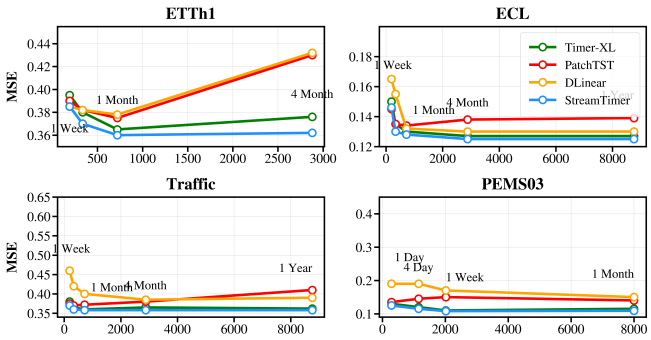

Figure 3: Univariate forecasting performance (pred-96) on established benchmarks under channel independence, with lookback lengths extended to monthly and yearly scales.

on all sites, demonstrating its efficacy in handling complex meteorological data with extended contexts.

Revisiting Non-stationary Forecasting We delve into the widespread non-stationarity in univariate tasks. While normalization techniques like RevIN [Kim *et al.*, 2021] generally benefit Transformer performance by aligning statistical distributions, their limitations become pronounced in ultra-long contexts. We argue that standard normalization oversimplifies the time-varying statistics inherent in long history. Specifically, as the context length increases, both the internal distribution shifts within the input and the statistical discrepancy between input and output are significantly exacerbated. Forcing such complex sequences into a unified distribution constrains the model capacity of Transformers. The model is compelled to handle these distorted distributions rather than learning distinct temporal variations, leading to side effects such as mode collapse and oversmoothed predictions. In contrast, our proposed normalization optimizes the distributional properties, thereby liberating the model's capacity to focus on the forecasting task itself rather than struggling with distribution alignment. As evidenced in Table 1, this approach allows Timer-XL and Timer to achieve superior results on ERA5-S without the pitfalls of standard normalization.

4.2 Multivariate Time Series Forecasting

Setups Following the experimental protocols of Timer-XL [Liu *et al.*, 2024b], we evaluate multivariate forecasting performance. To advance the development of generated transformers, we focus on the rolling forecast setting, where a single trained model adapts to various prediction horizons by recursively integrating previous predictions into the lookback window for the next iteration. Furthermore, we establish a long-context multivariate benchmark, the ERA5 multi-station land-surface temperature prediction (ERA5-MS). This benchmark is designed to facilitate the learning of complex temporal dynamics and inter-variable correlations with sufficient training samples.

Results As shown in Table 2, StreamTimer achieves the best results on the established benchmarks. Our comparison focuses primarily on its predecessors, Timer and Timer-XL.

Station	Beijing		Hongkong		London		New York		Paris		Seoul		Shanghai		**Average**	
Model	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Timer	0.076	0.212	0.189	0.321	0.277	0.414	0.185	0.332	0.262	0.413	0.090	0.227	0.136	0.285	0.174	0.315
+ RevIN	0.076	0.211	0.192	0.324	0.277	0.415	0.187	0.334	0.268	0.409	0.090	0.227	0.137	0.285	0.175	0.315
+ Ours	**0.073**	**0.208**	0.181	0.315	0.269	0.409	0.182	0.330	0.261	0.408	**0.087**	**0.224**	**0.130**	**0.278**	0.169	0.310
Timer-XL	0.074	0.210	0.179	0.316	0.262	0.404	0.182	0.327	0.254	0.399	0.090	0.229	0.134	0.282	0.168	0.310
+ RevIN	0.074	0.210	0.183	0.317	0.278	0.418	**0.181**	0.330	0.264	0.407	0.090	0.227	0.133	0.281	0.172	0.313
+ Ours	**0.073**	**0.208**	**0.173**	**0.312**	**0.260**	**0.403**	**0.180**	**0.317**	**0.251**	**0.396**	0.088	**0.224**	**0.130**	0.279	**0.165**	**0.306**

Table 1: Univariate forecasting (input-3072-pred-96) of ERA5-S, encompassing 117k time points in each station (40-years). We compare the original **Timer** and **Timer-XL** with versions using standard normalization (+ RevIN) and our proposed normalization (+ Ours).

Models		**StreamTimer (Ours)**		Timer-XL (2025)		Timer (2024)		UniTST (2024)		iTransformer (2023)		DLinear (2023)		PatchTST (2022)		TimesNet (2022)		Stationary (2022)		Autoformer (2011)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.347**	**0.387**	0.364	0.397	0.371	0.404	0.379	0.415	0.387	0.418	0.369	0.400	0.373	0.403	0.452	0.463	0.452	0.478	0.467	0.499
	192	**0.376**	**0.407**	0.405	0.424	0.407	0.429	0.415	0.438	0.416	0.437	0.405	0.422	0.405	0.425	0.474	0.477	0.484	0.510	0.492	0.523
	336	**0.389**	**0.417**	0.427	0.439	0.434	0.445	0.440	0.454	0.434	0.450	0.435	0.445	0.423	0.440	0.493	0.489	0.511	0.522	0.519	0.531
	720	**0.399**	**0.434**	0.439	0.459	0.461	0.466	0.482	0.482	0.447	0.473	0.493	0.508	0.445	0.471	0.560	0.534	0.571	0.543	0.589	0.560
	Avg	**0.378**	**0.411**	0.409	0.430	0.418	0.436	0.429	0.447	0.421	0.445	0.426	0.444	0.412	0.435	0.495	0.491	0.505	0.513	0.517	0.528
ETTh2	96	**0.275**	**0.339**	0.277	0.343	0.285	0.344	0.343	0.398	0.304	0.362	0.305	0.371	0.289	0.347	0.340	0.374	0.348	0.403	0.358	0.397
	192	**0.336**	**0.376**	0.348	0.391	0.365	0.400	0.376	0.420	0.372	0.407	0.412	0.439	0.360	0.393	0.402	0.414	0.408	0.448	0.435	0.451
	336	**0.356**	**0.407**	0.375	0.418	0.412	0.440	0.399	0.435	0.418	0.440	0.527	0.508	0.389	0.420	0.452	0.452	0.424	0.457	0.454	0.475
	720	0.401	0.445	0.409	0.458	0.468	0.487	0.419	0.457	0.463	0.476	0.830	0.653	**0.398**	**0.440**	0.462	0.468	0.448	0.476	0.479	0.492
	Avg	**0.342**	**0.392**	0.352	0.402	0.382	0.418	0.384	0.428	0.389	0.421	0.518	0.493	0.359	0.400	0.414	0.427	0.407	0.446	0.431	0.454
ETTm1	96	**0.275**	**0.335**	0.290	0.341	0.281	0.338	0.289	0.348	0.311	0.365	0.307	0.350	0.285	0.346	0.338	0.375	0.414	0.414	0.466	0.466
	192	**0.322**	**0.365**	0.337	0.369	0.330	0.368	0.332	0.375	0.353	0.390	0.337	0.368	0.329	0.372	0.371	0.387	0.524	0.482	0.504	0.496
	336	**0.358**	**0.388**	0.374	0.392	0.367	0.393	0.365	0.397	0.387	0.411	0.366	0.387	0.363	0.394	0.410	0.411	0.541	0.497	0.574	0.530
	720	**0.416**	0.422	0.437	0.428	0.432	0.433	0.421	0.431	0.452	0.445	0.419	**0.419**	0.421	0.426	0.478	0.450	0.578	0.509	0.596	0.558
	Avg	**0.343**	**0.378**	0.359	0.382	0.352	0.383	0.352	0.388	0.376	0.403	0.357	0.381	0.349	0.385	0.399	0.406	0.514	0.475	0.535	0.512
ETTm2	96	**0.166**	**0.256**	0.175	0.257	0.175	0.257	0.171	0.260	0.183	0.272	0.167	0.263	0.172	0.259	0.187	0.267	0.237	0.306	0.255	0.339
	192	**0.228**	0.288	0.242	0.301	0.239	0.301	0.228	**0.230**	0.250	0.315	0.230	0.311	0.233	0.299	0.249	0.309	0.330	0.387	0.279	0.335
	336	0.283	**0.330**	0.293	0.337	0.293	0.342	0.282	0.336	0.311	0.356	0.298	0.361	**0.280**	0.331	0.321	0.351	0.404	0.424	0.331	0.374
	720	0.368	0.387	0.376	0.390	0.392	0.407	0.380	0.398	0.417	0.419	0.432	0.446	**0.357**	**0.382**	0.497	0.403	0.525	0.486	0.413	0.450
	Avg	**0.261**	0.315	0.271	0.322	0.275	0.327	0.265	**0.306**	0.290	0.340	0.282	0.345	0.261	0.318	0.314	0.333	0.374	0.401	0.320	0.374
ECL	96	**0.125**	**0.218**	0.127	0.219	0.129	0.221	0.130	0.225	0.133	0.229	0.138	0.238	0.132	0.232	0.184	0.288	0.185	0.287	0.256	0.357
	192	**0.141**	**0.233**	0.145	0.236	0.148	0.239	0.150	0.244	0.158	0.258	0.152	0.251	0.151	0.250	0.192	0.295	0.282	0.368	0.291	0.376
	336	**0.152**	**0.246**	0.159	0.252	0.164	0.256	0.166	0.262	0.168	0.262	0.167	0.268	0.171	0.272	0.200	0.303	0.289	0.377	0.290	0.379
	720	**0.177**	**0.270**	0.187	0.277	0.201	0.289	0.206	0.297	0.205	0.294	0.203	0.302	0.222	0.318	0.228	0.325	0.305	0.399	0.320	0.403
	Avg	**0.149**	**0.242**	0.155	0.246	0.161	0.251	0.163	0.257	0.164	0.258	0.165	0.265	0.169	0.268	0.201	0.303	0.265	0.358	0.289	0.379
Traffic	96	0.341	**0.236**	**0.340**	0.238	0.348	0.240	0.359	0.250	0.353	0.259	0.399	0.285	0.359	0.255	0.593	0.315	0.610	0.322	0.675	0.412
	192	**0.358**	**0.241**	0.360	0.247	0.369	0.250	0.373	0.257	0.373	0.267	0.409	0.290	0.377	0.265	0.596	0.317	0.626	0.346	0.679	0.423
	336	**0.374**	**0.251**	0.377	0.256	0.388	0.260	0.386	0.265	0.386	0.275	0.422	0.297	0.393	0.276	0.600	0.319	0.633	0.352	0.688	0.440
	720	**0.412**	**0.275**	0.418	0.279	0.431	0.285	0.421	0.286	0.425	0.296	0.461	0.319	0.436	0.305	0.619	0.335	0.651	0.366	0.693	0.457
	Avg	**0.371**	**0.251**	0.374	0.255	0.384	0.259	0.385	0.265	0.385	0.274	0.423	0.298	0.391	0.275	0.602	0.322	0.630	0.347	0.684	0.433
Weather	96	**0.148**	**0.200**	0.157	0.205	0.151	0.202	0.152	0.206	0.174	0.225	0.169	0.229	0.149	0.202	0.169	0.228	0.185	0.241	0.355	0.409
	192	**0.191**	**0.238**	0.206	0.250	0.196	0.245	0.198	0.249	0.227	0.268	0.211	0.268	0.194	0.245	0.222	0.269	0.286	0.325	0.421	0.450
	336	**0.241**	**0.279**	0.259	0.291	0.249	0.288	0.251	0.291	0.290	0.309	0.258	0.306	0.244	0.285	0.290	0.310	0.323	0.347	0.452	0.465
	720	**0.311**	0.335	0.337	0.344	0.330	0.344	0.322	0.340	0.374	0.360	0.320	0.362	0.317	**0.329**	0.376	0.364	0.436	0.401	0.513	0.496
	Avg	**0.223**	**0.263**	0.240	0.273	0.232	0.270	0.231	0.272	0.266	0.291	0.239	0.291	0.226	0.268	0.264	0.293	0.308	0.329	0.435	0.455
Solar-Energy	96	**0.159**	**0.216**	0.162	0.221	0.212	0.230	0.190	0.240	0.183	0.265	0.193	0.258	0.168	0.237	0.180	0.272	0.199	0.290	0.206	0.296
	192	**0.183**	**0.236**	0.187	0.239	0.232	0.246	0.223	0.264	0.205	0.283	0.214	0.274	0.189	0.257	0.199	0.286	0.243	0.307	0.254	0.328
	336	**0.203**	**0.251**	0.205	0.255	0.237	0.253	0.250	0.283	0.224	0.299	0.233	0.291	0.212	0.277	0.220	0.301	0.264	0.322	0.272	0.330
	720	**0.233**	0.276	0.238	0.279	0.252	**0.266**	0.292	0.311	0.239	0.316	0.246	0.307	0.240	0.305	0.251	0.321	0.310	0.339	0.326	0.347
	Avg	**0.195**	**0.245**	0.198	0.249	0.233	0.249	0.241	0.275	0.213	0.291	0.222	0.283	0.202	0.269	0.213	0.295	0.254	0.315	0.265	0.325
1st Count		**36**	**33**	1	0	0	1	0	2	0	0	0	1	3	3	0	0	0	0	0	0

Table 2: Multivariate forecasting results: we conduct rolling forecast with a single model trained on each dataset. Lookback length is 672, and forecast lengths are in $\{96, 192, 336, 720\}$. **Blue**: the best, underline: the second best.

While Timer-XL already demonstrates strong capabilities over Timer by leveraging extended contexts, StreamTimer pushes the performance boundary even further. It is particularly noteworthy that StreamTimer significantly outperforms Timer-XL, especially on non-stationary time series (improving by 8.2% in ETTh1). We attribute this substantial improvement over Timer-XL to two key innovations: first, our proposed normalization effectively resolves the distribution shift issues that hinder Timer-XL in ultra-long contexts; second, the streaming forecasting mechanism enhances stability during continuous inference. Furthermore, we find StreamTimer achieves 69 wins out of 80 tests under different metrics and conditions, and maintains strong performance even on low-dimensional datasets with low predictability (e.g., ETT), un-

Model	Beijing		Hongkong		London		New York		Paris		Seoul		Shanghai		**Average**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PatchTST	0.0815	0.222	0.190	0.326	0.275	0.414	0.185	0.333	0.265	0.407	0.0977	0.240	0.139	0.290	0.176	0.319
UniTST	0.0753	0.213	0.179	0.318	0.269	0.410	0.185	0.330	0.256	0.401	0.0901	0.230	0.135	0.284	0.170	0.312
Timer	0.0734	0.210	0.182	0.319	0.268	0.407	0.183	0.329	0.255	0.399	0.0877	0.226	0.132	0.281	0.169	0.310
Timer-XL	0.0736	0.209	0.174	0.309	0.263	0.404	0.182	0.327	0.252	0.396	0.0872	0.225	**0.130**	0.278	0.166	0.307
StreamTimer	**0.0732**	**0.207**	**0.172**	**0.306**	**0.260**	**0.401**	**0.181**	**0.325**	**0.251**	**0.394**	**0.0868**	**0.223**	**0.130**	**0.277**	**0.165**	**0.304**

Table 3: Multivariate forecasting (input-3072-pred-96) of ERA5-MS (40 years and 7 stations). We fairly evaluate Transformers that adopt patched time series. Different from ERA5-S, we train one model for all sites.

Models		Timer										Timer-XL									
		+RevIN		+SAN		+Dish-TS		+MPNorm		+Ours		+RevIN		+SAN		+Dish-TS		+MPNorm		+Ours	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.371	0.404	0.363	0.399	0.363	0.396	0.361	0.397	**0.347**	**0.387**	0.364	0.397	0.370	0.401	0.423	0.455	0.368	0.401	**0.352**	**0.392**
	192	0.407	0.429	0.395	0.418	0.400	0.418	0.396	0.422	**0.378**	**0.405**	0.405	0.424	0.412	0.427	0.490	0.496	0.397	0.422	**0.380**	**0.410**
	336	0.434	0.445	0.416	0.430	0.425	0.435	0.411	0.426	**0.392**	**0.421**	0.427	0.439	0.444	0.447	0.542	0.529	0.422	0.445	**0.393**	**0.421**
	720	0.461	0.466	0.456	0.464	0.451	0.458	0.454	0.462	**0.398**	**0.433**	0.439	0.459	0.472	0.475	0.663	0.614	0.447	0.451	**0.403**	**0.437**
	Avg	0.418	0.436	0.407	0.428	0.410	0.427	0.405	0.427	**0.382**	**0.416**	0.409	0.430	0.424	0.438	0.530	0.523	0.409	0.430	**0.382**	**0.415**
Weather	96	0.151	0.202	0.158	0.210	0.158	0.211	0.157	0.210	**0.149**	**0.201**	0.157	0.205	0.157	0.210	0.161	0.220	0.156	0.208	**0.152**	**0.201**
	192	0.196	0.245	0.203	0.252	0.204	0.254	0.196	0.249	**0.194**	**0.240**	0.206	0.250	0.202	0.251	0.214	0.267	0.200	0.248	**0.196**	**0.243**
	336	0.249	0.288	0.250	0.289	0.257	0.294	0.248	**0.286**	**0.244**	0.287	0.259	0.291	0.248	0.288	0.270	0.311	0.247	0.286	**0.245**	**0.281**
	720	0.330	0.344	0.313	**0.335**	0.331	0.347	0.318	0.341	**0.312**	0.339	0.337	0.344	**0.312**	0.335	0.343	0.367	0.313	0.336	0.314	**0.330**
	Avg	0.232	0.270	0.231	0.271	0.237	0.276	0.230	0.271	**0.225**	**0.267**	0.240	0.272	0.230	0.271	0.247	0.291	0.229	0.270	**0.227**	**0.264**

Table 4: Ablation study of different modules on Timer and Timer-XL. **+RevIN** columns contain the real baseline results from the main experiment. **+SAN**, **Dish-TS**, and **+MPNorm** denote adding the specific modules. **+Ours** denotes our normalization module.

derscoring its effectiveness and superiority.

Performance on ERA5-MS To evaluate the model's capability in multivariate time series forecasting, we conducted experiments on the real-world ERA5-MS benchmark as shown in Table 3. We compared StreamTimer with strong baselines ranging from channel-independent approaches PatchTST, UniTST to large-scale generative models Timer, Timer-XL. The results indicate that StreamTimer effectively combines the benefits of advanced tokenization and architectural design, surpassing the Timer-XL. By consistently achieving the best metrics in both MSE and MAE across diverse stations, StreamTimer proves to be the most effective solution for multivariate forecasting in this setting.

Task	StreamTimer		Timer-XL		Timer		PatchTST	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1 → ETTh2	**0.345**	0.388	0.367	0.398	0.354	**0.385**	0.386	0.400
ETTh1 → ETTm2	**0.273**	**0.344**	0.302	0.361	0.277	0.347	0.314	0.360
ETTh2 → ETTh1	**0.485**	**0.485**	0.508	0.499	0.588	0.537	0.565	0.509
ETTh2 → ETTm2	**0.282**	**0.342**	0.303	0.357	0.285	0.347	0.289	0.345
ETTm1 → ETTh2	**0.377**	**0.412**	0.402	0.429	0.395	0.415	0.447	0.434
ETTm1 → ETTm2	**0.231**	**0.294**	0.268	0.324	0.235	0.297	0.258	0.311
ETTm2 → ETTh2	**0.354**	**0.394**	0.358	0.396	0.394	0.410	0.411	0.418
ETTm2 → ETTm1	0.481	0.455	0.483	**0.446**	**0.477**	0.459	0.530	0.470

Table 5: Zero-shot forecasting results. **Bold** indicates the best result, and underline indicates the second best.

Zero Shot Forecasting Results To evaluate the generalization capability of StreamTimer on unseen datasets, we conducted zero-shot forecasting experiments. Following previous protocols [Jin *et al.*, 2024; Kudrat *et al.*, 2025], we selected ETTh1, ETTh2, ETTm1, and ETTm2 as source datasets and performed direct inference on the remaining datasets as targets. Table 5 presents the results for a prediction horizon of 192, where StreamTimer demonstrates consistent superiority across transfer tasks, significantly outperforming PatchTST and the baseline models, Timer and Timer-XL. This robustness across different granularities and domains is primarily attributed to StreamTimer's streaming dynamic adaptation mechanism. This mechanism enables the model to keenly capture and adapt to the distribution evolution of the target domain during the inference phase, thereby effectively mitigating the distribution shift challenges between the source and unseen target domains.

Ablation on Hybrid-scale Normalization To verify the effectiveness of our proposed normalization strategy, we conducted a comprehensive ablation study comparing it with several state-of-the-art normalization techniques, including RevIN [Kim *et al.*, 2021], SAN [Liu *et al.*, 2023], Dish-TS [Fan *et al.*, 2023], and MPNorm [Zhang and Dong, 2025]. Table 4 reports the results on ETTh1 and Weather datasets using both Timer and Timer-XL as backbones. The results demonstrate that our proposed module consistently achieves the lowest MSE and MAE across almost all prediction lengths. While RevIN and MPNorm serve as strong baselines, our method further boosts performance, particularly on the ETTh1 dataset where non-stationarity is significant. Notably, while SAN and Dish-TS employ model-based statistics prediction that necessitates complex pre-processing and auxiliary modules, this added complexity yields negligible performance gains.

Ablation on KV Cache As shown in Table 6, our ablation study uncovers that RevIN induces severe distribution shift during KV cache streaming. Specifically, RevIN's error rises

Backbone		Timer				Timer-XL			
Normalization		+ RevIN		+ Ours		+ RevIN		+ Ours	
Dataset	Horizon	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	196	0.420	0.439	0.377	0.407	0.425	0.441	0.382	0.412
	336	0.462	0.478	0.389	0.417	0.456	0.477	0.392	0.424
	720	0.524	0.540	0.399	0.434	0.531	0.547	0.403	0.438
Weather	196	0.238	0.275	0.194	0.240	0.241	0.274	0.195	0.242
	336	0.383	0.365	0.245	0.285	0.391	0.370	0.244	0.280
	720	0.491	0.430	0.312	0.339	0.485	0.427	0.311	0.333

Table 6: Ablation study on ETTh1 and Weather datasets using KV cache streaming. We compare +RevIN and +Ours methods based on Timer and Timer-XL backbones.

markedly as the horizon extends to 720, reflecting its failure to manage statistical variations in long streams. On the contrary, our method maintains a stable and low MSE, verifying that it causes almost no harm to model performance in streaming scenarios.

4.3 Model Analysis

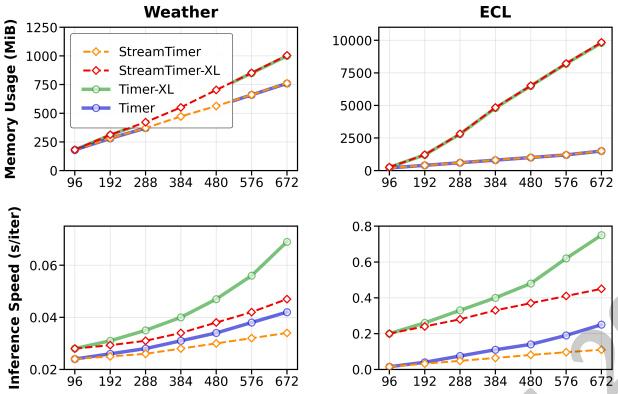

Figure 4: Efficiency analysis. We compare StreamTimer with baselines on multivariate datasets with variable numbers ranging from ten to hundred and increase the forecast horizon.

Model Efficiency To comprehensively evaluate model efficiency, it is essential to consider the input context length T, the number of variables N, and crucially, the prediction horizon L. Unlike 1D sequences, the computational overhead in multivariate time series forecasting is dictated by the interplay of these three dimensions. We provide a theoretical complexity analysis comparing StreamTimer against the baseline models, Timer and Timer-XL. Specifically, the complexity of Timer, which adopts channel independence, scales as $\mathcal{O}(NT^2L)$. Meanwhile, Timer-XL, which incorporates channel dependence to capture inter-variable correlations, sees its complexity escalate to $\mathcal{O}(N^2T^2L)$. Crucially, the computational burden of both architectures involves a multiplicative coupling with the prediction horizon L and quadratic terms of T, leading to significant overhead accumulation in long-term forecasting scenarios. In contrast, leveraging the streaming forecasting mechanism, StreamTimer effectively avoids the redundant re-computation of Keys and Values. As derived in Table 10, StreamTimer significantly optimizes the overall complexity to $\mathcal{O}(NTL)$. This indicates that the computational cost of StreamTimer scales linearly with the total sequence length, rather than multiplicatively. As shown in Figure 4, by eliminating these redundant operations, StreamTimer demonstrates an order-of-magnitude efficiency advantage, particularly in tasks involving extensive contexts and extended prediction horizons.

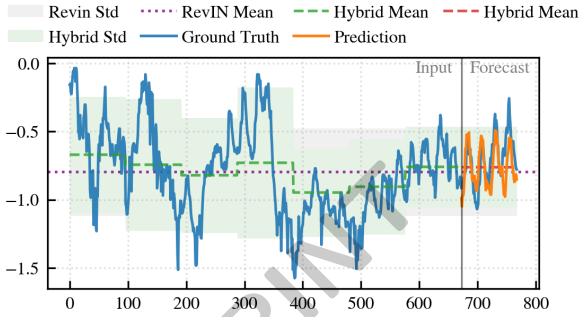

Figure 5: Visualization of proposed normalization.

Visualization of Hybrid-Scale Normalization Figure 5 visualizes the effectiveness of the proposed normalization strategy. The blue and orange solid lines represent the ground truth and the model prediction, respectively. While RevIN relies on a static global mean (purple dotted line), which fails to capture temporal variations, our Hybrid-Scale Normalization utilizes local means (green and red dashed lines) that dynamically track local trends. Specifically, the red dashed line indicates that the model successfully infers the future distribution mean based on historical patterns. The alignment between the local statistics and the actual data demonstrates the method's precision in mitigating the impact of non-stationarity.

5 Conclusions

In this paper, we propose StreamTimer, a streaming inference framework designed to address the limitations of fixed lookback windows in time series forecasting. By reformulating the forecasting process into a continuous state update mechanism, we effectively leverage extremely long historical information without incurring additional computational burden. Our core technical innovations—particularly Cache-Consistent Normalization tailored for non-stationary data—effectively overcome the domain-specific barriers to applying KV caching in time series. Extensive experiments validate the efficiency of StreamTimer, demonstrating substantial reductions in both computational and memory costs while matching or even exceeding predictive performance. These results prove that decoupling context length from computational cost via streaming is both feasible and efficient, paving a new direction for the development of ultra-long sequence modeling and Time Series Foundation Models.

Acknowledgments

This work was supported in part by the Zhejiang Provincial Science and Technology Plan Project under Grant 2024C03261 and in part by Key Scientific Research Base for Digital Conservation of Cave Temples(Zhejiang University), National Cultural Heritage Administration.

Contribution Statement

Xiyu Meng and Yuhan Wu contributed equally to this work. Yabo Dong is the corresponding author.

References

[Ansari and Stella, 2024] Abdul Fatir Ansari and Stella. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024.

[Chang *et al.*, 2024] Ching Chang, Wei-Yao Wang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters, 2024.

[Fan *et al.*, 2023] Wei Fan, Pengyang Wang, Dongkun Wang, Dongjie Wang, Yuanchun Zhou, and Yanjie Fu. Dish-ts: A general paradigm for alleviating distribution shift in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7522–7529, 2023.

[Jin *et al.*, 2024] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-llm: Time series forecasting by reprogramming large language models, 2024.

[Kim *et al.*, 2021] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

[Kudrat *et al.*, 2025] Dilfira Kudrat, Zongxia Xie, Yanru Sun, Tianyu Jia, and Qinghua Hu. Patch-wise structural loss for time series forecasting. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 31841–31859. PMLR, 13–19 Jul 2025.

[Lin *et al.*, 2023] Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. Segrnn: Segment recurrent neural network for long-term time series forecasting, 2023.

[Lin *et al.*, 2024] Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with 1k parameters. In *Forty-first International Conference on Machine Learning*, 2024.

[Liu *et al.*, 2023] Zhiding Liu, Mingyue Cheng, Zhi Li, Zhenya Huang, Qi Liu, Yanhu Xie, and Enhong Chen. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[Liu *et al.*, 2024a] Xu Liu, Juncheng Liu, Gerald Woo, Taha Aksu, Yuxuan Liang, Roger Zimmermann, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Moirai-moe: Empowering time series foundation models with sparse mixture of experts. *arXiv preprint arXiv:2410.10469*, 2024.

[Liu *et al.*, 2024b] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting, 2024.

[Liu *et al.*, 2024c] Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer-xl: Long-context transformers for unified time series forecasting. *arXiv preprint arXiv:2410.04803*, 2024.

[Liu *et al.*, 2024d] Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *Forty-first International Conference on Machine Learning*, 2024.

[Luo and Wang, 2024] Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations*, 2024.

[Makridakis *et al.*, 2020] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020.

[Muñoz-Sabater *et al.*, 2021] Joaquín Muñoz-Sabater, Emanuel Dutra, Anna Agustí-Panareda, Clément Albergel, Gabriele Arduini, Gianpaolo Balsamo, Souhail Boussetta, Margarita Choulga, Shaun Harrigan, Hans Hersbach, et al. Era5-land: A state-of-the-art global reanalysis dataset for land applications. *Earth system science data*, 13(9):4349–4383, 2021.

[Nie *et al.*, 2023] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[Qiu *et al.*, 2024] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods, 2024.

[Shi *et al.*, 2025] Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts, 2025.

[Wang *et al.*, 2024a] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing

for time series forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[Wang *et al.*, 2024b] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. *CoRR*, abs/2407.13278, 2024.

[Woo *et al.*, 2024] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024.

[Wu *et al.*, 2024] Yuhan Wu, Xiyu Meng, Yang He, Junru Zhang, Haowen Zhang, Yabo Dong, and Dongming Lu. Multi-view self-supervised contrastive learning for multivariate time series. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 9582–9590, 2024.

[Wu *et al.*, 2025] Yuhan Wu, Xiyu Meng, Huajin Hu, Junru Zhang, Yabo Dong, and Dongming Lu. Affirm: Interactive mamba with adaptive fourier filters for long-term time series forecasting. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 21599–21607. AAAI Press, 2025.

[Xiao *et al.*, 2024] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*, 2024.

[Zeng *et al.*, 2023] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 11121–11128. AAAI Press, 2023.

[Zhang and Dong, 2025] Jiayu Zhang and Yuantong Dong. Multi-period normalization for long-term time series forecasting. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025.

[Zhou *et al.*, 2021] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11106–11115. AAAI Press, 2021.

[Zhou *et al.*, 2023] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained LM. In *NeurIPS*, 2023.