TimeEmb: A Lightweight Static-Dynamic Disentanglement Framework for Time Series Forecasting

Mingyuan Xia[*]
Jilin University
xiamy2322@mails.jlu.edu.cn

Chunxu Zhang[*]
Jilin University
zhangchunxu@jlu.edu.cn

Zijian Zhang[†]
Jilin University
zhangzijian@jlu.edu.cn

Hao Miao
The Hong Kong Polytechnic University
hao.miao@polyu.edu.hk

Qidong Liu
Xi'an Jiaotong University
liuqidong@xjtu.edu.cn

Yuanshao Zhu
City University of Hong Kong
yuanshao@ieee.org

Bo Yang
Jilin University
ybo@jlu.edu.cn

Abstract

Temporal non-stationarity, the phenomenon that time series distributions change over time, poses fundamental challenges to reliable time series forecasting. Intuitively, the complex time series can be decomposed into two factors, *i.e.,* time-invariant and time-varying components, which indicate static and dynamic patterns, respectively. Nonetheless, existing methods often conflate the time-varying and time-invariant components, and jointly learn the combined long-term patterns and short-term fluctuations, leading to suboptimal performance facing distribution shifts. To address this issue, we initiatively propose a lightweight static-dynamic decomposition framework, TimeEmb, for time series forecasting. TimeEmb innovatively separates time series into two complementary components: (1) time-invariant component, captured by a novel global embedding module that learns persistent representations across time series, and (2) time-varying component, processed by an efficient frequency-domain filtering mechanism inspired by full-spectrum analysis in signal processing. Experiments on real-world datasets demonstrate that TimeEmb outperforms state-of-the-art baselines and requires fewer computational resources. We conduct comprehensive quantitative and qualitative analyses to verify the efficacy of static-dynamic disentanglement. This lightweight framework can also improve existing time-series forecasting methods with simple integration. To ease reproducibility, the code is available at https://github.com/showmeon/TimeEmb.

1 Introduction

The proliferation of edge devices and mobile sensing results in a large amount of time series data, enabling various real-world applications [58, 51, 15, 26]. In this study, we focus on time series forecasting, which plays a pivotal role in decision-making across critical domains including energy management [11], transportation systems [57, 7], and financial markets [44].

[*]These authors contributed equally to this work.
[†]Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

https://doi.org/10.52202/085713-2560

Traditional statistical methods, *e.g.,* ARMA [2], employ moving average techniques to model temporal dependencies. With the advances of neural networks, deep learning methods have revolutionized temporal pattern extraction, delivering superior performance. These methods include recurrent neural networks (RNNs) models [13, 35] that capture sequential dynamics, convolutional neural networks (CNNs) models [8, 18] to obtain hierarchical features, and transformer-based models [46] to learn long-range dependencies with self-attention mechanisms. Recently, Multi-layer perceptron (MLP) methods [56] have demonstrated their effectiveness and superior efficiency compared to transformer-based counterparts.

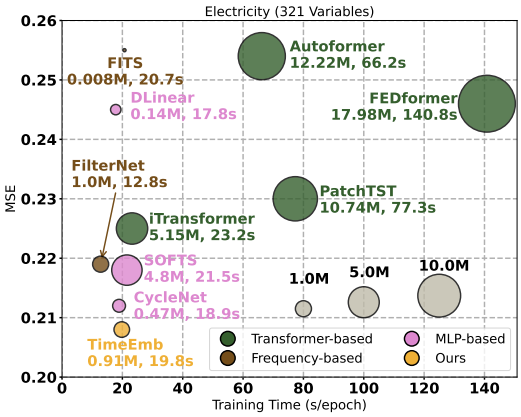


Figure 1: Efficiency and performance comparison on the Electricity dataset.

Despite the advancements of existing methods, a fundamental challenge persists in modeling complex temporal dependencies, *i.e.,* non-stationarity. Real-world time series often exhibit dynamic distribution shifts due to evolving trends and external interventions, showing high non-stationarity [32, 20]. This dynamic distribution shift violates the independent and identical distribution (IID) assumption of most existing forecasting methods [9]. It poses great challenges to robust temporal dependency modeling [38] and calls for a novel method that can handle such non-stationarity and learn comprehensive temporal dependencies.

Intuitively, time series can be considered as a combination of two complementary parts: static time-invariant and dynamic time-varying components [21, 36, 6]. **Time-invariant component** represents stable long-term patterns in time series. For example, traffic flow typically follows a regular pattern, with peaks in the morning and troughs at night. **Time-varying component** reflects local fluctuations in time series, *e.g.,* abnormal traffic flow caused by extreme weather or accidents. We contend that effective disentanglement of these two components can prevent the model from mistaking short-term noise for long-term patterns. It can explicitly capture stable long-term and dynamic local dependencies, thereby improving the effectiveness of time series forecasting.

However, it is non-trivial to develop this kind of model. In general, there remain three major limitations unsolved for time series disentanglement: **(1) Ignorance of long-term invariant pattern modeling.** Existing seasonal-trend disentanglement methods often generate the trend component by moving the average kernel, and consider the rest as the seasonal component [51, 56, 59, 25]. It is performed on local time series by smoothing [37], and can hardly learn the global static patterns in the whole time series. **(2) Rigorous assumption.** To pursue explicit disentanglement, some methods rely on strong assumptions that may not always hold in practice. For example, CycleNet [14] assumes a fixed periodic pattern in the dataset and extracts it using a learnable recurrent cycle. However, this assumption does not always hold, as the complex periodicities can vary or have diverse lengths. Moreover, relying on a pre-defined cycle length leads to limited flexibility and unstable efficacy. It cannot learn periodicity without providing an exact cycle length. **(3) High model complexity.** The quadratic complexity of the self-attention mechanism hinders practical application [51, 59]. As shown in Figure 1, Transformer-based methods exhibit relatively large model sizes and high training costs. Recent methods based on frequency analysis and MLP partially alleviate this with more efficient architectures. However, a satisfactory balance between performance and efficiency remains elusive.

To address these problems, we propose TimeEmb, a lightweight static-dynamic disentanglement framework. TimeEmb decomposes the original time series into time-invariant and time-varying components, and processes them accordingly. Specifically, we introduce a learnable time-invariant embedding bank to extract static time-invariant patterns. These embeddings are consistent across all time series segments within the entire dataset, aiming to capture long-term and stable temporal patterns. In addition, the embedding bank provides specific embedding for individual timesteps. This enables the model to adapt to local data distribution shifts since time-invariant patterns may differ at different timesteps. By separating the time-invariant component from the time series, we obtain the remaining time-varying component, illustrating dynamic disturbance. Frequency analysis describes complex signals using their intensity in the frequency spectrum [1], which presents clear intrinsic periodicity features. Inspired by this, we design an efficient frequency filter to process the

time-varying component through dense weighting. Based on the explicit decomposition and parallel processing of the static and dynamic components, TimeEmb achieves state-of-the-art performance. Meanwhile, due to its lightweight architecture, it requires fewer computational resources. As shown by its optimal position in Figure 1, the proposed TimeEmb strikes an excellent balance between performance and efficiency.

Our major contributions are summarized as follows:

- For the first time, we propose to leverage a learnable embedding bank to capture the global recurrent features while adapting to local distribution shifts.
- We propose TimeEmb, which explicitly disentangles the time series and systematically addresses the time-invariant component using a learnable embedding bank and time-varying component via frequency filtering.
- The proposed TimeEmb can easily and seamlessly serve as a plug-in to enhance existing methods with minimum additional computational cost.
- Experiments on seven benchmark datasets from diverse scenarios demonstrate the superior performance of the proposed TimeEmb. TimeEmb is efficient in terms of computation and storage compared to existing state-of-the-art baselines.

2 Related Work

Transformer-based Time Series Forecasting. Transformers have shown strong sequence modeling capabilities in time series forecasting [51, 58, 22]. PatchTST [28] segments sequences into fixed-length patches for local-global modeling, while iTransformer [22] and Informer [58] reduce attention complexity to improve scalability. However, attention-based models still incur considerable computational and memory costs [12, 43], limiting deployment in resource-constrained settings. In contrast, TimeEmb leverages lightweight spectral modules, *i.e.,* including an embedding bank and frequency filter, to achieve strong performance with reduced overhead.

MLP-based Time Series Forecasting. Recently, MLP methods, *e.g.,* TSMixer [4] and TimeMixer [49], have demonstrated competitive forecasting performance with reduced complexity. DLinear [56] further improves efficiency by separating trend and residual components. By contrast, TimeEmb provides an explicit disentanglement framework in the frequency domain, enabling simultaneous modeling of time-invariant and time-varying patterns beyond what time-domain MLPs can express.

Frequency-based Time Series Forecasting. Recent work has explored Fourier-based representations to model periodicity and reduce noise sensitivity [52, 59, 29]. While most methods apply global spectral analysis, TimeEmb introduces a fine-grained disentanglement strategy: a time-invariant component is learned across the full spectrum via embedding, while the dynamic part is filtered adaptively by a learnable frequency modulation. This structured spectral design extends the utility of frequency-domain modeling for complex time series.

Embedding-enhanced Forecasting. Embedding strategies have been adopted to encode positional, spatial, or temporal context [28, 39, 10]. For instance, STID [39] and D2STGNN [41] use spatiotemporal embeddings, while SOFTS [10] shares embeddings across channels. Unlike these, TimeEmb establishes a learnable temporal embedding bank that captures global time-invariant patterns across the dataset, with each embedding specializing in a specific time slot to model static structures in a data-driven and frequency-aware manner.

LLM-based Time Series Forecasting. With the rapid development of Large Language Models, recent research has explored their potential in time series forecasting by treating temporal signals as sequential tokens [19, 16]. LLM-based approaches benefit from strong generalization and transfer capabilities, enabling zero-shot or few-shot forecasting across diverse domains [17, 24, 48]. Nevertheless, these models are typically resource-intensive and require massive pretraining corpora, making them impractical for lightweight or domain-specific applications. Compared with these paradigms, TimeEmb focuses on a compact yet effective disentanglement mechanism that achieves comparable forecasting accuracy with substantially reduced computational cost and training complexity.

 https://doi.org/10.52202/085713-2560

3 Methodology

3.1 Framework Overview

Given historical time series $\boldsymbol{X} \in \mathbb{R}^{L \times D}$ with L timesteps and D channels, time series prediction aims to infer the future states of H timesteps, *i.e.,* $\widehat{\boldsymbol{X}} \in \mathbb{R}^{H \times D}$.

TimeEmb addresses time series forecasting via disentangled representation learning in the frequency domain. The core idea is to decompose the input sequence into a time-invariant component $\boldsymbol{X}_s$ and a time-varying component $\boldsymbol{X}_d$.

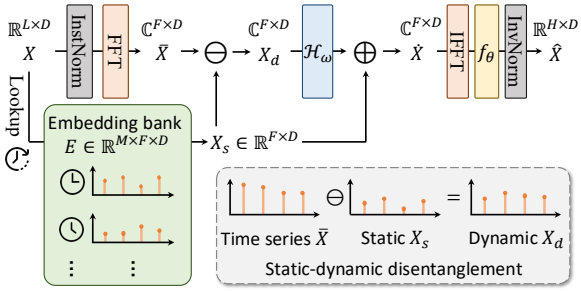


Figure 2: TimeEmb framework overview.

Specifically, we first transform the input series $\boldsymbol{X}$ into its frequency representation $\overline{\boldsymbol{X}}$ via the Fourier transform. Then, we retrieve $\boldsymbol{X}_s$ from a learnable embedding bank $\boldsymbol{E}$ based on the input timestamp, capturing long-term stable patterns. The dynamic part $\boldsymbol{X}_d$ is obtained by subtracting $\boldsymbol{X}_s$ from $\overline{\boldsymbol{X}}$. To model complex dynamics, we apply a learnable frequency filter $\mathcal{H}_\omega$ to $\boldsymbol{X}_d$, emphasizing informative frequencies and suppressing noise. The filtered dynamic and static components are then fused and transformed back to the time domain for final prediction. This frequency-based decomposition allows TimeEmb to efficiently capture both periodic structures and transient variations in a lightweight and interpretable manner.

3.2 Domain Transformation

Viewing time series data from the perspective of the frequency domain offers unique insights into its underlying structure. Unlike the time domain, where patterns may be obscured by noise or nonlinearity, the frequency spectrum reveals the distribution of different periodic components and their relative energy contributions. Transforming time series into the frequency domain decomposes it into distinct frequency components, describing the complex signal as a linear combination of sine and cosine waves with varying frequencies and amplitudes. This process helps reveal underlying periodicities and hidden features that are otherwise obscure in the time domain [42].

Given a discrete temporal sequence $\boldsymbol{X} \in \mathbb{R}^{L \times D}$ (we consider the univariate case with $D = 1$ for clarity), we first conduct instance normalization InstNorm() to standardize each instance's distribution at every timestep. Then, its frequency-domain representation $\overline{\boldsymbol{X}} \in \mathbb{C}^{F \times D}$ can be obtained using real-valued Fast Fourier Transform (rFFT) [27],

$$\overline{\boldsymbol{X}}[k] = \sum_{n=0}^{L-1} \boldsymbol{X}[n] e^{-j2\pi kn/L}, \quad k = 0, 1, ..., F-1, \tag{1}$$

where $j = \sqrt{-1}$ is the imaginary unit. Due to the conjugate symmetry property of real signals in the Fourier domain, the number of unique frequency components is $F = \lfloor L/2 \rfloor + 1$, allowing for a compact representation without redundancy.

3.3 Static Component via Embedding Bank

Existing approaches to modeling time-invariant patterns, such as seasonal-trend decomposition [51, 56], typically divide a time series into trend and residual components using local smoothing methods. However, this method merely considers locally stable and dynamic parts in the input time series, and fails to uncover the long-standing invariant features in the dataset. Recently, CycleNet [14] attempts to address this by learning a periodic embedding, but it depends on a predefined period length from expert knowledge, and slight changes can cause severe performance drops.

To address these limitations, TimeEmb proposes a flexible and learnable mechanism to capture long-term, recurrent patterns shared across time series through a temporal embedding bank. For example, in traffic forecasting, we aim to capture recurring daily structures such as typical rush-hour

patterns. Since intra-day patterns also vary over time (*e.g.,* hourly traffic flow fluctuations), we construct embeddings for each timestep.

In specific, we define a learnable embedding bank $\boldsymbol{E} \in \mathbb{R}^{M \times F \times D}$ consisting of M embeddings to preserve invariant patterns in a day. M controls the granularity of intra-day specific patterns. For instance, when $M = 24$, $\boldsymbol{E}$ assigns an embedding to each hour; when $M = 96$, it captures common patterns every 15 minutes. To guarantee the embedding learns the general pattern across the time series, we leverage the last timestep of the input $\boldsymbol{X}$ as t_{last}. This index enables us to retrieve embedding from $\boldsymbol{E}$, *i.e.,* $\boldsymbol{X}_s = \boldsymbol{E}[t_{last} \bmod M]$. Then, we separate the embedding $\boldsymbol{X}_s$ from time series $\overline{\boldsymbol{X}}$ and obtain the time-varying component $\boldsymbol{X}_d$ as follows,

$$\boldsymbol{X}_d = \overline{\boldsymbol{X}} - \boldsymbol{X}_s. \tag{2}$$

In this operation, we subtract the real number $\boldsymbol{X}_s$ from the real part of the complex number $\overline{\boldsymbol{X}}$, which can reduce the computational and storage cost of the embedding bank. The embedding bank $\boldsymbol{E}$ is optimized across the entire dataset and learns to encode consistent patterns that emerge at the same time across different days. For instance, when $M = 24$, each embedding is tuned to capture the average behavior at a specific hour of the day (*e.g.,* peaks around 8:00 and lows around 23:00), enabling the model to represent both the global temporal structure and local variations. Importantly, this embedding structure is flexible: while we focus on day-level periodicities, it can be naturally extended to model weekly or other common-sense periods by adjusting M, without relying on domain knowledge. This design enables TimeEmb to learn shared expressive representations of time-invariant components, which are essential for disentangled modeling and robust generalization.

3.4 Dynamic Component via Frequency Filtering

To effectively model the dynamic component $\boldsymbol{X}_d$, we apply a learnable spectral filter in the frequency domain. This design is motivated by the Convolution Theorem [23], *i.e.,* circular convolution in the time domain is equivalent to element-wise multiplication in the frequency domain. Thus, frequency-domain filtering provides an efficient and expressive way to implement time-invariant linear operations on temporal signals.

We introduce a complex-valued spectral modulation vector $\boldsymbol{\omega} \in \mathbb{C}^{F \times 1}$, shared across channels, to selectively reweight different frequency bands. The filtering operation is defined as:

$$\mathcal{H}_{\boldsymbol{\omega}}(\boldsymbol{X}_d)[k] = \boldsymbol{X}_d[k] \odot \boldsymbol{\omega}[k], \tag{3}$$

where $\odot$ represents dot product.

This operation can be interpreted as learning the frequency response function of a linear time-invariant (LTI) system [50]. By optimizing ω end-to-end, the model can flexibly approximate linear time-invariant transformations of temporal signals. This provides both theoretical generality and practical flexibility for modeling diverse temporal dynamics. Theoretical analysis can be referred to **Appendix A**. After modulation, the filtered dynamic component is fused with the time-invariant part $\boldsymbol{X}_s$ to recover the full frequency representation:

$$\dot{\boldsymbol{X}} = \mathcal{H}_{\boldsymbol{\omega}}(\boldsymbol{X}_d) + \boldsymbol{X}_s. \tag{4}$$

3.5 Prediction Layer

We leverage a prediction layer $f_{\boldsymbol{\theta}}$ to produce the final prediction given representation $\dot{\boldsymbol{X}}$. It can be customized to specific requirements. We adopt a two-layer MLP architecture in TimeEmb,

$$f_{\boldsymbol{\theta}}(\boldsymbol{X}) = \boldsymbol{W}_2(\text{ReLU}(\boldsymbol{W}_1\boldsymbol{X} + \boldsymbol{b}_1)) + \boldsymbol{b}_2, \tag{5}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{d \times L}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{H \times d}$ are projection matrices, H denotes the forecasting horizon, and $\boldsymbol{b}_1 \in \mathbb{R}^d$, $\boldsymbol{b}_2 \in \mathbb{R}^H$ are the corresponding biases.

To restore the time series to its original scale, we conduct inverse normalization with the instance-specific mean and variance. Consequently, the final prediction $\widehat{\boldsymbol{X}} \in \mathbb{R}^{H \times D}$ is computed as,

 https://doi.org/10.52202/085713-2560

$$\widehat{\boldsymbol{X}} = \text{InvNorm}(f_{\boldsymbol{\theta}}(\text{IFFT}(\dot{\boldsymbol{X}}))). \tag{6}$$

3.6 Optimization Objective

For model optimization, we employ the Mean Squared Error (MSE) to measure the loss between prediction and ground truth. Inspired by the self-correlation of the values in time series [47], we introduce Mean Absolute Error (MAE) loss in the frequency domain to alleviate the influence of self-correlation. In summary, our optimization objective function $\mathcal{L}$ can be expressed as follows,

$$\mathcal{L}(\widehat{\boldsymbol{X}}, \boldsymbol{Y}) = \alpha \text{MAE}(\text{FFT}(\widehat{\boldsymbol{X}}), \text{FFT}(\boldsymbol{Y})) + (1-\alpha)\text{MSE}(\widehat{\boldsymbol{X}}, \boldsymbol{Y}), \tag{7}$$

where $\alpha \in [0, 1]$ is hyper-parameter. The workflow of TimeEmb is detailed in **Appendix B**.

3.7 Computational Efficiency Analysis

We analyze the computational complexity of the core components of our TimeEmb, *i.e.,* time-invariant embedding bank and frequency filtering.

Time-invariant Embedding. The embedding bank $\boldsymbol{E} \in \mathbb{R}^{M \times F \times D}$ supports two lightweight operations: *embedding lookup* and *frequency-wise subtraction*. Given an input time series $\boldsymbol{X} \in \mathbb{R}^{L \times D}$, an embedding $\boldsymbol{X}_s \in \mathbb{R}^{F \times D}$ is retrieved based on its last timestamp with complexity $\mathcal{O}(M)$. The subtraction step $\boldsymbol{X}_d = \overline{\boldsymbol{X}} - \boldsymbol{X}_s$ involves $\mathcal{O}(F \times D)$ operations. Thus, the overall complexity is linear, *i.e.,* $\mathcal{O}(M + F \times D)$.

The embedding bank is also parameter-efficient: for example, in ETTh1 with $L = 96$, $M = 24$, $F = 49$, and $D = 7$, the total parameters required are only $24 \times 49 \times 7 = 8,232$.

Frequency Filtering. The spectral modulation of the dynamic component $\boldsymbol{X}_d$ is performed by element-wise multiplication with the learnable filter $\boldsymbol{\omega} \in \mathbb{C}^{F \times 1}$, yielding a complexity of $\mathcal{O}(F \times D)$.

Finally, the dominant cost in TimeEmb arises from the Fourier Transform, which operates at $\mathcal{O}(D \times L \log L)$. Overall, the computational complexity of key components is linear, making it highly efficient and scalable for long sequences and multivariate inputs.

4 Experiments

In this section, we conduct extensive experiments with real-world time series benchmarks to sufficiently assess the performance of our proposed model, including comparison with SOTA baselines (Section 4.2), compatibility evaluation (Section 4.3), time series disentanglement capability analysis (Section 4.4), and modules' effectiveness verification (Section 4.5).

4.1 Experimental Setup

4.1.1 Datasets and Baselines

Following the mainstream evaluation setup in existing time series prediction studies [51, 58], we conduct experiments on seven real-world benchmark datasets, including four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) [58], Weather [51], Electricity (ECL) [51], and Traffic [51]. Following prior works [51, 22], we split the ETTs dataset into training, validation, and test sets with a ratio of 6:2:2, while the other datasets were split in a ratio of 7:1:2.

To comprehensively evaluate the effectiveness, we select comprehensive SOTA baselines across three representative frameworks: (1) **Frequency-based models**: FilterNet [54], FITS [53], and FreTS [55]; (2) **MLP-based models**: DLinear [56] and CycleNet [14]; and (3) **Transformer-based models**: iTransformer [22], PatchTST [28], and Fredformer [34]. Detailed introduction of datasets and baselines can be found in **Appendix C**.

4.1.2 Implementation Details

To ensure fair comparison, we adapt common experimental settings: lookback window lengths $L \in \{96, 336, 720\}$ and prediction lengths $H \in \{96, 192, 336, 720\}$ for all baselines across datasets [51,

Table 1: Performance comparison with prediction lengths $H \in \{96, 192, 336, 720\}$ and lookback window length $L = 96$. The best results are highlighted in **bold** and the second best are underlined.

Model		**TimeEmb (ours)**		CycleNet 2024		Fredformer 2024		FilterNet 2024		iTransformer 2024		PatchTST 2023		FITS 2024		FreTS 2023		DLinear 2023	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.366**±0.001	**0.387**±0.001	0.378	0.391	0.373	0.392	0.375	0.394	0.386	0.405	0.394	0.406	0.386	0.396	0.395	0.407	0.386	0.400
	192	**0.417**±0.001	**0.416**±0.001	0.426	0.419	0.433	0.420	0.436	0.422	0.441	0.436	0.440	0.435	0.436	0.423	0.448	0.440	0.437	0.432
	336	**0.457**±0.001	**0.436**±0.001	0.464	0.439	0.470	0.437	0.476	0.443	0.487	0.458	0.491	0.462	0.478	0.444	0.499	0.472	0.481	0.459
	720	**0.459**±0.002	0.460±0.001	0.461	0.460	0.467	**0.456**	0.474	0.469	0.503	0.491	0.487	0.479	0.502	0.495	0.558	0.532	0.519	0.516
	avg	**0.425**±0.001	**0.425**±0.001	0.432	0.427	0.435	0.426	0.440	0.432	0.454	0.447	0.453	0.446	0.451	0.440	0.475	0.463	0.456	0.452
ETTh2	96	**0.277**±0.001	**0.328**±0.001	0.285	0.335	0.293	0.342	0.292	0.343	0.297	0.349	0.288	0.340	0.295	0.350	0.309	0.364	0.333	0.387
	192	**0.356**±0.001	**0.379**±0.001	0.373	0.391	0.371	0.389	0.369	0.395	0.380	0.400	0.376	0.395	0.381	0.396	0.395	0.425	0.477	0.476
	336	0.400±0.002	0.417±0.001	0.421	0.433	**0.382**	**0.409**	0.420	0.432	0.428	0.432	0.440	0.451	0.426	0.438	0.462	0.467	0.594	0.541
	720	**0.416**±0.001	0.437±0.002	0.453	0.458	**0.415**	**0.434**	0.430	0.446	0.427	0.445	0.436	0.453	0.431	0.446	0.721	0.604	0.831	0.657
	avg	**0.362**±0.001	**0.390**±0.001	0.383	0.404	0.365	0.393	0.378	0.404	0.383	0.407	0.385	0.410	0.383	0.408	0.472	0.465	0.559	0.515
ETTm1	96	**0.304**±0.001	**0.343**±0.001	0.319	0.360	0.326	0.361	0.318	0.358	0.334	0.368	0.329	0.365	0.355	0.375	0.335	0.372	0.345	0.372
	192	**0.354**±0.001	**0.373**±0.001	0.360	0.381	0.363	0.380	0.364	0.383	0.377	0.391	0.380	0.394	0.392	0.393	0.388	0.401	0.380	0.389
	336	**0.379**±0.001	**0.393**±0.001	0.389	0.403	0.395	0.403	0.396	0.406	0.426	0.420	0.400	0.410	0.424	0.414	0.421	0.426	0.413	0.413
	720	**0.435**±0.001	**0.428**±0.001	0.447	0.441	0.453	0.438	0.456	0.444	0.491	0.459	0.475	0.453	0.487	0.449	0.486	0.465	0.474	0.453
	avg	**0.368**±0.001	**0.384**±0.001	0.379	0.396	0.384	0.395	0.384	0.398	0.407	0.410	0.396	0.406	0.415	0.408	0.408	0.416	0.403	0.407
ETTm2	96	**0.163**±0.001	**0.242**±0.001	**0.163**	0.246	0.177	0.259	0.174	0.257	0.180	0.264	0.184	0.264	0.183	0.266	0.189	0.277	0.193	0.292
	192	**0.226**±0.001	**0.285**±0.001	0.229	0.290	0.243	0.301	0.240	0.300	0.250	0.309	0.246	0.306	0.247	0.305	0.258	0.326	0.284	0.362
	336	0.286±0.001	**0.324**±0.001	**0.284**	0.327	0.302	0.340	0.297	0.339	0.311	0.348	0.308	0.346	0.307	0.342	0.343	0.390	0.369	0.427
	720	**0.383**±0.001	**0.381**±0.001	0.389	0.391	0.397	0.396	0.392	0.393	0.412	0.407	0.409	0.402	0.407	0.399	0.495	0.480	0.554	0.522
	avg	**0.265**±0.001	**0.308**±0.001	0.266	0.314	0.279	0.324	0.276	0.322	0.288	0.332	0.287	0.330	0.286	0.328	0.321	0.368	0.350	0.401
Weather	96	**0.150**±0.001	**0.190**±0.001	0.158	0.203	0.163	0.207	0.162	0.207	0.174	0.214	0.176	0.217	0.166	0.213	0.174	0.208	0.196	0.255
	192	**0.200**±0.001	**0.238**±0.001	0.207	0.247	0.211	0.251	0.210	0.250	0.221	0.254	0.221	0.256	0.213	0.254	0.219	0.250	0.237	0.296
	336	**0.259**±0.001	**0.282**±0.001	0.262	0.289	0.267	0.292	0.265	0.290	0.278	0.296	0.275	0.296	0.269	0.294	0.273	0.290	0.283	0.335
	720	0.339±0.001	0.336±0.001	0.344	0.344	0.343	0.341	0.342	0.340	0.358	0.347	0.352	0.346	0.346	0.343	**0.334**	**0.332**	0.345	0.381
	avg	**0.237**±0.001	**0.262**±0.001	0.243	0.271	0.246	0.272	0.245	0.272	0.258	0.278	0.256	0.279	0.249	0.276	0.250	0.270	0.265	0.317
Electricity	96	**0.136**±0.001	0.231±0.001	**0.136**	**0.229**	0.147	0.241	0.147	0.245	0.148	0.240	0.164	0.251	0.200	0.278	0.176	0.258	0.197	0.282
	192	0.153±0.001	0.246±0.001	**0.152**	**0.244**	0.165	0.258	0.160	0.250	0.162	0.253	0.173	0.262	0.200	0.280	0.175	0.262	0.196	0.285
	336	**0.170**±0.001	**0.264**±0.001	**0.170**	**0.264**	0.177	0.273	0.173	0.267	0.178	0.269	0.190	0.279	0.214	0.295	0.185	0.278	0.209	0.301
	720	**0.208**±0.001	**0.297**±0.001	0.212	0.299	0.213	0.304	0.210	0.309	0.225	0.317	0.230	0.313	0.255	0.327	0.220	0.315	0.245	0.333
	avg	**0.167**±0.001	0.260±0.001	0.168	**0.259**	0.175	0.269	0.173	0.268	0.178	0.270	0.189	0.276	0.217	0.295	0.189	0.278	0.212	0.300
Traffic	96	0.432±0.002	0.279±0.001	0.458	0.296	0.406	0.277	0.430	0.294	**0.395**	**0.268**	0.427	0.272	0.651	0.391	0.593	0.378	0.650	0.396
	192	0.442±0.001	0.289±0.001	0.457	0.294	0.426	0.290	0.452	0.307	**0.417**	**0.276**	0.454	0.289	0.602	0.363	0.595	0.377	0.598	0.370
	336	0.456±0.002	0.295±0.002	0.470	0.299	**0.432**	**0.281**	0.470	0.316	0.433	0.283	0.450	0.282	0.609	0.366	0.609	0.385	0.605	0.373
	720	0.487±0.003	0.311±0.001	0.502	0.314	**0.463**	**0.300**	0.498	0.323	0.467	0.302	0.484	0.301	0.647	0.385	0.673	0.418	0.645	0.394
	avg	0.454±0.002	0.293±0.001	0.472	0.301	0.431	0.287	0.463	0.310	**0.428**	**0.282**	0.454	0.286	0.627	0.376	0.618	0.390	0.625	0.383

Table 2: Performance comparison of average prediction lengths with lookback lengths $L \in \{336, 720\}$. The best results are highlighted in **bold** and the second best are in underlined.

Lookback	$L = 336$								$L = 720$							
Model	**TimeEmb**		CycleNet		FilterNet		iTransformer		**TimeEmb**		CycleNet		SOFTS		DLinear	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.410**	**0.423**	0.415	0.426	0.423	0.437	0.440	0.447	**0.418**	**0.433**	0.430	0.439	0.434	0.455	0.437	0.448
ETTm1	**0.340**	**0.371**	0.355	0.379	0.352	0.381	0.365	0.392	**0.345**	**0.376**	0.355	0.381	0.364	0.396	0.367	0.391
ETTm2	**0.247**	**0.303**	0.251	0.309	0.265	0.325	0.286	0.337	**0.248**	**0.308**	0.249	0.312	0.268	0.331	0.261	0.327
Weather	**0.221**	0.255	0.226	0.266	0.224	**0.239**	0.236	0.272	**0.218**	**0.257**	0.224	0.266	0.230	0.272	0.240	0.292

58, 14]. Forecasting metrics include MSE and MAE, with results averaged over five independent runs. TimeEmb is trained for 30 epochs with early stopping (patience $= 5$ on the validation set). Batch sizes are 256 for ETTs and the Weather dataset, and 64 for others. Learning rates are selected from $\{0.0005, 0.001, 0.002, 0.005\}$, with TimeEmb's hidden layer size fixed at 512. Experiments use PyTorch 2.1 [33] on an NVIDIA RTX 4090 24GB GPU, with details in **Appendix C**.

4.2 Overall Performance

Table 1 presents the comparison results with $L = 96$ and $H \in \{96, 192, 336, 720\}$. The baseline results are from the original papers. Several conclusions can be made as follows:

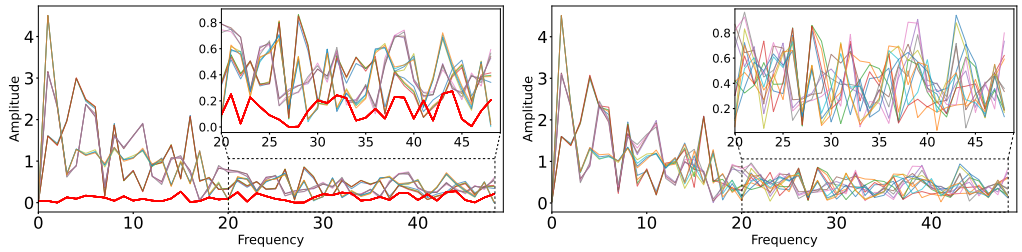
(1) ***TimeEmb consistently outperforms strong baselines across diverse datasets.*** Across multiple benchmarks and forecast horizons, TimeEmb achieves a significant reduction in MSE, with relative improvements ranging from 3.0% to 8.7% on average. This highlights the effectiveness of our frequency-based dynamic-static decomposition framework, which explicitly separates and models time-invariant and time-varying components.

(2) ***TimeEmb surpasses disentanglement-based baselines by offering more expressive and flexible decomposition.*** While CycleNet relies on a single long-period embedding and DLinear adopts a local moving average for trend extraction, both approaches struggle to capture long-term temporal patterns effectively. In contrast, TimeEmb leverages a global, timestamp-aware embedding bank to learn and represent recurring invariant patterns, enabling more accurate long-range forecasting.

https://doi.org/10.52202/085713-2560

Table 3: Performance of integrating TimeEmb with different backbones on Electricity and Weather. The best results are **bold**. Impr. indicates the performance improvement by equipping TimeEmb.

Dataset	Electricity								Weather							
Horizon	96		192		336		720		96		192		336		720	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Linear	0.196	0.279	0.195	0.282	0.208	0.298	0.243	0.330	0.197	0.256	0.238	0.295	0.285	0.335	**0.346**	0.381
+ our model	**0.173**	**0.270**	**0.179**	**0.274**	**0.193**	**0.288**	**0.233**	**0.320**	**0.170**	**0.218**	**0.222**	**0.260**	**0.275**	**0.298**	0.349	**0.345**
Impr.	+11.7%	+3.2%	+8.2%	+2.8%	+7.2%	+3.4%	+4.1%	+3.0%	+13.7%	+14.8%	+6.7%	+11.9%	+3.5%	+11.0%	-0.9%	+9.4%
MLP	0.177	0.265	0.183	0.271	0.197	0.287	0.234	0.320	0.180	0.234	0.223	0.274	0.268	0.309	**0.342**	0.370
+ our model	**0.137**	**0.234**	**0.155**	**0.250**	**0.172**	**0.267**	**0.211**	**0.303**	**0.154**	**0.197**	**0.203**	**0.243**	**0.263**	**0.288**	0.344	**0.344**
Impr.	+22.6%	+11.7%	+15.3%	+7.7%	+12.7%	+7.0%	+9.8%	+5.3%	+14.4%	+15.8%	+9.0%	+11.3%	+1.9%	+6.8%	-0.6%	+7.0%
DLinear	0.195	0.278	0.194	0.281	0.207	0.297	0.243	0.330	0.195	0.254	0.237	0.295	0.281	0.329	0.347	0.385
+ our model	**0.171**	**0.271**	**0.181**	**0.281**	**0.190**	**0.291**	**0.223**	**0.321**	**0.168**	**0.230**	**0.216**	**0.277**	**0.264**	**0.316**	**0.333**	**0.370**
Impr.	+12.3%	+2.5%	+6.7%	+0.0%	+8.2%	+2.0%	+8.2%	+2.7%	+13.8%	+9.4%	+8.9%	+6.1%	+6.0%	+4.0%	+4.0%	+3.9%
iTransformer	0.153	0.245	0.166	**0.256**	0.182	**0.274**	0.218	0.306	0.181	0.222	0.226	0.260	0.284	0.302	0.360	0.352
+ our model	**0.142**	**0.242**	**0.163**	0.260	**0.175**	0.275	**0.203**	**0.299**	**0.162**	**0.208**	**0.210**	**0.251**	**0.269**	**0.296**	**0.346**	**0.344**
Impr.	+7.2%	+1.2%	+1.8%	-1.6%	+3.8%	-0.4%	+6.9%	+2.3%	+10.5%	+6.3%	+7.1%	+3.5%	+5.3%	+2.0%	+3.9%	+2.3%



(a) Time series $\overline{\boldsymbol{X}}$ spectrum (colorful lines) and time-invariant embedding $\boldsymbol{X}_s$ spectrum (bold red line)

(b) Time-varying component $\boldsymbol{X}_d$ spectrum

Figure 3: Disentangled features visualization in frequency spectrum. Frequency components from 20 to 49 are zoomed in at the top right corner.

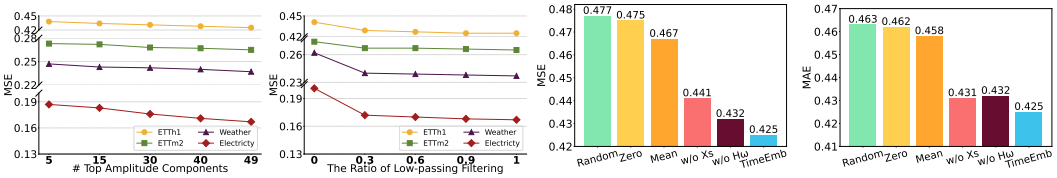
(3) ***TimeEmb outperforms frequency-domain models by jointly modeling invariant and dynamic components.*** FilterNet and FITS adopt a fixed filtering approach, which may not effectively handle non-stationary frequency components, as they ignore the global invariant patterns. Conversely, except for the frequency filter for the time-varying component, the embeddings in TimeEmb can preserve long-term invariant patterns, indicating the structural information of time series.

To evaluate model efficiency, we compare the number of trainable parameters, training time, and MSE on the Electricity dataset against mainstream baselines, as shown in Figure 1. Notably, TimeEmb uses over *5× fewer parameters* than the representative Transformer-based model iTransformer, while simultaneously achieving the *best predictive performance*. Benefiting from its lightweight design, TimeEmb significantly accelerates training without compromising accuracy, demonstrating an exceptional balance between efficiency and effectiveness.

To further assess the model's ability to capture long-term dependencies, we evaluate TimeEmb under extended lookback windows. Table 2 reports the average performance across all prediction lengths for $L = 336$ and $L = 720$. Full results are deferred to **Appendix D**. TimeEmb maintains state-of-the-art performance under long input horizons, showcasing its strong temporal modeling capacity.

4.3 Compatibility Analysis

To assess the generalizability of our proposed disentanglement mechanism for decoupling time-invariant and time-varying components, we integrate it into several state-of-the-art time series forecasting models, spanning both MLP-based and Transformer-based architectures. Specifically, our method only replaces the backbone prediction layer with the alternative model in the time domain, enabling seamless integration with different models. As shown in Table 3, incorporating our method consistently improves baseline performance across various prediction horizons, validating its effectiveness as a plug-and-play enhancement for diverse forecasting frameworks. Importantly, this integration incurs minimal computational overhead, enabling seamless adoption without significantly increasing model complexity or training cost. These results highlight the broad applicability of our disentanglement framework and its potential to strengthen existing models with negligible trade-offs.



(a) Amplitude analysis (b) Frequency analysis (c) MSE on ETTh1 (d) MAE on ETTh1

Figure 5: Key components contribution analysis.

4.4 Disentangled Features Visualization

To evaluate the disentanglement capability of TimeEmb in separating time-invariant and time-varying components, we conduct diverse visualization illustrations. We first select the first channel from the ETTm2 dataset and extract a dozen time series ending at 0 o'clock (*i.e.,* time index $t_{last} = 0$) from different days. In Figure 3(a), we illustrate the frequency-domain representations $\overline{\boldsymbol{X}}$ of these series as colorful lines, and the corresponding learned time-invariant embedding $\boldsymbol{X}_s$ as a bold red line. For clarity, we zoom in on frequency components in the range of 20 to 49. We can observe that the multiple $\overline{\boldsymbol{X}}$ from different days exhibit similar spectral structures, and the learned corresponding time-invariant embedding $\boldsymbol{X}_s$ captures the common pattern to a certain extent. Figure 3 (b) shows the time-varying component $\boldsymbol{X}_d$, which are relatively distinct from one another. The results clearly indicate that the original time series are hard to distinguish, but they become more separable after subtracting the time-invariant embedding. It shows that our TimeEmb successfully captures the shared time-invariant components across the input sequences, preserving the general structural information.

In addition, we present the distribution of the data before and after disentanglement from a high-level perspective.

We project the data samples from the Electricity test set onto a two-dimensional space using the T-SNE [45]. To capture week-level time-invariant patterns, we add an embedding bank composed of 7 learnable embeddings.

The time series $\overline{\boldsymbol{X}}$ and the time-varying component $\boldsymbol{X}_d$ are depicted in Figure 4 (a) and (b), respectively. Time series corresponding to different days of the week are color-coded. For example, the time series for Monday is in dark blue, and that for Sunday is in light yellow. As can be observed from Figure 4 (a), the time series $\overline{\boldsymbol{X}}$ from different days of the week tend to be intermixed, suggesting the existence of inherent similar patterns among them. After separating the time-invariant component from $\overline{\boldsymbol{X}}$, the time-varying components $\boldsymbol{X}_d$ for different days of the week carry specific information, as evidenced by their distinct and isolated distribution in Figure 4 (b). This visualization supports our previous finding again: by disentangling the time-invariant component $\boldsymbol{X}_s$, the time-varying component $\boldsymbol{X}_d$ becomes more distinguishable compared to the original time series $\overline{\boldsymbol{X}}$.

More visualization of TimeEmb can be found in **Appendix D**.

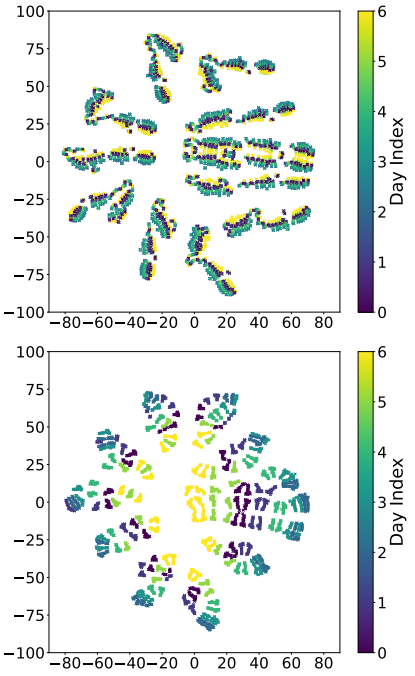


Figure 4: T-SNE visualization results.

4.5 Ablation Study

We conduct a comprehensive ablation study to evaluate the contributions of the key components in TimeEmb. The analysis is performed from two main perspectives: (1) the frequency composition of the time-invariant embedding, and (2) the impact of removing or altering individual modules.

https://doi.org/10.52202/085713-2560

4.5.1 Frequency Spectrum Analysis of $\boldsymbol{X}_s$

To examine how different frequency components contribute to the time-invariant embedding $\boldsymbol{X}_s$, we design two controlled perturbation strategies. **Amplitude-based masking**: For each input $\overline{\boldsymbol{X}}$, we preserve only the top-k frequency components of $\boldsymbol{X}_s$ with the highest amplitudes, and zero out the rest. **Frequency-based filtering**: We apply a low-pass filter by retaining only a certain proportion of low-frequency components, discarding the high-frequency parts. The results are shown in Figure 5 (a) and (b), with full details provided in **Appendix D**. As shown in Figure 5(a), model performance improves as more high-amplitude components are retained, indicating that both principal and subordinate frequencies carry useful invariant information [52]. Similarly, Figure 5(b) shows that increasing the proportion of low-frequency components leads to better performance, reflecting the importance of capturing both short-term and long-term periodicities in the invariant representation. These findings support the use of the full spectrum in constructing $\boldsymbol{X}_s$.

4.5.2 Component-wise Ablation

To assess the individual impact of key modules, we construct several variants of TimeEmb by altering or removing components: **Random**: The embedding bank is randomly initialized between training and testing. **Zero/Mean**: The embedding bank is fixed to zeros or the global mean value, respectively. **w/o** $\boldsymbol{X}_s$: The time-invariant component is entirely removed. **w/o** $\mathcal{H}_\omega$: The frequency filter is removed from the dynamic processing path. The results in Figure 5 (c) and (d) demonstrate that both the embedding bank and the frequency filter substantially contribute to model performance. In particular, removing either module leads to notable degradation, confirming the importance of jointly modeling the time-invariant and time-varying components. Complete ablation results are reported in **Appendix D**. Together, these findings validate the effectiveness of our systematic disentanglement framework, in which $\boldsymbol{X}_s$ and $\boldsymbol{X}_d$ are processed independently via dedicated structures to capture complementary temporal characteristics.

5 Conclusion

In this paper, we tackle the crucial issue of temporal non-stationarity in time series forecasting using a well-structured decomposition framework. We introduce TimeEmb, a lightweight yet effective architecture that combines global temporal embeddings and spectral filtering. TimeEmb enables separate processing of the disentangled time-variant and time-invariant components. Specifically, we utilize learnable embeddings to preserve the long-term invariant patterns within time series. Moreover, we devise a frequency filter to capture the temporal dependencies of the time-varying component. Extensive experiments confirm that our method not only attains state-of-the-art performance but also offers interpretable insights into temporal patterns via its dual-path design. It achieves an outstanding balance between performance and efficiency. Furthermore, it can be easily integrated with existing methods, thereby enhancing the ability to predict time series.

Acknowledgements

This work was supported by the China Postdoctoral Science Foundation under Grant No. 2025M771587; the Open Funding Programs of State Key Laboratory of AI Safety (2025-09); National Science and Technology Major Project under Grant No. 2021ZD0112500; the National Natural Science Foundation of China under Grant Nos. U22A2098, 62172185, 62206105, and 62202200; the Major Science and Technology Development Plan of Jilin Province under Grant No.20240212003GX, the Major Science and Technology Development Plan of Changchun under Grant No.2024WX05; the XJTU Research Fund for AI Science (No. 2025YXYC004).

References

[1] Richard Asselin. Frequency filter for time integrations. *Monthly Weather Review*, 100(6): 487–490, 1972.

[2] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[3] RN Bracewell. Signal analysis. *Proceedings of the IEEE*, 66(9):1101–1102, 1978.

[4] Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *TMLR*, 2023.

[5] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[6] Tao Dai, Beiliang Wu, Peiyuan Liu, Naiqi Li, Jigang Bao, Yong Jiang, and Shu-Tao Xia. Periodicity decoupling framework for long-term series forecasting. In *The International Conference on Learning Representations*, 2024.

[7] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. Deepmove: Predicting human mobility with attentional recurrent networks. In *WWW*, pages 1459–1468, 2018.

[8] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In *NIPS*, volume 32, pages 4650–4661. Curran Associates, Inc., 2019.

[9] Hannah Rosa Friesacher, Emma Svensson, Susanne Winiwarter, Lewis Mervin, Adam Arany, and Ola Engkvist. Temporal distribution shift in real-world pharmaceutical data: Implications for uncertainty quantification in qsar models. *arXiv preprint arXiv:2502.03982*, 2025.

[10] Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. Softs: Efficient multivariate time series forecasting with series-core fusion. In *NeurIPS*, 2024.

[11] Tao Hong, Pierre Pinson, Yi Wang, Rafał Weron, Dazhi Yang, and Hamidreza Zareipour. Energy forecasting: A review and outlook. *IEEE Open Access Journal of Power and Energy*, 7:376–388, 2020.

[12] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2019.

[13] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, pages 95–104, 2018.

[14] Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: Enhancing time series forecasting through modeling periodic patterns. In *NeurIPS*, 2024.

[15] Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *AAAI*, 2025.

[16] Chenxi Liu, Qianxiong Xu, Hao Miao, Sun Yang, Lingzheng Zhang, Cheng Long, Ziyue Li, and Rui Zhao. Timecma: Towards llm-empowered multivariate time series forecasting via cross-modality alignment. In *AAAI*, volume 39, pages 18780–18788, 2025.

[17] Chenxi Liu, Shaowen Zhou, Qianxiong Xu, Hao Miao, Cheng Long, Ziyue Li, and Rui Zhao. Towards cross-modality modeling for time series analytics: A survey in the llm era. In *IJCAI*, 2025.

[18] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: time series modeling and forecasting with sample convolution and interaction. In *NIPS*, pages 5816–5828, 2022.

[19] Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18915–18923, 2025.

[20] Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-tao Xia. Timebridge: Non-stationarity matters for long-term time series forecasting. 2025.

 https://doi.org/10.52202/085713-2560

[21] Yong Liu, Chenyu Li, Jianmin Wang, and Mingsheng Long. Koopa: Learning non-stationary time series dynamics with koopman predictors. *NeurIPS*, 36:12271–12290, 2023.

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

[23] R. T. M. C. Lu. *Algorithms for Discrete Fourier Transform and Convolution*. Springer, Berlin, Heidelberg, 1989. ISBN 978-1-4419-5126-4.

[24] Yucong Luo, Yitong Zhou, Mingyue Cheng, Jiahao Wang, Daoyu Wang, Tingyue Pan, and Jintao Zhang. Time series forecasting as reasoning: A slow-thinking approach with reinforced llms. *arXiv preprint arXiv:2506.10630*, 2025.

[25] Hao Miao, Ziqiao Liu, Yan Zhao, Chenjuan Guo, Bin Yang, Kai Zheng, and Christian S Jensen. Less is more: Efficient time series dataset condensation via two-fold modal matching. *PVLDB*, 18(2):226–238, 2024.

[26] Hao Miao, Yan Zhao, Chenjuan Guo, Bin Yang, Kai Zheng, Feiteng Huang, Jiandong Xie, and Christian S. Jensen. A unified replay-based continuous learning framework for spatio-temporal prediction on streaming data. In *ICDE*, pages 1050–1062, 2024.

[27] RE Morrow. The fast fourier transform. *IEEE spectrum*, 4(12):63–70, 1967.

[28] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *ICLR*, 2023.

[29] Kin G. Olivares, Cristian Challú, Azul Garza, Max Mergenthaler Canseco, and Artur Dubrawski. NeuralForecast: User friendly state-of-the-art neural forecasting models. PyCon Salt Lake City, Utah, US 2022, 2022. URL https://github.com/Nixtla/neuralforecast.

[30] Alan V Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

[31] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *ICLR*, 2020.

[32] Pranay Pasula. Real world time series benchmark datasets with distribution shifts: Global crude oil price and volatility. *arXiv preprint arXiv:2308.10846*, 2023.

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[34] Xihao Piao, Zheng Chen, Taichi Murayama, Yasuko Matsubara, and Yasushi Sakurai. Fredformer: Frequency debiased transformer for time series forecasting. In *SIGKDD*, pages 2400–2410, 2024.

[35] Syama Sundar Rangapuram, Matthias Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *NIPS*, pages 7796–7805, 2018.

[36] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. 31, 2018.

[37] CLEVELAND RB. Stl: A seasonal-trend decomposition procedure based on loess. *J Off Stat*, 6:3–73, 1990.

[38] Weijieying Ren, Tianxiang Zhao, Wei Qin, and Kunpeng Liu. T-sas: Toward shift-aware dynamic adaptation for streaming data. In *CIKM*, pages 4244–4248, 2023.

[39] Zezhi Shao, Zhao Zhang, Fei Wang, Wei Wei, and Yongjun Xu. Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting. In *CIKM*, page 4454–4458, 2022.

[40] Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S Jensen. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *PVLDB*, 15 (11):2733–2746, 2022.

[41] Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S Jensen. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *Proceedings of the VLDB Endowment*, 15(11):2733–2746, 2022.

[42] Pushpendra Singh, Shiv Dutt Joshi, Rakesh Kumar Patney, and Kaushik Saha. The fourier decomposition method for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2199):20160871, 2017.

[43] Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. Scale efficiently: Insights from pretraining and finetuning transformers. In *ICLR*, 2020.

[44] Stephen J Taylor. *Modelling financial time series*. world scientific, 2008.

[45] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 9(86): 2579–2605, 2008.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017.

[47] Hao Wang, Lichen Pan, Yuan Shen, Zhichao Chen, Degui Yang, Yifei Yang, Sen Zhang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. Label correlation biases direct time series forecast. In *ICLR*, 2025.

[48] Jiahao Wang, Mingyue Cheng, and Qi Liu. Can slow-thinking llms reason over time? empirical studies in time series forecasting. *arXiv preprint arXiv:2505.24511*, 2025.

[49] 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 *ICLR*, 2024.

[50] Jan C Willems. From time series to linear system—part i. finite dimensional linear time invariant systems. *Automatica*, 22(5):561–580, 1986.

[51] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *NeurIPS*, volume 34, pages 22419–22430, 2021.

[52] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *ICLR*, 2023.

[53] Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with 10k parameters. In *ICLR*, 2024.

[54] K. Yi, J. Fei, Q. Zhang, H. He, S. Hao, D. Lian, and W. Fan. Filternet: Harnessing frequency filters for time series forecasting. In *NeurIPS*, 2024.

[55] Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are more effective learners in time series forecasting. In *NeurIPS*, 2023.

[56] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, volume 37, pages 11121–11128, 2023.

[57] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 1–4, 2016.

 https://doi.org/10.52202/085713-2560

[58] 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 *AAAI*, volume 35, pages 11106–11115, 2021.

[59] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, pages 27268–27286, 2022.

A In-Depth Analysis of TimeEmb

A.1 Innovation Discussions

TimeEmb vs. Disentanglement Methods. While prior disentanglement approaches [31, 51] focus primarily on separating trend and residual components based on local statistics within individual time series, TimeEmb introduces two fundamental advancements. First, instead of local decomposition, our model leverages a learnable embedding bank to capture globally consistent and recurrent patterns across the entire dataset, effectively preserving system-level invariants. Second, TimeEmb is the first to incorporate learnable frequency-domain filtering into the disentanglement framework, enabling efficient and expressive modeling of dynamic components in the spectral space.

TimeEmb vs. Embedding-enhanced Methods. Embedding-enhanced models [40, 39] typically use identifier-based embeddings (*e.g.,* time slot, spatial ID) to encode auxiliary information. In contrast, TimeEmb adopts a decomposition-based design where a learnable temporal embedding bank explicitly models the time-invariant signal component. This enables data-driven recovery of latent periodic patterns without relying on predefined identifiers or external priors.

A.2 Theoritical Support

In this section, we theoretically analyze the core design of TimeEmb from a frequency-domain perspective. We focus on two main aspects: the completeness of frequency-domain representation and operations, and the expressiveness of the learnable spectral filtering mechanism for modeling dynamic temporal signals.

A.2.1 Completeness of Frequency-Domain Representation and Operations

TimeEmb operates entirely in the frequency domain by applying the real-valued Fast Fourier Transform (rFFT) to input sequences. For a real-valued time series $\boldsymbol{X} \in \mathbb{R}^{L \times D}$, its spectral representation is obtained as $\overline{\boldsymbol{X}} \in \mathbb{C}^{F \times D}$, where $F = \lfloor L/2 \rfloor + 1$ due to the conjugate symmetry of the spectrum. The rFFT is defined as:

$$\overline{\boldsymbol{X}}[k] = \sum_{n=0}^{L-1} \boldsymbol{X}[n] \cdot e^{-2\pi j k n / L}, \quad k = 0, \ldots, F-1. \tag{8}$$

This transformation is invertible via the corresponding inverse real FFT (irFFT), guaranteeing that no information is lost in the process. Thus, rFFT offers a complete and efficient frequency representation of real-valued signals [30, 5]. Beyond transformation, TimeEmb performs a sequence of operations entirely in the frequency domain:

1. Subtraction of a time-invariant embedding $\boldsymbol{X}_s$ from the input spectrum $\overline{\boldsymbol{X}}$;
2. Frequency-wise modulation of the residual $\boldsymbol{X}_d = \overline{\boldsymbol{X}} - \boldsymbol{X}_s$ via a learnable filter $\boldsymbol{\omega}$;
3. Reconstruction of the final spectrum $\dot{\boldsymbol{X}} = \boldsymbol{X}_s + \boldsymbol{X}_d \odot \boldsymbol{\omega}$, followed by an irFFT to recover the output in the time domain.

Each of these operations, *i.e.,* subtraction, modulation, and addition, is algebraically well-defined and closed in the frequency domain. Because the rFFT is invertible, the entire transformation chain in TimeEmb is representation-complete: all original information is preserved, while allowing structured manipulation in the spectral space.

This design offers several important advantages. First, it enables precise modeling of periodicity and oscillatory behavior, which are often hard to localize in the time domain. Second, working entirely in the frequency space allows for efficient and interpretable decomposition of long-range temporal patterns. Lastly, the model avoids any information loss due to projection or truncation, ensuring theoretical soundness in its design.

A.2.2 Expressiveness of Frequency-Domain Filtering

To model the dynamic (time-varying) component of the input sequence, TimeEmb applies a frequency-domain filter over the residual spectrum. Formally, given the residual $\boldsymbol{X}_d \in \mathbb{C}^{F \times D}$, a learnable

 https://doi.org/10.52202/085713-2560

modulation vector $\boldsymbol{\omega} \in \mathbb{C}^{F\times 1}$ is applied as:

$$\mathcal{H}_{\omega}(\boldsymbol{X}_d)[k] = \boldsymbol{X}_d[k] \odot \boldsymbol{\omega}[k]. \tag{9}$$

This operation is grounded in the **Convolution Theorem**: pointwise multiplication in the frequency domain corresponds to convolution in the time domain [30]. Therefore, the frequency filter can be interpreted as learning the impulse response of a Linear Time-Invariant (LTI) system directly in the spectral domain.

This interpretation grants the model several expressive and practical advantages [3]. First, it enables the learning of flexible signal transformations that go beyond local convolutions, *e.g.,* capturing long-range dependencies with global frequency-aware operations. Second, the filtering process is computationally efficient, operating in $\mathcal{O}(F \times D)$, and avoids the kernel length constraints inherent in time-domain CNNs. Finally, this formulation provides intuitive control over the model's sensitivity to various periodic structures, allowing it to emphasize or suppress spectral bands depending on task-specific dynamics.

In essence, the frequency filter in TimeEmb serves as a powerful and compact operator that simulates a broad family of spectral responses.

A.3 Convolution Theorem

Frequency filtering modifies a signal's frequency content. Given a signal $x[n]$ and a filter with frequency response $H[k]$, the filtered signal $Y[k] = X[k]H[k]$ in the frequency domain. By the convolution theorem, the filtered signal $y[n] = \text{IDFT}(Y[k]) = (x \circledast h)[n]$ in the time domain. The proof is as follows:

Let $x[n]$ and $h[n]$ be length - N sequences with DFTs $X[k]$ and $H[k]$:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}, \quad k = 0, 1, \cdots, N-1, \tag{10}$$

$$H[k] = \sum_{n=0}^{N-1} h[n] e^{-j\frac{2\pi}{N}kn}, \quad k = 0, 1, \cdots, N-1. \tag{11}$$

The circular convolution of $x[n]$ and $h[n]$ is defined as $y[n] = (x \circledast h)[n] = \sum_{m=0}^{N-1} x[m]h[(n-m) \bmod N]$, where $\circledast$ represents the circular convolution operation, and $(n-m) \bmod N$ denotes the modulo N operation of $n-m$.

The DFT of $y[n]$, denoted as $Y[k]$:

$$Y[k] = \sum_{n=0}^{N-1} y[n] e^{-j\frac{2\pi}{N}kn} \tag{12}$$

$$= \sum_{n=0}^{N-1} \left(\sum_{m=0}^{N-1} x[m]h[(n-m) \bmod N] \right) e^{-j\frac{2\pi}{N}kn}. \tag{13}$$

Let $l = (n-m) \bmod N$, and the above equation can be rewritten as:

$$Y[k] = N \sum_{m=0}^{N-1} x[m] \sum_{l=0}^{N-1} h[l] e^{-j\frac{2\pi}{N}k(l+m)}. \tag{14}$$

According to the exponential operation rule, we have:

$$Y[k] = \left(\sum_{m=0}^{N-1} x[m] e^{-j\frac{2\pi}{N}km} \right) \left(\sum_{l=0}^{N-1} h[l] e^{-j\frac{2\pi}{N}kl} \right). \tag{15}$$

Therefore:

$$Y[k] = X[k]H[k]. \tag{16}$$

Algorithm 1 Workflow of TimeEmb.

Input: Time series $\boldsymbol{X} \in \mathbb{R}^{L \times D}$.
Output: Prediction $\widehat{\boldsymbol{X}} \in \mathbb{R}^{H \times D}$.

1: // Domain transformation
2: $\overline{\boldsymbol{X}} = \text{FFT}(\text{InstNorm}(\boldsymbol{X}))$
3: // Time series disentanglement
4: $\boldsymbol{X}_d = \overline{\boldsymbol{X}} - \boldsymbol{X}_s$ {Eq. (2)}
5: // Frequency filtering
6: $\dot{\boldsymbol{X}} = \mathcal{H}_{\boldsymbol{\omega}}(\boldsymbol{X}_d) + \boldsymbol{X}_s$ {Eq. (4)}
7: // Final prediction
8: $\widehat{\boldsymbol{X}} = \text{InvNorm}(f_{\boldsymbol{\theta}}(\text{IFFT}(\dot{\boldsymbol{X}})))$ {Eq. (6)}
9: **Return:** $\widehat{\boldsymbol{X}}$

Table 4: Dataset statistics. "Channels" denotes the number of variables in each dataset; "M of each bank" denotes the capacity of each embedding bank utilized in TimeEmb. "d" refers to the day-level embedding bank, while "w" indicates the week-level embedding bank.

Datasets	ETTh1	ETTh2	ETTm1	ETTm2	Electricity	Weather	Traffic
Channels	7	7	7	7	321	21	862
Timesteps	17420	17420	69680	69680	26304	52696	17544
Frequency	Hourly	Hourly	15min	15min	Hourly	10min	Hourly
Domain	Electricity	Electricity	Electricity	Electricity	Electricity	Weather	Traffic
M of each bank	24 (d)	24 (d)	24 (d)	24 (d)	24 (d) + 7 (w)	24 (d)	24 (d) + 7 (w)

The same is true for inverse derivation. We can ultimately infer that:

$$y[n] = \text{IDFT}(\mathcal{Y}[k]) = (x \circledast h)[n] = \sum_{m=0}^{N-1} x[m]h[(n-m) \mod N]. \tag{17}$$

In conclusion, we have proved that the DFT of the circular convolution is equal to the product of the DFTs, and the IDFT of the product of the DFTs is equal to the circular convolution, which means that frequency filtering (multiplication in the DFT domain) is equivalent to circular convolution in the time domain.

B Algorithm

We present the pipeline of TimeEmb in Algorithm 1. We begin by performing a domain transformation for frequency analysis (line 2). Specifically, we apply instance normalization to the input sequence $\boldsymbol{X}$, followed by a Fast Fourier Transformation (FFT) to obtain the frequency series $\overline{\boldsymbol{X}}$. Next, to disentangle the time series, we retrieve the corresponding embedding from the embedding bank, which serves as the time-invariant component $\boldsymbol{X}_s$. We then separate it from $\overline{\boldsymbol{X}}$ to extract the time-varying component $\boldsymbol{X}_d$ (line 4). Subsequently, we implement frequency filtering with a spectral modulation operator $\boldsymbol{\omega}$ to effectively model the dynamic component. After this step, we add back the time-invariant series $\boldsymbol{X}_s$ (line 6). The combined series then undergoes the Inverse Fast Fourier Transform (IFFT), followed by a projection layer, and concludes with inverse normalization to generate the final prediction (line 8).

C Experimental Setup

C.1 Datasets

We detail the description of the datasets here:

ETT (Electricity Transformer Temperature) contains two subsets of data: ETTh and ETTm. These datasets are based on hourly and 15-minute intervals, collected from electricity transformers between July 2016 and July 2018.

 https://doi.org/10.52202/085713-2560

Weather records 21 weather features, including air temperature and humidity, every ten minutes throughout 2020.

Electricity collects 321 clients' electricity consumption hourly from 2012 to 2014.

Traffic comprises the hourly data of 862 sensors of San Francisco freeways from 2015 to 2016.

Detailed statistics are displayed in Table 4.

C.2 Baselines

We compare TimeEmb with 9 representative and state-of-the-art models to evaluate the performance and effectiveness, including Frequency-based models, MLP-based models, and Transformer-based models. The details of these baselines are as follows:

FilterNet proposes two kinds of learnable filters-Plain shaping filter and Contextual shaping filter-to approximately surrogate the linear and attention mappings widely adopted in time series literature. The detailed implementation is available at https://github.com/aikunyi/FilterNet.

FITS conducts time series analysis using interpolation in the complex frequency domain, achieving low cost with 10K parameters. The detailed implementation is available at https://github.com/VEWOXIC/FITS.

FreTS presents a new approach to utilizing MLPs in the frequency domain, effectively capturing the underlying patterns of time series while benefiting from a global view and energy compaction. The detailed implementation is available at https://github.com/aikunyi/FreTS.

DLinear employs a straightforward one-layer linear model to capture temporal relationships through season-trend decomposition. The detailed implementation is available at https://github.com/cure-lab/LTSF-Linear.

SOFTS introduces an efficient MLP-based model that utilizes a centralized strategy to enhance performance and lessen dependence on the quality of each channel. The detailed implementation is available at https://github.com/Secilia-Cxy/SOFTS.

CycleNet utilizes an RCF technique to separate the inherent periodic patterns within sequences and then performs predictions on the residual components of the modeled cycles. The detailed implementation is available at https://github.com/ACAT-SCUT/CycleNet.

iTransformer uses attention and feed-forward networks on inverted dimensions. It embeds time points of individual series into variate tokens for the attention mechanism to capture multivariate correlations. Additionally, the feed-forward network is applied to each variate token to learn non-linear representations. The detailed implementation is available at https://github.com/thuml/iTransformer.

PatchTST breaks down time series data into subseries-level patches, which helps in extracting local semantic information. The detailed implementation is available at https://github.com/yuqinie98/PatchTST.

Fredformer is a Transformer-based framework that addresses frequency bias by equally learning features across various frequency bands, which ensures the model does not neglect lower amplitude features that are crucial for accurate forecasting. The detailed implementation is available at https://github.com/chenzRG/Fredformer.

C.3 Implementation Details

We implemented TimeEmb with PyTorch and conducted experiments on a single NVIDIA RTX4090 GPU that has 24GB of memory. TimeEmb was trained for 30 epochs, with early stopping implemented and a patience level of 5 based on the validation set. The batch size was set to 256 for both the ETT and Weather datasets, while a batch size of 64 was used for the remaining datasets. This adjustment was necessary because the latter datasets have a larger number of channels, which requires a smaller batch size to prevent out-of-memory issues. The learning rate was chosen from the range 0.0005, 0.001, 0.002, 0.005, based on the performance on the validation set. The size of the hidden layer in TimeEmb was consistently set to 512.

Table 5: Full results with lookback lengths $L = 336$. The best results are in **bold** and the second best are underlined.

Model		TimeEmb		CycleNet		FilterNet		SOFTS		iTransformer		DLinear	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.367**	**0.394**	0.374	0.396	0.379	0.404	0.386	0.405	0.396	0.415	0.374	0.398
	192	**0.403**	**0.414**	0.406	0.415	0.417	0.428	0.428	0.432	0.434	0.438	0.430	0.440
	336	**0.422**	**0.425**	0.431	0.430	0.437	0.443	0.449	0.448	0.452	0.451	0.442	0.445
	720	**0.446**	**0.459**	0.450	0.464	0.458	0.472	0.460	0.476	0.476	0.485	0.497	0.507
	avg	**0.410**	**0.423**	0.415	0.426	0.423	0.437	0.431	0.440	0.440	0.447	0.436	0.448
ETTh2	96	**0.276**	**0.333**	0.279	0.341	0.302	0.356	0.298	0.356	0.334	0.379	0.281	0.347
	192	**0.335**	**0.378**	0.342	0.385	0.350	0.393	0.360	0.394	0.413	0.424	0.367	0.404
	336	**0.370**	**0.405**	0.371	0.413	0.376	0.414	0.385	0.415	0.414	0.432	0.438	0.454
	720	**0.396**	**0.433**	0.426	0.451	0.414	0.444	0.449	0.463	0.433	0.454	0.598	0.549
	avg	**0.344**	**0.387**	0.355	0.398	0.361	0.402	0.373	0.407	0.399	0.422	0.421	0.439
ETTm1	96	**0.282**	**0.332**	0.299	0.348	0.289	0.344	0.296	0.350	0.303	0.357	0.307	0.350
	192	**0.323**	**0.361**	0.334	0.367	0.331	0.369	0.336	0.374	0.345	0.383	0.340	0.373
	336	**0.353**	**0.380**	0.368	0.386	0.364	0.389	0.371	0.396	0.375	0.397	0.377	0.397
	720	**0.403**	**0.410**	0.417	0.414	0.425	0.423	0.433	0.432	0.435	0.432	0.433	0.433
	avg	**0.340**	**0.371**	0.355	0.379	0.352	0.381	0.359	0.388	0.365	0.392	0.364	0.388
ETTm2	96	0.160	**0.243**	**0.159**	0.247	0.177	0.265	0.174	0.259	0.184	0.273	0.165	0.257
	192	0.218	**0.283**	**0.214**	0.286	0.232	0.304	0.240	0.307	0.262	0.322	0.227	0.307
	336	**0.265**	**0.316**	0.269	0.322	0.284	0.339	0.295	0.342	0.307	0.351	0.304	0.362
	720	**0.346**	**0.370**	0.363	0.382	0.367	0.390	0.377	0.396	0.390	0.402	0.431	0.441
	avg	**0.247**	**0.303**	0.251	0.309	0.265	0.325	0.272	0.326	0.286	0.337	0.282	0.342
Weather	96	**0.144**	0.189	0.148	0.200	0.150	**0.183**	0.160	0.209	0.163	0.213	0.174	0.235
	192	**0.187**	0.233	0.190	0.240	0.193	**0.221**	0.204	0.250	0.203	0.250	0.219	0.281
	336	**0.238**	0.271	0.243	0.283	0.246	**0.258**	0.249	0.284	0.253	0.288	0.264	0.317
	720	0.315	0.326	0.322	0.339	**0.308**	**0.295**	0.324	0.335	0.326	0.338	0.324	0.363
	avg	**0.221**	0.255	0.226	0.266	0.224	**0.239**	0.234	0.270	0.236	0.272	0.245	0.299
Electricity	96	0.128	0.223	0.128	0.223	0.132	0.224	**0.127**	**0.221**	0.133	0.229	0.140	0.237
	192	0.146	0.240	0.144	**0.237**	**0.143**	**0.237**	0.148	0.242	0.156	0.251	0.153	0.250
	336	0.161	0.256	0.160	0.254	**0.155**	**0.253**	0.166	0.261	0.172	0.267	0.169	0.267
	720	0.198	0.289	0.198	**0.287**	**0.195**	0.292	0.202	0.293	0.209	0.304	0.203	0.299
	avg	0.158	0.252	0.158	**0.250**	**0.156**	0.252	0.161	0.254	0.168	0.263	0.166	0.263
Traffic	96	0.381	0.263	0.386	0.268	0.398	0.289	**0.346**	**0.246**	0.361	0.255	0.410	0.282
	192	0.398	0.271	0.404	0.276	0.422	0.303	**0.373**	**0.258**	0.380	0.268	0.423	0.288
	336	0.411	0.278	0.416	0.281	0.437	0.312	**0.385**	**0.265**	0.389	0.273	0.436	0.296
	720	0.439	0.294	0.445	0.300	0.464	0.325	0.419	**0.283**	**0.415**	0.285	0.466	0.315
	avg	0.407	0.277	0.413	0.281	0.430	0.307	**0.381**	**0.263**	0.386	0.270	0.434	0.295

D Detailed Results

D.1 Full Results with Lookback Window Length L $\in \{336, 720\}$

To evaluate the performance of TimeEmb in modeling long-term temporal dependencies, we further conduct experiments with extended lookback window lengths of 336 and 720. As shown in Table 5 and Table 6, TimeEmb consistently achieves competitive or superior performance across various forecast horizons under these challenging settings. Unlike many baseline models whose performance degrades significantly as the input length increases, TimeEmb maintains stable accuracy, demonstrating strong temporal generalization.

This performance stems from the architectural design of TimeEmb. The time-invariant embedding bank allows the model to effectively summarize recurring structural patterns, regardless of input length. Meanwhile, the frequency-domain filter adaptively emphasizes relevant dynamic components without being constrained by local receptive fields. Together, these modules enable TimeEmb to capture both long-range dependencies and localized variations efficiently.

Overall, these results indicate that TimeEmb is not only effective under standard settings, but also exhibits strong scalability and resilience when applied to long-context forecasting tasks—a desirable property for real-world time series applications.

https://doi.org/10.52202/085713-2560

Table 6: Full results with lookback lengths $L = 720$. The best results are in **bold** and the second best are underlined.

Model		TimeEmb		CycleNet		FilterNet		SOFTS		iTransformer		DLinear	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.372**	**0.400**	0.379	0.403	0.390	0.418	0.384	0.416	0.401	0.430	0.379	0.402
	192	**0.413**	0.427	0.416	**0.425**	0.424	0.439	0.423	0.442	0.434	0.452	0.419	0.429
	336	**0.438**	**0.443**	0.447	0.445	0.450	0.461	0.446	0.461	0.468	0.475	0.456	0.456
	720	**0.449**	**0.462**	0.477	0.483	0.484	0.488	0.481	0.500	0.525	0.520	0.493	0.506
	avg	**0.418**	**0.433**	0.430	0.439	0.437	0.452	0.434	0.455	0.457	0.469	0.437	0.448
ETTh2	96	0.290	0.348	**0.271**	**0.337**	0.297	0.357	0.295	0.357	0.306	0.369	0.309	0.373
	192	0.347	0.387	**0.332**	**0.380**	0.361	0.400	0.365	0.401	0.372	0.409	0.409	0.433
	336	0.376	0.411	**0.362**	**0.408**	0.397	0.431	0.398	0.426	0.403	0.434	0.508	0.495
	720	**0.399**	**0.439**	0.415	0.449	0.435	0.460	0.448	0.473	0.434	0.464	0.851	0.653
	avg	0.353	0.396	**0.345**	**0.394**	0.373	0.412	0.377	0.414	0.379	0.419	0.519	0.489
ETTm1	96	**0.293**	**0.346**	0.307	0.353	0.301	0.358	0.299	0.357	0.317	0.367	0.309	0.353
	192	**0.326**	**0.366**	0.337	0.371	0.340	0.379	0.342	0.381	0.347	0.385	0.345	0.376
	336	**0.356**	**0.382**	0.364	0.387	0.375	0.398	0.375	0.401	0.377	0.402	0.376	0.398
	720	**0.405**	**0.410**	0.410	0.411	0.434	0.426	0.441	0.443	0.429	0.431	0.436	0.436
	avg	**0.345**	**0.376**	0.355	0.381	0.363	0.390	0.364	0.396	0.368	0.396	0.367	0.391
ETTm2	96	0.163	0.251	**0.159**	**0.249**	0.180	0.271	0.181	0.272	0.187	0.278	0.163	0.256
	192	0.219	0.290	**0.214**	**0.289**	0.239	0.313	0.234	0.310	0.251	0.319	0.220	0.300
	336	**0.265**	**0.320**	0.268	0.326	0.283	0.341	0.284	0.342	0.307	0.355	0.283	0.347
	720	**0.343**	**0.372**	0.353	0.384	0.361	0.394	0.373	0.398	0.391	0.411	0.376	0.406
	avg	**0.248**	**0.308**	0.249	0.312	0.266	0.330	0.268	0.331	0.284	0.341	0.261	0.327
Weather	96	**0.143**	**0.193**	0.149	0.203	0.153	0.208	0.152	0.205	0.168	0.222	0.169	0.227
	192	**0.188**	**0.237**	0.192	0.244	0.199	0.250	0.199	0.251	0.209	0.256	0.213	0.271
	336	**0.236**	**0.274**	0.242	0.283	0.248	0.287	0.248	0.288	0.267	0.302	0.259	0.311
	720	**0.306**	**0.325**	0.312	0.333	0.313	0.333	0.322	0.343	0.337	0.352	0.319	0.359
	avg	**0.218**	**0.257**	0.224	0.266	0.228	0.270	0.230	0.272	0.245	0.283	0.240	0.292
Electricity	96	0.129	0.225	**0.128**	**0.223**	0.137	0.235	0.137	0.232	0.142	0.243	0.134	0.232
	192	0.145	0.241	**0.143**	**0.237**	0.160	0.259	0.157	0.252	0.160	0.261	0.148	0.245
	336	0.161	0.257	**0.159**	**0.254**	0.174	0.274	0.172	0.268	0.179	0.281	0.163	0.263
	720	**0.197**	0.289	**0.197**	**0.287**	0.212	0.307	0.198	0.291	0.220	0.316	0.198	0.296
	avg	0.158	0.253	**0.157**	**0.250**	0.171	0.269	0.166	0.261	0.175	0.275	0.161	0.259
Traffic	96	0.374	0.268	0.374	0.268	0.386	0.285	**0.355**	**0.253**	0.358	0.254	0.388	0.275
	192	0.387	0.268	0.390	0.275	0.401	0.281	**0.369**	**0.261**	0.375	0.263	0.399	0.279
	336	0.401	0.275	0.405	0.282	0.408	0.288	**0.387**	**0.271**	**0.387**	0.273	0.414	0.291
	720	0.434	0.292	0.441	0.302	0.447	0.306	**0.409**	**0.286**	0.418	0.292	0.449	0.308
	avg	0.399	0.274	0.403	0.282	0.411	0.290	**0.380**	**0.268**	0.385	0.271	0.413	0.288

D.2 Full Results of Efficiency Analysis

To verify the lightweight characteristics of TimeEmb, we conduct several efficiency experiments on it. We first compare the maximum memory(MB), training time, and MSE on the ETTm1 dataset against mainstream baselines. Subsequently, we calculate the extra parameters from TimeEmb based on the compatibility study, which can also prove the extensibility and efficiency of TimeEmb. The detailed results are displayed in Table 7 and Table 8. It indicates that TimeEmb can operate in a resource-constrained environment and achieve excellent performance. Moreover, TimeEmb can play the role of a plug-in module in other models, improving their performance at a low cost.

D.3 Full Results of Ablation Study

D.3.1 Ablation Results of Embedding Frequency Spectrum Analysis

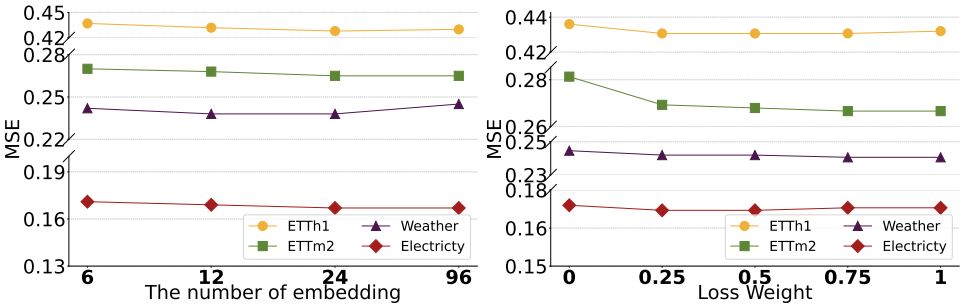
The complete results of our embedding frequency spectrum ablation are displayed in Table 9 and Table 10. The term "k" in Table 9 represents the number of frequency components of $\boldsymbol{X}_s$ selected based on their top amplitudes, while the term "γ" in Table 10 refers to the ratio of low-passing filtering applied to the embeddings. The results reveal that leveraging the entire frequency spectrum leads to the best performance. Note that the more frequency components are covered, the better performance TimeEmb can achieve, which indicates that each frequency in the band is important.

Table 7: Efficiency comparison on ETTm1 dataset shows TimeEmb leads in performance and efficiency.

Model	Training Time(s/epoch)	MSE	Max Memory(MB)
CycleNet	1.77	<u>0.447</u>	91.37
Fredformer	4.46	0.453	1512.32
FilterNet	<u>1.63</u>	0.456	**79.51**
iTransformer	2.44	0.482	275.12
SOFTS	2.00	0.466	183.95
TimeEmb	**1.61**	**0.435**	<u>82.36</u>

Table 8: We equip Fredformer and CycleNet with TimeEmb on ETTh2. It brings stable performance(MSE) gains with trivial extra training costs.

Horizon	96	192	336	720
Fredformer	0.293	0.371	0.382	0.415
+TimeEmb	**0.289**	**0.358**	**0.360**	**0.387**
Impr.	1.4%	3.5%	5.8%	6.7%
former param	32820131	33465731	9894911	13656959
current param	32830860	33476460	9905640	13667688
extra param	**10729 (0.03%-0.11%)**			
CycleNet	0.285	0.373	0.421	0.453
+TimeEmb	**0.277**	**0.351**	**0.399**	**0.415**
Impr.	2.8%	5.9%	5.2%	8.4%
former param	99080	148328	222200	419192
current param	109809	159057	232929	429921
extra param	**10729 (2.56%-10.83%)**			



(a) Results of different embedding lengths (b) Results of different loss weight

Figure 6: Hyper-parameter analysis.

D.3.2 Ablation Results of Key Components in TimeEmb

In this section, we create different versions of the model by altering or removing specific components from TimeEmb. The results are available in Table 11. **Random**: The embedding bank is randomly initialized between training and testing. **Zero/Mean**: The embedding bank is fixed to zeros or the global mean value, respectively. **"w/o $\boldsymbol{X}_s$"** represents removing the time-invariant embedding $\boldsymbol{X}_s$. **"w/o $\mathcal{H}_\omega$"** indicates removing frequency filter $\mathcal{H}_\omega$. **"w/o RevIN"** refers to removing the reversible instance normalization. It is observed that the time-invariant component embedding contributes most in TimeEmb.

https://doi.org/10.52202/085713-2560

Table 9: Ablation study results. $\boldsymbol{E}$ contains frequency components with top-k amplitude. The best results are in **bold**.

k		5		15		30		40		49(full)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.375	0.393	0.375	0.392	0.374	0.392	0.371	0.390	**0.366**	**0.387**
	192	0.427	0.422	0.425	0.421	0.424	0.420	0.422	0.419	**0.417**	**0.416**
	336	0.465	0.441	0.463	0.440	0.463	0.440	0.459	0.439	**0.457**	**0.436**
	720	0.486	0.473	0.472	0.467	0.463	0.463	0.460	0.461	**0.459**	**0.460**
	avg	0.438	0.432	0.434	0.430	0.431	0.429	0.428	0.427	**0.425**	**0.425**
ETTm2	96	0.167	0.247	0.166	0.247	0.166	0.247	0.165	0.244	**0.163**	**0.242**
	192	0.238	0.295	0.236	0.293	0.230	0.288	0.230	0.287	**0.226**	**0.285**
	336	0.293	0.330	0.292	0.328	0.288	0.326	0.287	0.324	**0.286**	**0.324**
	720	0.393	0.388	0.393	0.387	0.386	0.384	0.385	0.383	**0.383**	**0.381**
	avg	0.273	0.315	0.272	0.314	0.268	0.311	0.267	0.310	**0.265**	**0.308**
Weather	96	0.158	0.202	0.155	0.197	0.154	0.195	0.152	0.192	**0.150**	**0.190**
	192	0.212	0.250	0.206	0.244	0.205	0.242	0.203	0.240	**0.200**	**0.238**
	336	0.268	0.291	0.265	0.288	0.264	0.287	0.261	0.284	**0.259**	**0.282**
	720	0.350	0.344	0.347	0.342	0.346	0.341	0.344	0.340	**0.339**	**0.336**
	avg	0.247	0.272	0.243	0.268	0.242	0.266	0.240	0.264	**0.237**	**0.262**
Electricity	96	0.157	0.250	0.151	0.247	0.145	0.241	0.140	0.236	**0.136**	**0.231**
	192	0.171	0.261	0.168	0.261	0.161	0.255	0.156	0.250	**0.153**	**0.246**
	336	0.188	0.278	0.185	0.278	0.179	0.273	0.173	0.267	**0.170**	**0.264**
	720	0.231	0.315	0.229	0.316	0.220	0.307	0.214	0.302	**0.208**	**0.297**
	avg	0.187	0.276	0.183	0.276	0.176	0.269	0.171	0.264	**0.167**	**0.260**

Table 10: Ablation study results. $\boldsymbol{E}$ contains frequency components filtered with low-passing filtering by γ. The best results are in **bold**.

γ		0		0.3		0.6		0.9		1	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.376	0.391	0.370	0.390	0.368	0.389	0.367	0.388	**0.366**	**0.387**
	192	0.431	0.422	0.420	0.418	0.420	0.417	0.418	0.417	**0.417**	**0.416**
	336	0.470	0.441	0.459	**0.437**	**0.457**	**0.437**	**0.457**	**0.437**	**0.457**	**0.436**
	720	0.487	0.471	0.465	0.463	0.461	0.461	**0.459**	**0.461**	**0.459**	**0.460**
	avg	0.441	0.431	0.429	0.427	0.427	0.426	**0.425**	0.426	**0.425**	**0.425**
ETTm2	96	0.173	0.252	0.164	0.243	0.164	0.243	0.164	0.243	**0.163**	**0.242**
	192	0.237	0.294	0.227	**0.285**	0.228	0.286	0.227	**0.285**	**0.226**	**0.285**
	336	0.295	0.332	0.286	0.324	0.287	**0.324**	**0.286**	**0.324**	**0.286**	**0.324**
	720	0.391	0.389	**0.382**	0.382	0.383	0.382	0.383	0.382	0.383	**0.381**
	avg	0.274	0.317	0.267	0.309	0.267	0.309	0.266	0.309	**0.265**	**0.308**
Weather	96	0.182	0.221	0.151	0.191	0.151	0.191	**0.150**	**0.190**	**0.150**	**0.190**
	192	0.228	0.259	0.202	0.240	0.201	0.239	0.201	**0.238**	**0.200**	**0.238**
	336	0.282	0.299	0.261	0.284	0.260	0.283	**0.259**	0.283	**0.259**	**0.282**
	720	0.356	0.347	0.344	0.340	0.343	0.339	0.343	0.339	**0.339**	**0.336**
	avg	0.262	0.282	0.240	0.264	0.239	0.263	0.238	0.263	**0.237**	**0.262**
Electricity	96	0.178	0.259	0.141	0.236	0.139	0.234	0.138	0.233	**0.136**	**0.231**
	192	0.184	0.266	0.158	0.251	0.156	0.249	0.154	0.248	**0.153**	**0.246**
	336	0.200	0.282	0.175	0.268	0.172	0.266	0.171	0.265	**0.170**	**0.264**
	720	0.241	0.316	0.214	0.301	0.211	0.299	0.210	0.298	**0.208**	**0.297**
	avg	0.201	0.281	0.172	0.264	0.170	0.262	0.168	0.261	**0.167**	**0.260**

D.4 Hyper-Parameter Analysis

We conduct experiments to evaluate the impact of essential hyper-parameters of TimeEmb in prediction performance, including the number of embeddings M and loss weight α. The hyper-parameters are adjusted individually for each setting based on the performance displayed in Figure 6. Full results can be referred to Table 12 and Table 13.

For the number of embeddings of embedding bank $\boldsymbol{E}$, we set $M \in \{6, 12, 24, 96\}$, and the results are displayed in Figure 6. From the results, it can be observed that, (1) The variation in M settings affects model performance slightly, highlighting the TimeEmb's robustness and its ability to perform well with minimal manual tuning, making it both user-friendly and easy to deploy. (2) Different datasets exhibit distinct characteristics due to their different periodic patterns. For dataset ETTm2, the best performance is achieved when $M = 96$, as the shorter time intervals require a finer granularity of

Table 11: Ablation study results of key modules. The best results are in **bold**.

Dataset		ETTh1		ETTm2		Weather		Electricity	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
TimeEmb	96	**0.366**	**0.387**	**0.163**	**0.242**	0.150	**0.190**	**0.136**	**0.231**
	192	**0.417**	**0.416**	**0.226**	**0.285**	0.200	**0.238**	**0.153**	**0.246**
	336	0.457	**0.436**	**0.286**	**0.324**	0.259	**0.282**	**0.170**	**0.264**
	720	**0.459**	**0.460**	**0.383**	**0.381**	0.339	**0.336**	**0.208**	**0.297**
	avg	**0.425**	**0.425**	**0.265**	**0.308**	0.237	**0.262**	**0.167**	**0.260**
Random	96	0.407	0.415	0.240	0.318	0.197	0.237	0.242	0.344
	192	0.453	0.443	0.299	0.351	0.249	0.279	0.251	0.354
	336	0.499	0.471	0.361	0.386	0.314	0.321	0.291	0.393
	720	0.550	0.524	0.452	0.434	0.422	0.384	0.370	0.455
	avg	0.477	0.463	0.338	0.372	0.296	0.305	0.289	0.387
Zero	96	0.405	0.414	0.240	0.317	0.198	0.237	0.239	0.342
	192	0.452	0.442	0.299	0.351	0.248	0.279	0.248	0.352
	336	0.497	0.470	0.360	0.385	0.314	0.321	0.289	0.391
	720	0.547	0.523	0.450	0.433	0.421	0.383	0.367	0.453
	avg	0.475	0.462	0.337	0.372	0.295	0.305	0.286	0.385
Mean	96	0.397	0.410	0.204	0.289	0.175	0.221	0.241	0.339
	192	0.444	0.439	0.266	0.327	0.220	0.261	0.250	0.349
	336	0.488	0.464	0.328	0.364	0.267	0.296	0.291	0.387
	720	0.537	0.517	0.427	0.418	0.346	0.348	0.374	0.453
	avg	0.467	0.458	0.306	0.350	0.252	0.282	0.289	0.382
w/o. $\boldsymbol{X}_s$	96	0.376	0.391	0.173	0.252	0.182	0.221	0.178	0.259
	192	0.431	0.422	0.237	0.294	0.228	0.259	0.184	0.266
	336	0.470	0.441	0.295	0.332	0.282	0.299	0.200	0.282
	720	0.487	0.471	0.391	0.389	0.356	0.347	0.241	0.316
	avg	0.441	0.431	0.274	0.317	0.262	0.282	0.201	0.281
w/o. $\mathcal{H}_{\boldsymbol{\omega}}$	96	**0.366**	0.391	0.164	0.244	0.151	0.192	0.137	0.233
	192	**0.417**	0.421	0.229	0.287	0.201	0.239	0.155	0.249
	336	**0.451**	0.441	**0.286**	**0.324**	0.259	0.283	0.172	0.269
	720	0.492	0.474	0.385	0.382	0.341	0.337	0.210	0.299
	avg	0.432	0.432	0.266	0.309	0.238	0.263	0.169	0.263
w/o. RevIN	96	0.383	0.403	0.193	0.286	**0.147**	0.191	0.137	0.235
	192	0.450	0.449	0.283	0.352	**0.194**	**0.238**	0.154	0.252
	336	0.485	0.463	0.427	0.449	**0.248**	0.290	0.171	0.271
	720	0.517	0.511	0.516	0.482	**0.326**	0.346	0.213	0.306
	avg	0.459	0.457	0.355	0.392	**0.229**	0.266	0.169	0.266

embedding to capture temporal patterns. For datasets ETTh1 and Weather, $M = 24$ yields optimal results, effectively capturing data complexity while preventing overfitting.

For the loss weight, we conduct experiments with $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$. Results in Figure 6 show that an optimal value of α improves TimeEmb's performance. Combining both time domain and frequency domain losses outperforms using time domain loss alone ($\alpha = 0$), highlighting that integrating information from both domains enhances TimeEmb's ability to capture diverse patterns in the time series data.

We examine the impact of several key hyperparameters on TimeEmb 's performance: the number of embeddings in the embedding bank, denoted as "M", and the loss weight in the optimization objective, denoted as "α". The detailed results are presented in Table 12 and Table 13. It demonstrates that appropriate hyper-parameters can enhance the performance of TimeEmb.

D.5 Visualization of the Learned Embeddings

We present the time-invariant component embeddings in Figure 7. The "x" in the term "hour x" represents the hour-index of the last timestep of the input series. Figure 7 depicts the distinct embeddings learned from various datasets and channels. For instance, Figure 7 (a) presents the time-invariant components learned in channel 302 of the electricity dataset, where the hour index is 5. In contrast, Figure 7 (b) shows the time-invariant components in channel 15 of the Weather dataset corresponding to the input series, with the hour index being 2. These embeddings, derived from the global sequence, capture the time-invariant components, offering vital supplementary information to the model and enabling a better understanding of the stable patterns within the time series data.

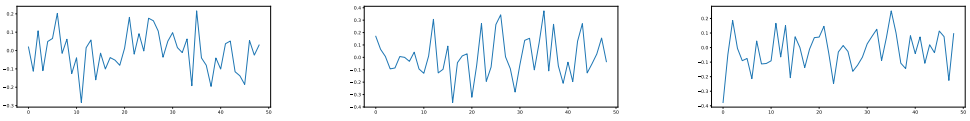
https://doi.org/10.52202/085713-2560

Table 12: Influence of the number of time-invariant embedding M. The best results are in **bold**.

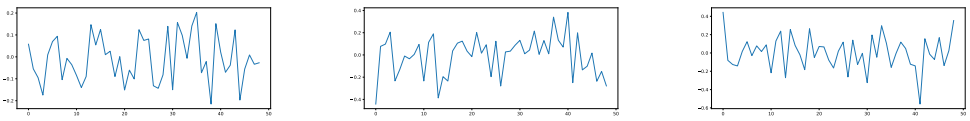
M		6		12		24		96	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.372	0.389	0.370	0.388	**0.366**	**0.387**	0.371	0.390
	192	0.424	0.419	0.422	0.417	**0.417**	**0.416**	0.420	0.417
	336	0.464	0.437	0.461	0.436	**0.457**	**0.436**	0.458	0.437
	720	0.481	0.465	0.464	0.462	**0.459**	**0.460**	0.460	0.461
	avg	0.435	0.428	0.429	0.426	**0.425**	**0.425**	0.427	0.426
ETTm2	96	0.168	0.248	0.167	0.246	0.164	0.243	**0.163**	**0.242**
	192	0.232	0.290	0.231	0.289	0.227	**0.285**	**0.226**	**0.285**
	336	0.291	0.328	0.290	0.327	**0.285**	**0.323**	0.286	0.324
	720	0.387	0.385	0.385	0.383	**0.383**	0.382	**0.383**	**0.381**
	avg	0.270	0.313	0.268	0.311	**0.265**	**0.308**	**0.265**	**0.308**
Weather	96	0.159	0.199	0.153	0.193	**0.150**	**0.190**	0.157	0.198
	192	0.205	0.242	0.202	0.239	**0.201**	**0.238**	0.209	0.246
	336	0.262	0.284	**0.259**	**0.282**	**0.259**	**0.282**	0.266	0.288
	720	0.342	0.338	**0.339**	**0.336**	0.342	0.339	0.349	0.344
	avg	0.242	0.266	**0.238**	0.263	**0.238**	**0.262**	0.245	0.269
Electricity	96	0.140	0.235	0.138	0.234	**0.136**	**0.231**	**0.136**	**0.231**
	192	0.157	0.249	0.155	0.248	**0.153**	**0.246**	**0.153**	0.247
	336	0.173	0.266	0.172	0.266	**0.170**	**0.264**	**0.170**	**0.264**
	720	0.212	0.299	0.212	0.300	**0.208**	0.297	**0.208**	**0.296**
	avg	0.171	0.262	0.169	0.262	**0.167**	**0.260**	**0.167**	**0.260**

Table 13: Influence of loss weight α. The best results are in **bold**.

α		0		0.25		0.5		0.75		1	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.382	0.402	**0.366**	0.389	**0.366**	0.388	**0.366**	**0.387**	0.367	**0.387**
	192	0.423	0.424	0.418	0.419	0.418	0.417	**0.417**	**0.416**	**0.417**	**0.416**
	336	0.463	0.444	0.463	0.440	0.460	0.438	**0.457**	0.437	**0.457**	**0.436**
	720	**0.459**	0.460	0.464	**0.457**	0.468	0.462	0.472	0.464	0.474	0.465
	avg	0.432	0.433	**0.428**	**0.426**	**0.428**	**0.426**	**0.428**	**0.426**	0.429	**0.426**
ETTm2	96	0.170	0.251	0.166	0.245	0.165	0.244	**0.164**	**0.243**	**0.164**	**0.243**
	192	0.234	0.293	0.230	0.288	0.228	0.286	**0.227**	**0.285**	**0.227**	**0.285**
	336	0.294	0.333	**0.285**	0.325	0.286	0.324	**0.285**	**0.323**	0.287	0.324
	720	0.405	0.398	0.386	0.386	0.386	0.384	**0.383**	**0.382**	**0.383**	**0.382**
	avg	0.276	0.319	0.267	0.311	0.266	0.310	**0.265**	**0.308**	**0.265**	0.309
Weather	96	0.154	0.197	0.151	0.193	0.151	0.193	**0.150**	**0.190**	**0.150**	**0.190**
	192	0.203	0.243	0.202	0.240	0.201	0.239	**0.201**	**0.238**	**0.201**	**0.238**
	336	0.263	0.288	0.260	0.285	0.260	0.283	**0.259**	0.283	**0.259**	**0.282**
	720	0.344	0.344	**0.342**	0.340	**0.342**	0.340	**0.342**	**0.339**	**0.342**	**0.339**
	avg	0.241	0.268	0.239	0.265	0.239	0.264	**0.238**	0.263	**0.238**	**0.262**
Electricity	96	0.137	0.234	**0.136**	**0.231**	**0.136**	**0.231**	0.137	**0.231**	0.137	0.232
	192	0.155	0.250	**0.153**	**0.246**	**0.153**	**0.246**	0.154	0.247	0.154	0.247
	336	0.172	0.267	**0.170**	**0.264**	**0.170**	**0.264**	0.171	**0.264**	0.171	**0.264**
	720	0.211	0.303	**0.208**	**0.297**	0.209	**0.297**	0.210	0.298	0.210	0.298
	avg	0.169	0.264	**0.167**	**0.260**	**0.167**	**0.260**	0.168	0.260	0.168	0.260



(a) ECL (b) Weather (c) Traffic

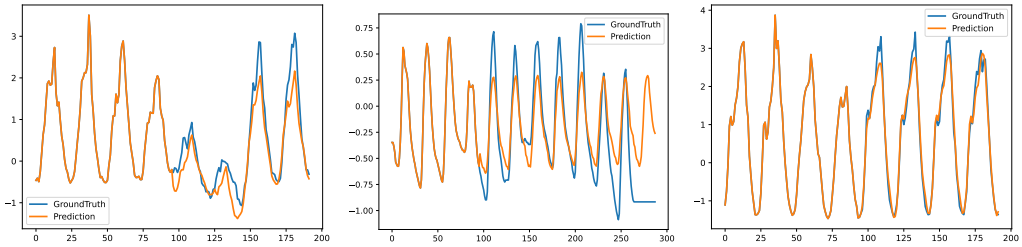


(d) ETTh1 (e) ETTm1 (f) ETTm2

Figure 7: Visualization of the learned time-invariant embeddings $\boldsymbol{X}_s$.

D.6 Visualization of Prediction

We present a prediction showcase on the ETTh2, Electricity, and Traffic datasets, as shown in Figure 8. The predictions closely align with the ground truth, demonstrating that TimeEmb is capable of capturing the complex temporal dependencies of these datasets.



(a) Electricity (b) ETTh2 (c) Traffic

Figure 8: Visualization of TimeEmb prediction and corresponding groundtruth.

E Limitation

While TimeEmb demonstrates strong performance and interpretability in frequency-domain time series modeling, several limitations remain. First, the current design adopts a fixed-resolution embedding bank, which limits its ability to adaptively capture stable patterns across multiple temporal granularities (*e.g.,* hourly, daily, weekly). A more flexible mechanism for multi-scale time-invariant representation would further enhance its capacity for learning multi-periodic structures. Second, the embedding structure is currently discrete, which may restrict its ability to model continuously evolving periodicity. Extending the embedding formulation to a continuous or kernelized representation could enable smoother generalization across unseen temporal slots.

https://doi.org/10.52202/085713-2560

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims in the abstract accurately reflect our contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitations of our work in Appendix E.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide complete experimental details in Appendix C. Additionally, we have shared the full reproducible code in an open repository (link provided in the abstract).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

https://doi.org/10.52202/085713-2560

Answer: [Yes]

Justification: We provide a link to the code and describe how to reproduce the experimental results in the README file of the code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe the complete experimental details and hyperparameter choices in Appendix C and Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The average improvement of TimeEmb over all baseline models is statistically significant at the confidence of 95%.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the computational resource requirements of our proposed method in Figure 1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our research aligns with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have discussed the broader impact of time series forecasting in both abstract and introduction.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

https://doi.org/10.52202/085713-2560

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The code and datasets used in the paper are publicly available and properly credited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

 https://doi.org/10.52202/085713-2560

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.