

ViTime: Foundation Model for Time Series Forecasting Powered by Vision Intelligence

Luoxiao Yang^{1,2} Yun Wang^{3,4} Xinqi Fan⁵ Israel Cohen³ Jingdong Chen⁴ Yue Zhao¹ Zijun Zhang²

Abstract

Time series forecasting (TSF) possesses great practical values in various fields, including power and energy, transportation, etc. TSF methods have been studied based on knowledge from classical statistics to modern deep learning. Yet, all of them were developed based on one fundamental concept, the numerical data fitting. Thus, the models developed have been long known for being problem-specific and lacking application generalizability. A TSF foundation model serving TSF tasks across different applications can reverse such an impression. The central question is then **how to develop such a TSF foundation model**. This paper offers a pioneering study in developing a TSF foundation model and proposes a vision intelligence-powered framework, ViTime, for the first time. In ViTime, a method synthesizing authentic time series periodic and trend patterns is developed to enrich sample pattern diversity. A deep architecture operating TSF in image metric space is designed to achieve significantly enhanced TSF generalizability. Extensive experiments demonstrate ViTime’s SOTA performance across multiple settings. In zero-shot scenarios, ViTime outperforms TimesFM by 9-15%. With just 10% fine-tuning data, ViTime surpasses both foundation models and fully-supervised benchmarks trained on complete datasets, with this performance gap widening further at 100% fine-tuning. Additionally, ViTime exhibits exceptional robustness, handling missing data without imputation and outperforming TimesFM by 20-30% under various data perturbations.

1. Introduction

Time series forecasting (TSF) is a classic but challenging topic that has been vigorously discussed in various application fields, including power and energy (Sharadga et al., 2020), environmental studies (Jacox et al., 2022), transportation studies (Lei et al., 2022), weather forecasting (Yang et al., 2021), stock market analysis (Lin et al., 2011), public healthcare (Liu et al., 2024). Although new heights of accuracies were repeatedly refreshed by new studies (Zhou et al., 2021; Wu et al., 2021; Nie et al., 2022; Zeng et al., 2023; Patro & Agneeswaran, 2024; Wu et al., 2022), most reported methods predominantly relied on a numerical fitting based modeling paradigm so that models were often dataset- or problem-specific and lack of application generalizability. The need to repeatedly train models for various TSF tasks has been the critical barrier of widely applying learning-based TSF methods in practice, especially ones with sophisticated mechanisms. Developing a TSF foundation model capable of serving diverse TSF tasks across different applications is thus of great practical value. The central question then becomes: *how can we develop such a TSF foundation model?*

Studying the TSF foundation model is still in its early stages, and existing efforts observed in literature are mainly devoted to exploring LLM-based and numerical fitting-based models. The LLM-based model leverages the inference capabilities of LLMs for zero-shot TSF tasks, including TimeGPT-1 (Garza & Mergenthaler-Canseco, 2023) and TIME-LLM (Jin et al., 2023). However, the prediction accuracy of LLM-based models heavily depends on the underlying capabilities of LLM, and to achieve optimal performance, the competent large language models, such as GPT-4 or Claude 3.5 (Zhou et al., 2023), are usually employed. Meanwhile, in fine-tuning LLM-based TSF foundation models for handling various downstream tasks demanding higher precision, the computational complexity becomes prohibitively expensive, resulting in a large, redundant, less precise, and price-unfriendly paradigm for the TSF foundation model (Tan et al., 2024).

The numerical fitting-based models are trained by being directly fit into numerical time series data, which manifests that the primary information concerned by these models is

¹Department of Automation, Xi’an University of Technology, China ²Department of Data Science, City University of Hong Kong, Hong Kong SAR ³Electrical and Computer Engineering, Technion, Technion, Israel ⁴School of Marine Science and Technology, Northwest Polytechnical University, China ⁵The Manchester Metropolitan University, UK. Correspondence to: Zijun Zhang <zijzhang@cityu.edu.hk>.

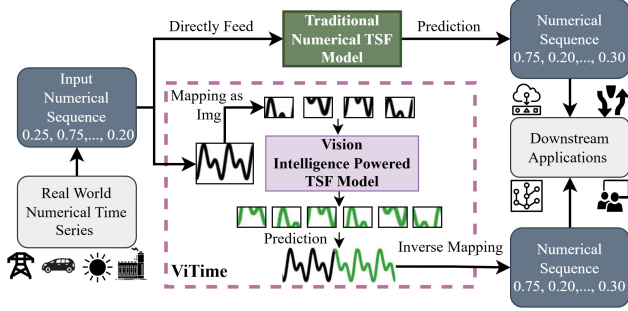


Figure 1: The pipeline comparison of the proposed ViTime with traditional numerical TSF model. It is observable that the proposed ViTime adopts a paradigm shift in TSF by processing time series in the binary image space, which aligns with human cognitive processes in time series analysis.

the numerical correlation along the temporal dimension, e.g., TimesFM (Das et al., 2024), ForecastPFN (Dooley et al., 2024), etc. In contrast, human cognition tends to conjecture trends through remembering correlations between visual representations rather than processing numerical values directly. Studies have shown that the human brain processes visual information more efficiently than numerical data. Pettersson (Pettersson, 1993) discovered that the human brain is more adept at processing visual information than numerical data. Similarly, Dondis (Dondis, 1974) demonstrated that the visual cortex rapidly identifies patterns, shapes, and colors, making processing images and videos quicker than texts and numbers. These findings lead to a hypothetical question: *On the path toward the TSF foundation model, might employing vision intelligence for time series modeling be more effective than conventional numerical methods?*

In addition, training data of TSF tasks typically consist of large-scale real-world datasets (Das et al., 2024), raising a critical question: *Can real-world datasets comprehensively capture the diverse range of universal time series patterns?* Specifically, what kind of foundational capabilities should a TSF foundation model possess to address a universal spectrum of time series problems?

To tackle these challenges, this paper develops a novel vision intelligence-based TSF foundation model, a Visual Time Foundation Model (ViTime), aiming to pioneer a new computational paradigm of building the TSF foundation model from the perspective of vision intelligence. Regarding the computational principle innovation aspect, ViTime operates by transforming numerical time series into binary images, converting numerical temporal correlations into pixel spatial patterns, and solving TSF tasks in binary image space, as shown in Figure 1. To offer a large volume of sufficiently diverse samples for training ViTime, an innovative time-series-data generation method, Real-Time Series (RealTS), is proposed. RealTS categorizes foundational

knowledge of time series analysis into "trend" and "periodicity" and synthesizes training data during the training of ViTime, ensuring it captures essential time series characteristics. Experimental results demonstrate that ViTime can achieve SOTA performance across diverse scenarios, including zero-shot generalization, fine-tuning with limited data, and robustness to data perturbations,

The main contributions of this work are listed as follows:

- **A Novel Vision Intelligence-Based Foundation Model for TSF.** We introduce ViTime, a pioneering TSF foundation model with detailed theoretical analysis that leverages vision intelligence by processing time series in the binary image space. This enables the model to exploit pixel spatial correlation instead of numerical temporal correlation, aligning with human visual cognitive processes.
- **RealTS: Advanced Data Generation and Augmentation for TSF Foundation Modeling.** To address the training-data sample diversity challenge in developing a TSF foundation model, the RealTS, a sophisticated time-series data generation method that synthesizes diverse and high-quality training data, is designed to ensure ViTime can generalize to a wide range of time series patterns.
- **Superior Performance.** Comparisons with SOTA TSF models demonstrate ViTime’s superior performance in zero-shot generalization, fine-tuning study, and robustness against data perturbations.

2. Related work

2.1. Problem-specific model for TSF

The problem-specific TSF methods adopt a fully supervised learning paradigm, where specific models are trained on particular datasets. Early discussions on problem-specific TSF modeling were mainly conducted on classical statistical and machine learning models, such as autoregressive (AR) models and AR variants (Vu, 2007), Splines and their extensions (Lewis & Stevens, 1991), linear regressors (Montgomery et al., 2015), support vector regressor (Montgomery et al., 2015), neural network based regressor (Montgomery et al., 2015), etc. In comparison, the latest TSF studies have shed light on modern deep learning methods, such as recurrent neural network (RNN) and RNN variants (Hewamalage et al., 2021), transformer and various former-based models (Zhou et al., 2021; Wu et al., 2021; Nie et al., 2022), Dlinear (Zeng et al., 2023), etc.

2.2. Foundation model for TSF

Recently, the advent of LLMs like GPT and Llama series, which aim to use a single, generalizable approach to predict

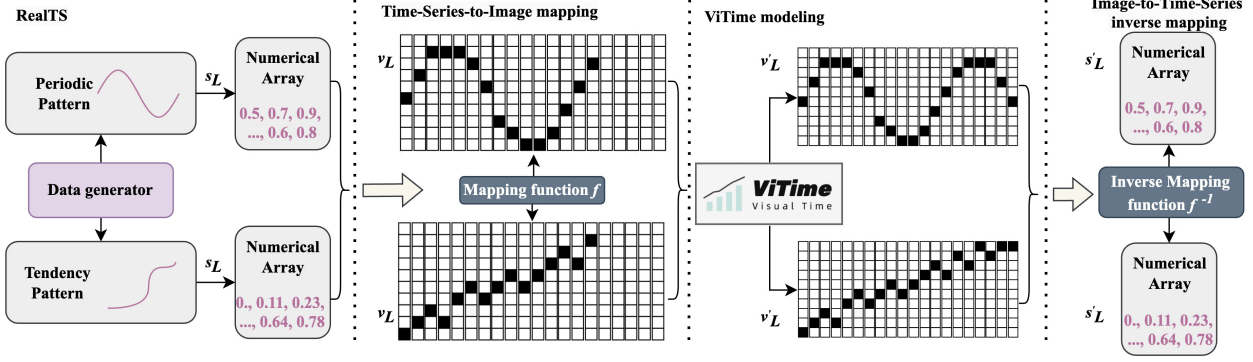


Figure 2: The overall architecture of proposed ViTime. It consists of four key modules: (1) The RealTS synthesis module **generates diverse training samples** s_L by combining fundamental time series patterns of trend and periodicity. (2) The mapping function transforms numerical time series s_L into binary images v_L while preserving temporal relationships through pixel coordinates. (3) The ViTime model performs visual pattern learning and prediction v'_L on the binary image representations v_L . (4) The inverse mapping function converts the predicted binary images v'_L back to numerical time series format s'_L . This architecture enables zero-shot generalization, allowing the model to adapt to various real-world time series tasks without fine-tuning despite being trained only on synthetic data.

across various datasets have sparked interest in developing foundation models for TSF.

Research on foundation models for TSF is still in its early stages, and current approaches can be categorized into LLM-based models, real-world dataset-based models, and synthesized dataset-based models. The LLM-based models leverage the **inference capabilities** of LLMs for zero-shot TSF tasks, including TimeGPT-1 (Garza & Mergenthaler-Canseco, 2023) and TIME-LLM (Jin et al., 2023). However, their accuracy heavily depends on the underlying capabilities of LLM, resulting in large, redundant, and less precise models (Tan et al., 2024). The real-world dataset-based models aim to train a foundation model on extensive real-world data to achieve zero-shot TSF, exemplified by Google’s TimesFM (Das et al., 2024). Despite being one of the leading TSF foundation models, its reliance on real-world numerical time series data limits its robustness and coverage of various unseen time series patterns. The synthesized dataset-based models employ synthetic data to train a foundation model, e.g., ForecastPFN (Dooley et al., 2024). However, the uncontrollable nature of generated data patterns in real space often leads to lower accuracy for foundation models trained with synthetic data than those trained with real-world data.

3. Method

3.1. Overall architecture

The overall framework of ViTime, schematically illustrated in Fig. 2, comprises four key modules: the RealTS synthesis module, the mapping function, the proposed ViTime model, and the inverse mapping function. To address the

dataset challenge of training a robust TSF foundation model, RealTS synthesizes a vast and diverse set of training samples by categorizing foundational knowledge of time series analysis into “trend” and “periodicity” patterns, which ensures ViTime captures essential time series characteristics across a wide range of scenarios. The core innovation of ViTime lies in its computational principle of **mapping numerical time series into binary images**. This approach allows ViTime to remember temporal pattern correlations through ordered pixel coordinates while maintaining the ability to convert results back to numerical format. The visual modeling process of ViTime learns to extract relevant features and patterns from the time series visual representation, utilizing the historical distributions of the generated binary images to predict future trends. Finally, the inverse mapping function is employed to convert the predicted image back into numerical time series data for further analysis.

In the following sections, we will introduce each component of ViTime in detail: RealTS, mapping & inverse mapping function, and ViTime Model.

3.2. Real time series synthesis

In this paper, we hypothesize that a robust foundation model for TSF should integrate two essential types of time series fluctuation knowledge, the periodic and trend patterns, which encompass the inherent patterns and directional changes in time series data. Real-world datasets, however, often lack representation of the full spectrum of these periodic and trend-based fluctuations, limiting the ability of the model to generalize across different scenarios and effectively learn underlying dynamics.

To address this challenge, we propose a novel time series generation algorithm, RealTS. RealTS systematically generates a large volume of synthetic time series data that exhibit diverse periodic and trend characteristics. The proposed RealTS can facilitate more comprehensive training of foundation models, exposing them to various patterns and improving their ability to generalize to unseen real-world data.

The RealTS algorithm probabilistically selects between generating periodic or trend-based time series. Given the total length L of the synthesized time series, the algorithm determines the data prior hypothesis between periodic φ_p and trend-based φ_t patterns with probability (α) . The distribution of generated time series $P(D)$ is defined as follows:

$$\begin{aligned} \mathbf{s}_L &\sim P(D) = P(\mathbf{s}_L|L) \\ &= \alpha \int P(\mathbf{s}_L|L, B_p) P(B_p|\varphi_p) P(\varphi_p) d\varphi_p \\ &\quad + (1 - \alpha) \int P(\mathbf{s}_L|L, B_t) P(B_t|\varphi_t) P(\varphi_t) d\varphi_t \end{aligned} \quad (1)$$

where \mathbf{s}_L is the synthesized time series with length L ; $P(\varphi)$ represents the prior probability of hypothesis φ ; $P(B|\varphi)$ is the likelihood of observing the data behavior B under hypothesis φ . Data behavior B is introduced to further detail the generation behavior within different data modes. RealTS employs two data behavior modes for periodic hypothesis and three for trend hypothesis as follows:

- **Periodic Hypothesis:** Inverse Fast Fourier Transform Behavior (IFFTB) and Periodic Wave Behavior (PWB).
- **Trend Hypothesis:** Random Walk Behavior (RWB), Logistic Growth Behavior (LGB) and Trend Wave Data Behavior (TWDB)

Detailed formulas for each behavior mode and illustrative examples are provided in Appendix A.

3.3. Binary image-based time series metric space

In ViTime, time series are fed and operated with a binary image form, leveraging a binary image-based time series metric space, as described in Definition 3.1.

Definition 3.1 (Binary image-based time series metric space). The binary image-based time series metric space is defined as a group (V, d) , where V is a set of elements defined in Equation (2):

$$\begin{aligned} \mathcal{V} = \left\{ v \in \mathbb{R}^{c \times h \times L} \mid v_{i,j,k} \in \{0, 1\}, \right. \\ \left. i \in [c], j \in [h], k \in [L], \sum_{j=1}^h v_{i,j,k} = 1 \right\} \end{aligned} \quad (2)$$

where $d : V \times V \rightarrow \mathbb{R}$ is a distance function based on the Earth Mover's Distance (EMD), as defined in Equation (3):

$$\begin{aligned} d(v_1, v_2) \\ = \int_{i=1}^c \int_{k=1}^t \inf_{\gamma \in \Pi(\mathbf{v}_1^{i,1:h,k}, \mathbf{v}_2^{i,1:h,k})} \mathbb{E}_{x,y \sim \gamma} \|x - y\|_1 dk di \end{aligned} \quad (3)$$

where c represents the number of variates, L is the length of the time series, and h is the resolution of V .

To enable the transition from numerical time-series values to the binary image-based metric space, we introduce mapping and inverse mapping functions as follows. Let $\mathcal{S} = \{s \in \mathbb{R}^{c \times L} \mid s_{i,k} \in \mathbb{R}\}$ represent the numerical value space of time series. The Time-Series-to-Image mapping function $f : \mathcal{S} \rightarrow \mathcal{V}$ and the Image-to-Time-Series inverse mapping function $f^{-1} : \mathcal{V} \rightarrow \mathcal{S}$ can be defined as follows:

$$\begin{aligned} \mathbf{v}_{i,1:h,k} &= \mathbf{f}(s_{i,k}) = \langle f_1(s_{i,k}), f_2(s_{i,k}), \dots, f_h(s_{i,k}) \rangle \\ f_j(s_{i,k}) &= \begin{cases} 1, & \text{if } s_{i,k} \geq \text{MS}, j = h \\ 1, & \text{if } s_{i,k} \leq \text{MS}, j = 1 \\ 1, & \text{if } j = \left\lfloor \frac{s_{i,k} + \text{MS}}{\frac{2\text{MS}}{h}} \right\rfloor \\ 0, & \text{otherwise.} \end{cases} \quad j \in [h] \end{aligned} \quad (4)$$

The Image-to-Time-Series inverse mapping function $f^{-1} : \mathcal{V} \rightarrow \mathcal{S}$ can be defined as follows:

$$s_{i,k} = \mathbf{f}^{-1}(\mathbf{v}_{i,1:h,k}) = \sum_{j=1}^h \left((j - 0.5) \frac{2\text{MS}}{h} - \text{MS} \right) v_{i,j,k} \quad (5)$$

where $\text{MS} > 0$ denotes the maximum scale of \mathcal{V} . Before mapping, zero-score normalization is typically applied to the numerical time series $s_{i,k}$ to standardize the scale.

Given that the numerical data synthesized by RealTS are one-channel time series, i.e., $\mathbf{s}_L \in \mathbb{R}^L \in \mathbb{R}^{1 \times L}$, thus the corresponding $\mathbf{v}_L \in \mathbb{R}^{1 \times h \times L}$ is obtained via

$$\mathbf{v}_L = \mathbf{f}(\mathbf{s}_L). \quad (6)$$

3.3.1. SYSTEM ERROR ANALYSIS

The system error (SE) emerges from the bidirectional mapping between discrete space \mathcal{V} and continuous space \mathcal{S} , which inherently impacts prediction fidelity. A rigorous analysis of SE is essential for ensuring reliable and robust predictions in image space \mathcal{V} . We begin our theoretical analysis of SE with Assumption 3.2 and Theorem 3.3.

Assumption 3.2. After applying zero-score normalization, the continuous space follows a standard normal distribution:

$$\mathcal{S} \sim N(\mathbf{0}, \mathbf{I})$$

Theorem 3.3 (System Error Upper Bound). *Given a tensor $\hat{s} \in \mathcal{S} \subset \mathbb{R}^{c \times t}$, the system error defined as $\|f^{-1}(\mathbf{f}(\hat{s})) - \hat{s}\|_1$ satisfies the following bound:*

$$\begin{aligned} SE &:= \mathbb{E} \|f^{-1}(\mathbf{f}(\hat{s})) - \hat{s}\|_1 \leq g(h, MS) \\ &= ct \left[MS \left(\frac{1}{h} (\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS) \right) \right. \\ &\quad \left. + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}} \right] \end{aligned} \quad (7)$$

where Φ denotes the cumulative distribution function of $N(\mathbf{0}, \mathbf{I})$.

Denote $MS \left(\frac{1}{h} (\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS) \right) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}}$ in Equation (7) as the upper bound of SE, whose convergence is guaranteed by Proposition 3.4.

Proposition 3.4 (Asymptotic Convergence with h). *For any $\varepsilon > 0$, there exists $\delta > 0$ such that when $h \rightarrow +\infty$ and $MS \geq \delta$, the SE upper bound converges to zero:*

$$\begin{aligned} \lim_{h \rightarrow +\infty} \left| MS \left(\frac{1}{h} (\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS) \right) \right. \\ \left. + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}} \right| = 0 \end{aligned} \quad (8)$$

The Proposition 3.4 reveals that when we fix MS and increase the spatial resolution h , the upper bound $|g(h, MS)|$ of SE will reduce accordingly. On the other hand, when h increases, the tensor sizes in \mathcal{V} will increase exponentially, leading to higher computational costs. As such, the selection of h must strike a balance between the accuracy of the estimation and the computational feasibility. Since the upper bound of SE decreases with an increase in h , it is generally preferable to choose the largest possible value of h based on available computational resources, resulting in a fixed value of h for a particular computational budget.

Additional theoretical analysis of the proposed binary image-based time series metric space and proofs are offered in Appendix C and Appendix D.

3.4. The proposed ViTime model

Figure 3 presents the architecture of the ViTime network, which comprises three network modules: the Visual Time Tokenizer, the Decoder, and the Refining Module. The time series binary image is first fed into the Visual Time Tokenizer and outputs embedded latent representations. Next, the decoder network is developed to decode latent representations and produce initial prediction results. To improve

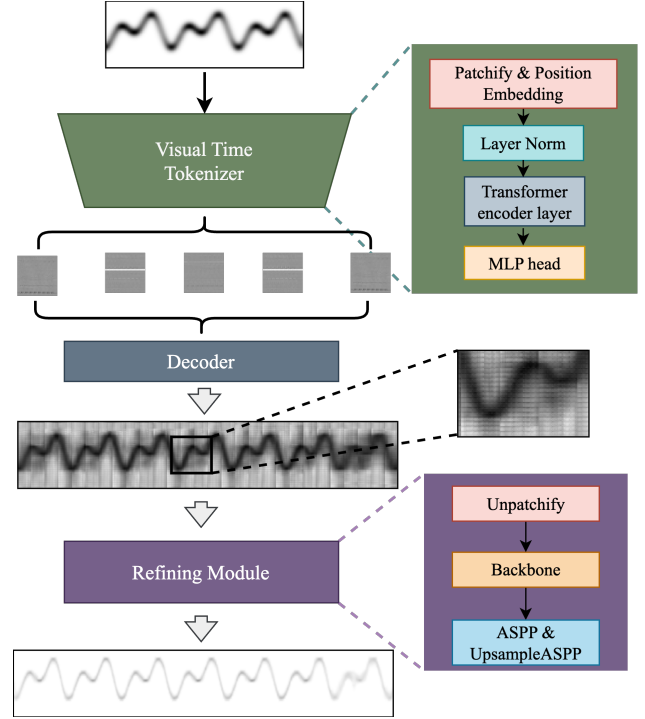


Figure 3: ViTime network. It consists of three key modules: (1) Visual Time Tokenizer that embeds binary time series images into latent representations, (2) Decoder that generates initial predictions from these latent representations, and (3) Refining Module that improves final prediction quality by refining the initial results.

the generative quality of patch junctions, a Refining Module is designed to generate the final smooth prediction results.

Visual Time Tokenizer. The primary role of the Visual Time Tokenizer is to segment masked binary images into multiple patches and map these patches into the feature space. By leveraging the ViT (Dosovitskiy et al., 2020) architecture, the module captures spatial relationships between patches, thereby transforming temporal dependencies of the time series into spatial dependencies within the image space.

Decoder. The Decoder translates the tokenized patches back into the binary pixel metric space, providing an initial prediction where the ViT architecture is also adopted.

Refining Module. The transformer architecture in the Decoder can result in discontinuities at the patch junctions, which may affect the accuracy of the inverse mapping process. To address this issue, the Refining Module building with CNNs is employed. Initially, tokens decoded by Decoder are unpatched and fed into a CNN-based backbone. Next, the ASPP (Chen et al., 2015) module expands the model receptive field. Finally, the output is upsampled to

the binary pixel metric space, generating the final image prediction result.

The modeling process of ViTime is as follows:

$$\mathbf{v}'_{\mathbf{L}} = \text{ViTime}(\mathbf{v}_{\mathbf{L}} \odot \mathbf{M}_{\mathbf{L}}) \quad (9)$$

where $\mathbf{M}_{\mathbf{L}}$ denotes temporal masks.

Loss function. The loss function employed in this study is defined as follows:

$$\mathcal{L} = d(\mathbf{v}'_{\mathbf{L}}, \mathbf{v}_{\mathbf{L}}) + \alpha \text{KLD}(\mathbf{v}'_{\mathbf{L}}, \mathbf{v}_{\mathbf{L}}) \quad (10)$$

where d denotes the distance function defined in Equation (3), KLD denotes Kullback–Leibler divergence, and α is the hyperparameter balance quantity between d and KLD. The combined EMD and KLD loss addresses structural and probabilistic alignment in the binary image space. EMD minimizes spatial discrepancies in \mathcal{V} , counteracting SE from discretization, while KLD refines distributional consistency to mitigate quantization artifacts. This dual approach balances geometric fidelity (via EMD) and statistical accuracy (via KLD), crucial given the resolution-computation trade-off governed by h .

4. Computational experiments

4.1. Experimental Configuration

Datasets and evaluation metrics

Seven popular publicly accessible datasets: Electricity, Traffic, Weather, ETTh1, ETTh2, ETTm1, and ETTm2 (Wu et al., 2021) are employed in computational experiments to validate the effectiveness of the proposed ViTime.

Existing numerical fitting-based TSF foundation models, e.g., TimesFM, are typically pretrained on comprehensive real-world datasets. While the specific nomenclature of the testing set may not be explicitly listed in the training data, there is a possibility that the real-world dataset encompasses similar data sources, potentially leading to issues of test set leakage. To address this concern and ensure a more rigorous and equitable experimental comparison, we propose two novel metrics for zero-shot evaluation, the Rescale-Mean Absolute Error (ReMAE) and Rescale-Mean Squared Error (ReMSE). The fundamental principle underlying ReMAE and ReMSE involves rescaling the test dataset across various time resolutions, as illustrated in Equation (11). The time series interpolation (TSI) method is employed to rescale the original test time series of length T to βT :

$$S_{\beta T} = \text{TSI}(S_T, \text{rescaling factor} = \beta). \quad (11)$$

The formulas for ReMAE and ReMSE are

$$\text{ReMSE} = \frac{\sum_{\beta \in \mathbf{U}} \text{MSE}(S'_{\beta T}, S_{\beta T})}{\text{len}(\mathbf{U})} \quad (12)$$

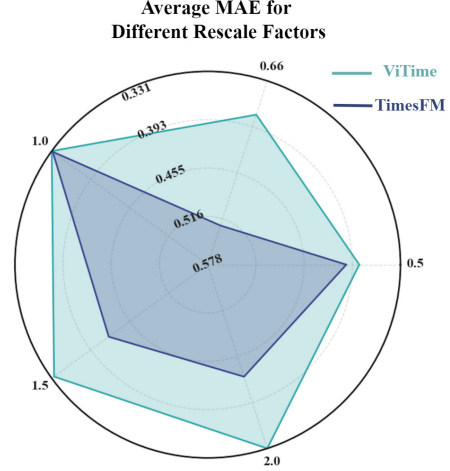


Figure 4: Radar plots comparing the average MAE of ViTime and TimesFM across different rescale factors. The radial axis represents MAE, with lower values (larger radius) indicating better performance. Each axis corresponds to a specific rescale factor.

$$\text{ReMAE} = \frac{\sum_{\beta \in \mathbf{U}} \text{MAE}(S'_{\beta T}, S_{\beta T})}{\text{len}(\mathbf{U})} \quad (13)$$

$$\mathbf{U} = [0.5, 0.66, 1, 1.5, 2].$$

Model setup

The ViTime model is developed using data sequences synthesized by RealTS. During each training epoch, 20,000 sequences are randomly generated. After training, zero-shot testing and fine-tuning are implemented accordingly. For multivariate time series, a channel-independent strategy (Nie et al., 2022) is applied, predicting each variable separately before combining them to form the final multivariate forecast.

The default parameters for the ViTime model are set as follows: $h = 128$, $MS = 3.5$, maximum lookback window $T = 512$, and maximum prediction length $l = 720$. Gaussian blur with a kernel size of $(31, 31)$ is applied to the binary image to facilitate faster convergence. For a fair comparison, all considered models employ a look-back length of 512 to forecast future sequences of lengths 96, 192, 336, 720. More details on training are available in the Appendix B.

4.2. Comparison of ViTime to SOTA TSF Benchmarks Under Zero-shot Setting

To assess the zero-shot performance of ViTime, we consider two leading SOTA TSF models: TimesFM (Das et al., 2024), introduced by Google Research in April 2024 and ac-

Table 1: Comparisons of zero-shot forecasting results. The best zero-shot results are **bolded**, the second best are underlined and the result better than supervised PatchTST are marked with **purple**.

Method	ViTime-1072		ViTime		PatchTST-ZS		TimesFM		PatchTST	
Training data	RealTS		RealTS		RealTS		Real world		Real world	
Supervision	Zero-shot		Zero-shot		Zero-shot		Zero-shot		Supervised	
Metric	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
Electricity	0.246	0.320	0.263	<u>0.340</u>	1.414	0.920	0.367	0.404	<i>0.153</i>	<i>0.247</i>
Traffic	0.677	0.365	<u>0.760</u>	<u>0.408</u>	2.053	1.002	0.875	0.518	<i>0.353</i>	<i>0.256</i>
Weather	0.262	0.276	<u>0.263</u>	<u>0.279</u>	0.910	0.584	0.284	0.307	<i>0.222</i>	<i>0.263</i>
ETTh1	0.464	0.446	<u>0.490</u>	<u>0.453</u>	1.477	0.903	0.513	0.476	<i>0.447</i>	<i>0.449</i>
ETTh2	0.330	0.373	<u>0.338</u>	<u>0.380</u>	1.096	0.775	0.355	0.391	<i>0.365</i>	<i>0.406</i>
ETTM1	0.464	0.420	<u>0.490</u>	<u>0.430</u>	1.295	0.797	0.670	0.503	<i>0.359</i>	<i>0.386</i>
ETTM2	0.293	<u>0.348</u>	<u>0.297</u>	0.339	0.805	0.612	0.336	0.359	<i>0.257</i>	<i>0.320</i>

Table 2: Comparisons of Fine-tuning forecasting results. FT is short for fine-tuning. The best results are **bolded**, and the second best is underlined.

Method	Data proportion	ETTh1	ETTh2	ETTM1	ETTM2	Electricity	Traffic	Weather
TimesFM (FT)	10%	0.426	0.410	0.388	0.334			
GPT4TS (FT)	10%	0.525	0.421	0.441	0.335			
TIME-LLM (FT)	10%	0.522	0.394	0.427	0.323			
ViTime (FT)	10%	<u>0.424</u>	<u>0.372</u>	<u>0.378</u>	<u>0.316</u>	<u>0.252</u>	<u>0.254</u>	<u>0.252</u>
PatchTST	10%	0.542	0.431	<u>0.466</u>	<u>0.343</u>	<u>0.268</u>	<u>0.286</u>	<u>0.283</u>
PatchTST	100%	0.434	0.381	0.382	0.317	0.253	0.264	0.264
SiMBA	100%	0.433	0.393	0.396	0.327	0.274	0.291	0.281
TIMESNET	100%	0.450	0.427	0.406	0.332	0.295	0.336	0.267
ViTime (FT)	100%	0.408	0.349	0.367	0.300	0.247	0.250	0.251

knowledge as one of the most powerful TSF model as well as PatchTST (Nie et al., 2022), a well-established model in supervised and transfer learning for TSF over the past two years. All models employ a lookback length of 512 to ensure a fair comparison. For this comparison, we consider five variants: (1) ViTime - our proposed TSF foundation model, trained on generative data from RealTS and adopting a zero-shot paradigm; (2) ViTime-1072 - we allow a longer lookback length, 1072, to test the peak accuracy of ViTime. (3) PatchTST-ZS - trained on the same RealTS-generated data as ViTime, using a numerical fitting paradigm to create a zero-shot version of PatchTST. (4) Fully supervised PatchTST - trained on training datasets to serve as a benchmark for maximum achievable accuracy through supervision. (5) TimesFM - powerful TSF foundation model pre-trained on extensive real-world datasets.

Table 1 reports the average zero-shot results. The proposed ViTime can achieve comparable accuracy to the fully supervised PatchTST on unseen forecasting data, which suggests the effectiveness of the proposed ViTime framework. Moreover, it is observable that the performance of ViTime significantly surpasses PatchTST-ZS. Such a result demonstrates that modeling TSF tasks from a vision intelligence perspective can significantly enhance model robustness and accuracy. Finally, ViTime consistently outperforms TimesFM

across various datasets, demonstrating an average improvement of 9% in forecasting accuracy. This advantage becomes even more pronounced in long sequence forecasting scenarios, where ViTime achieves an impressive 15% accuracy improvement.

Figure 4 illustrates the performance of ViTime and TimesFM across different rescaling factors. It is clear that TimesFM achieves high accuracy at rescaling factor = 1, but the accuracy significantly decreases for other factors, indicating potential data leakage issues in the training set. In contrast, ViTime consistently delivers robust prediction results across all rescaling factors.

4.3. Comparison of ViTime to SOTA TSF Benchmarks Under Fine-tuning Settings

While zero-shot results demonstrate the predictive capability of ViTime on unseen data, some high-precision TSF tasks might require further fine-tuning studies to enhance prediction accuracy. Thus, this section focuses on fine-tuning studies across various specialized datasets.

To comprehensively evaluate the fine-tuning performance of ViTime, we compare ViTime with other foundation models and SOTA supervised TSF models. Foundational models including TimesFM (Das et al., 2024), GPT4TS (Zhou et al.,

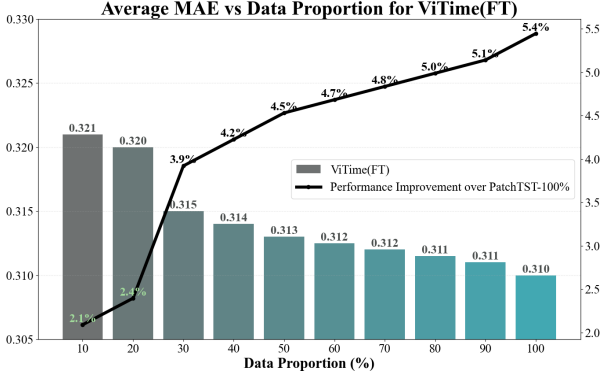


Figure 5: Performance with different fine-tuning data proportion.

2023), and TIME-LLM (Jin et al., 2023) are fine-tuned using 10% of the training data. Recent SOTA-supervised TSF models such as SiMBA (Patro & Agneeswaran, 2024), TIMESNET (Wu et al., 2022), and PatchTST (Nie et al., 2022) use 100% of the training data, as reported in their respective papers. We also fine-tune ViTime using from 10% to 100% of the training data to provide a comprehensive comparison.

Results of the fine-tuning study are provided in Table 2. ViTime fine-tuned with 10% of the training data can outperform other foundational models and the latest supervised models updated on 100% of the training data. Furthermore, as shown in Fig. 5, when the fine-tuning data proportion approaches 100%, the prediction accuracy of ViTime gradually increases and significantly surpasses all existing models, which suggests that ViTime excels in both low-data-availability environments (10% fine-tuning) and full-data-availability scenarios (100% fine-tuning), consistently outperforming both other foundation models and specialized supervised models.

4.4. Comparison of ViTime to SOTA TSF Benchmarks in Robust Inference

To further evaluate the generalizability of ViTime, this section conducts a comparative analysis against TimesFM across four data quality scenarios under zero-shot setting (Figure 6): TSF with original time series, noise-augmented time series, harmonic-augmented time series, and time series of missing data.

In TSF, considering the original time series, ViTime demonstrates superior performance in capturing the underlying periodicities and patterns. In TSF with noise-augmented time series, both models can extract meaningful patterns, but ViTime maintains consistent performance across longer sequences, whereas TimesFM exhibits drift in periodic alignments. For harmonic-augmented time series, ViTime excels

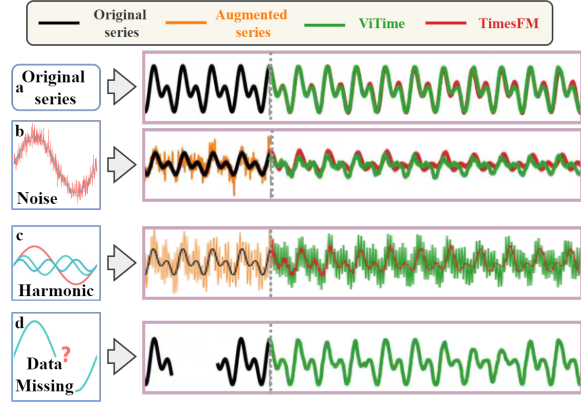


Figure 6: Performance comparison of ViTime versus TimesFM on TSF tasks under various data perturbations: a. Original time series. b. Time series with noises injected. c. Time series with harmonic added. d. Time series with missing data.

by accurately capturing both fundamental and harmonic wave patterns, while TimesFM struggles to disentangle these complex periodic structures.

The most striking difference emerges in the TSF scenario with missing data. As TimesFM is a numerical data fitting-based model, it requires additional imputation techniques. At the same time, ViTime can seamlessly handle these gaps by treating them as zero-valued pixels in its visual representation, showcasing the robust inference capabilities of a vision intelligence-based paradigm.

The complete numerical results and detailed ablation study are presented in Appendix E. We also present our discussion section in Appendix F.

5. Conclusions

This work developed a vision intelligence-powered computational paradigm, ViTime, for developing the TSF foundation model compared with numerical data fitting principles prevalently considered in literature. ViTime was inspired by human visual cognitive processes understanding and analyzing time series. By introducing a paradigm of operating numerical data in image space and the unique deep network based computing pipeline, ViTime elevated the SOTA performance on zero-shot/fine-tuning TSF without relying on prior data samples, demonstrating the great potential for reshaping the computational mechanism in TSF foundation model development. Moreover, data often suffer from diverse contamination and variability in reality. ViTime enabled robust performance under various real-world data perturbations and alterations.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

- Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *3rd International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings*, volume 40, pp. 834–848, 2015.
- Das, A., Kong, W., Sen, R., and Zhou, Y. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 10148–10167. PMLR, 21–27 Jul 2024.
- Dondis, D. A. *A primer of visual literacy*. MIT Press, Cambridge, MA, 1974.
- Dooley, S., Khurana, G. S., Mohapatra, C., Nguyen, A., Yoo, S., Bruckbauer, J., Socher, R., Mortimore, R., and Requeima, J. Forecastpfn: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, volume 36, 2024.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- Garza, A. and Mergenthaler-Canseco, M. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.
- Hewamalage, H., Bergmeir, C., and Bandara, K. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427, 2021.
- Jacox, M. G., Alexander, M. A., Amaya, D., Becker, E., Boer, G., Cai, W., Cotrim da Cunha, L., DeMott, C. A., Dias, D. F., Edwards, C. A., et al. Global seasonal forecasts of marine heatwaves. *Nature*, 604(7906):486–490, 2022.
- Jin, M., Wang, S., Ma, L., Li, P., Yang, K., Wen, Q., Liu, Y., Zhang, L., Ren, R., Du, X., et al. Time-ilm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.
- Lei, W., Alves, L. G., and Amaral, L. A. Forecasting the evolution of fast-changing transportation networks using machine learning. *Nature communications*, 13(1):4252, 2022.
- Lewis, P. A. and Stevens, J. G. Nonlinear modeling of time series using multivariate adaptive regression splines (mars). *Journal of the American Statistical Association*, 86(416):864–877, 1991.
- Lin, W.-Y., Hu, Y.-H., and Tsai, C.-F. Machine learning in financial crisis prediction: a survey. *IEEE Transactions on Systems, Man, and Cybernetics*, 42(4):421–436, 2011.
- Liu, C., Yang, S., Xu, Q., Fan, Z., Ding, Z., Jiang, R., Xie, X., and Song, X. Spatial-temporal large language model for traffic prediction. *arXiv preprint arXiv:2401.10134*, 2024.
- Montgomery, D. C., Jennings, C. L., and Kulahci, M. *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.
- Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.
- Patro, B. N. and Agneeswaran, V. S. Simba: Simplified mamba-based architecture for vision and multivariate time series. *arXiv preprint arXiv:2403.15360*, 2024.
- Pettersson, R. Visual information. *Educational Technology*, 1993.
- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.
- Sharadga, H., Hajimirza, S., and Balog, R. S. Time series forecasting of solar power generation for large-scale photovoltaic plants. *Renewable Energy*, 150:797–807, 2020.
- Tan, M., Merrill, M. A., Gupta, V., Althoff, T., and Hartvigsen, T. Are language models actually useful for time series forecasting? *arXiv preprint arXiv:2406.16964*, 2024.
- Vu, K. M. *The ARIMA and VARIMA time series: Their modelings, analyses and applications*. AuLac Technologies Inc., 2007.
- Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22419–22430, 2021.

- Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.
- Yang, L., Zheng, Z., and Zhang, Z. An improved mixture density network via wasserstein distance based adversarial learning for probabilistic wind speed predictions. *IEEE Transactions on Sustainable Energy*, 13(2):755–766, 2021.
- Yao, H., Wang, Y., Li, S., Zhang, L., Liang, W., Zou, J., and Finn, C. Improving out-of-distribution robustness via selective augmentation. In *International Conference on Machine Learning*, pp. 25407–25437. PMLR, 2022.
- Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.
- Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.
- Zhou, T., Niu, P., Wang, X., Sun, L., and Jin, R. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*, 2023.

A. Details of RealTS

We present RealTS, a versatile framework for synthesizing realistic time series data. RealTS employs multiple data behavior modes under two main hypotheses: periodic (φ_p) and trend (φ_t). This section details the various behavior modes, their configurations, and provides visual examples.

A.1. Periodic Hypothesis Behaviors

Under the periodic hypothesis φ_p , we employ two distinct data behavior modes:

A.1.1. INVERSE FAST FOURIER TRANSFORM BEHAVIOR (IFFTB)

To ensure the synthesized data adequately reflects the variation paradigms of real-world time series, we utilize IFFT as expressed in Equation (14) to simulate the underlying behavior of real-world periodic time series:

$$P(\mathbf{s}_L | L, B_p) |_{B_p = \text{IFFT}} = \iint_{-\infty}^{\infty} \mathbf{N}(\mathbf{A}_m; \mu_{\mathbf{A}_m}, \sigma_{\mathbf{A}_m}^2) \cdot \mathbf{N}(\phi; \mu_P, \sigma_P^2) \times \delta(\mathbf{s}_L - \text{IFFT}(\mathbf{A}_m, \phi, L)) d\phi d\mathbf{A}_m \quad (14)$$

where two empirical distributions of Fourier transform amplitudes and phases, $N(A_m; \mu_{A_m}, \sigma_{A_m}^2)$ and $N(\phi; \mu_P, \sigma_P^2)$, are maintained, and δ denotes the Dirac delta function. By sampling from empirical distributions, we can obtain the amplitude and Phase vector, which is then inversely transformed back to the time domain via IFFT.

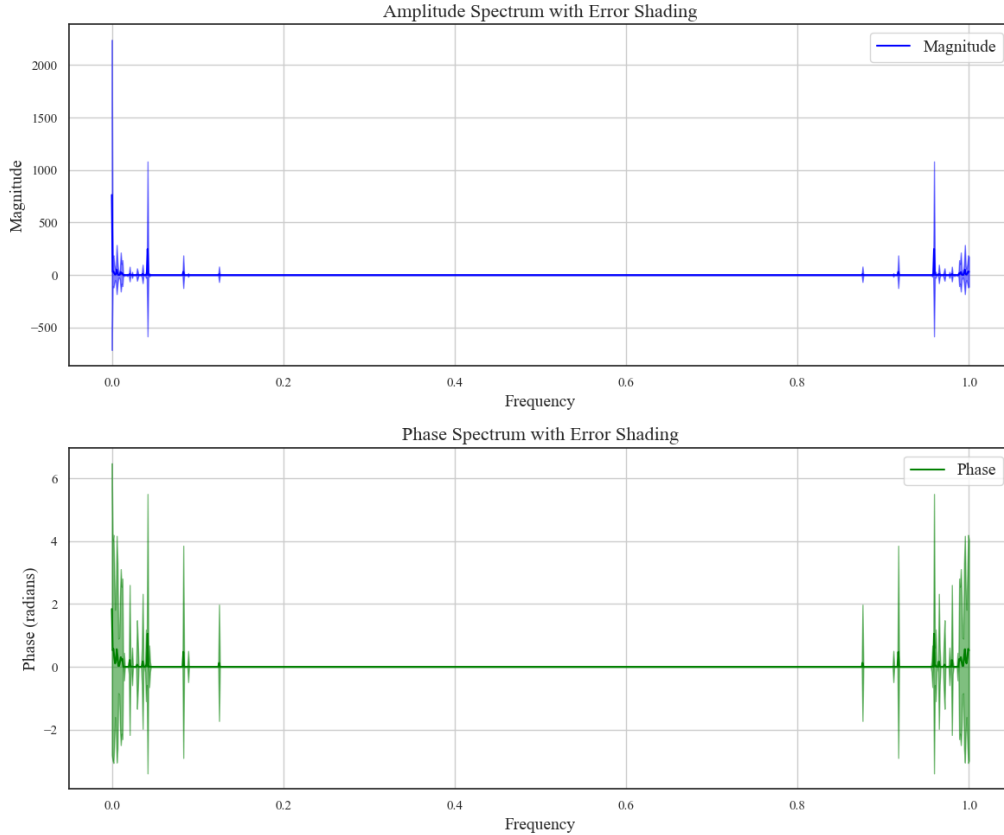


Figure 7: Empirical distribution I employed in IFFTB.

The empirical distributions utilized in \mathbf{A}_m and ϕ are illustrated in Figure 7-Figure 8. During experiments, we randomly select one of two empirical distributions for generating \mathbf{A}_m and ϕ . Figure 9 shows examples of time series generated using IFFTB.

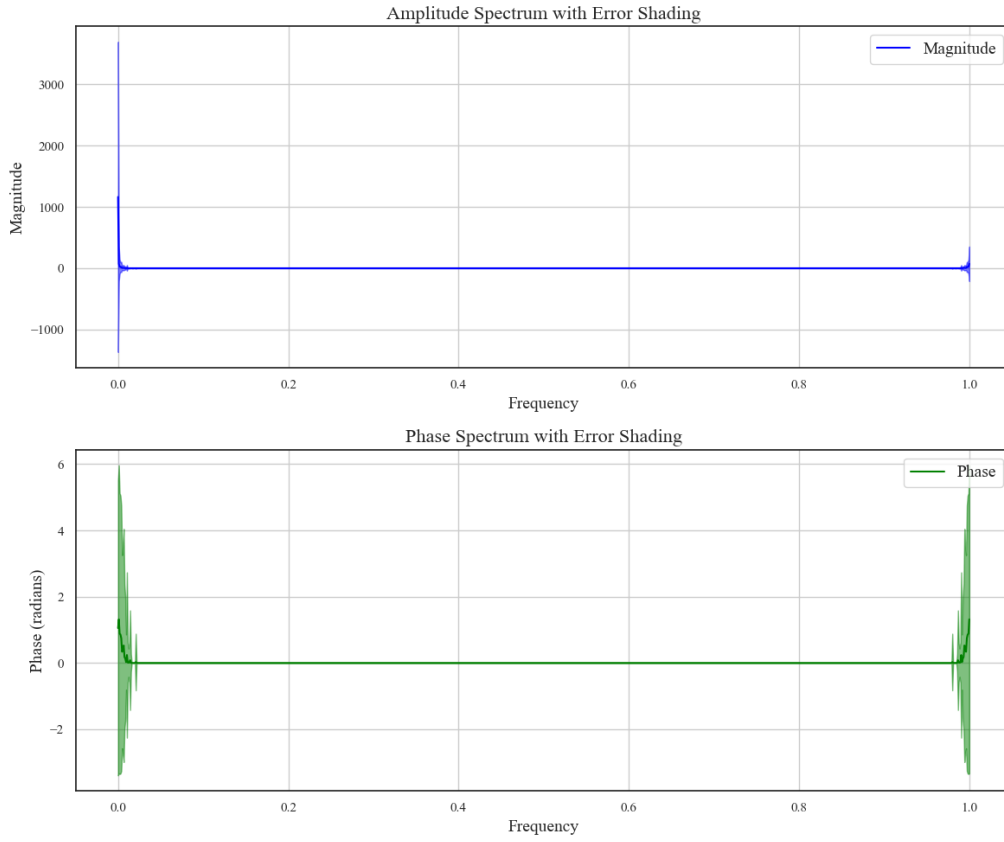


Figure 8: Empirical distribution II employed in IFFTB.

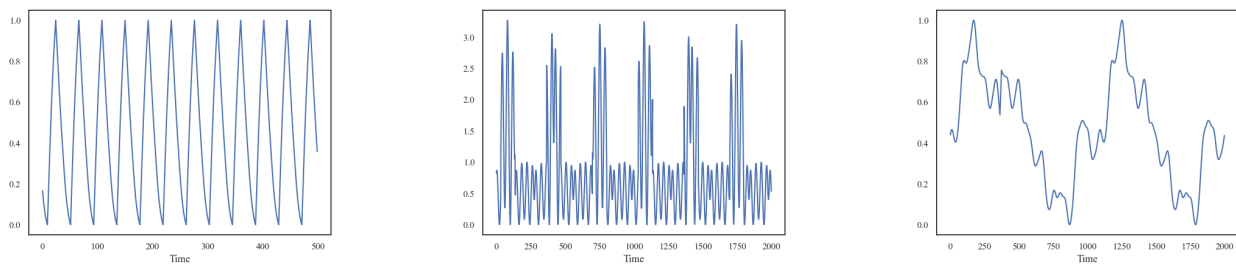


Figure 9: Examples of time series generated using IFFTB.

A.1.2. PERIODIC WAVE BEHAVIOR (PWB)

This behavior generates data by superimposing multiple periodic waves, which is modeled as a sum of sin, cos, and other periodic functions, f_{periodic} , with different frequencies and amplitudes:

$$P(\mathbf{s}_L | L, B_p) |_{B_p = \text{PWB}} = \iint_{-\infty}^{\infty} \mathbf{N} \left(\mathbf{s}_L; \sum_{i=1}^{k_{\text{PWB}}} A_i f_{\text{periodic}}(\omega_i t), \sigma_\epsilon^2 \right) \times \mathbf{P}(\mathbf{A}) \mathbf{P}(\omega) d\omega d\mathbf{A} \quad (15)$$

where $\mathbf{P}(\mathbf{A})$ and $\mathbf{P}(\omega)$ denote predefined prior distributions of amplitudes and frequency; k_{PWB} denotes the number of mixed periodic functions.

For PWB, we define the prior distributions for amplitude and frequency as:

$$\mathbf{A} \sim \mathbf{U}(0.5, 5) \quad (16)$$

$$\ln(\omega) \sim \mathbf{U}(\ln(11), \ln(21)) \quad (17)$$

The parameter k_{PWB} is modeled as:

$$P(k_{\text{PWB}} = k) = \frac{1}{8}, \text{ for } k = 1, 2, \dots, 8 \quad (18)$$

Figure 10 shows examples of time series generated using PWB.

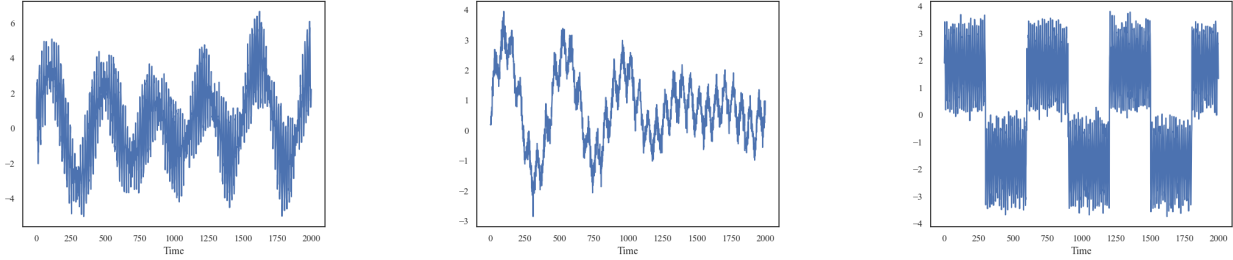


Figure 10: Examples of time series generated using PWB.

A.2. Trend Data Hypothesis Behaviors

Under the trend data hypothesis φ_t , we employ three distinct data behavior modes:

A.2.1. RANDOM WALK BEHAVIOR (RWB)

The RWB models data as a stochastic process where each value is the previous value plus a random step:

$$P(s_i | s_{i-1}, L, B_p) |_{B_p = \text{RWB}} = \mathbf{N}(0, \sigma^2) \quad (19)$$

Figure 11 shows examples of time series generated using RWB.

A.2.2. LOGISTIC GROWTH BEHAVIOR (LGB)

The LGB models data with a logistic growth function, capturing the S-shaped growth pattern:

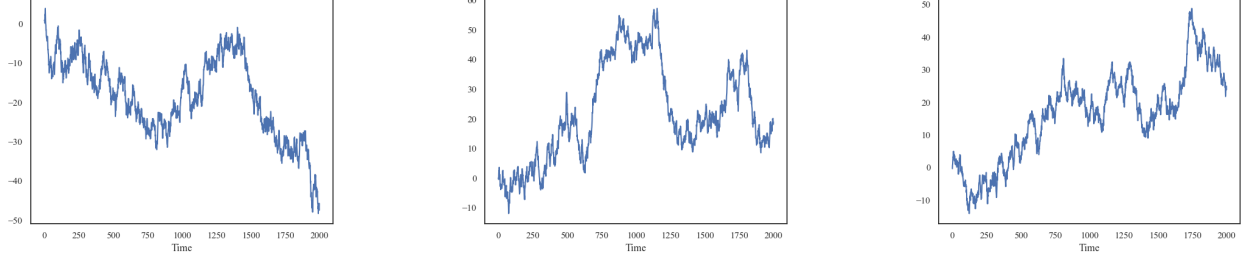


Figure 11: Examples of time series generated using RWB.

$$P(\mathbf{s}_L | L, B_p) |_{B_p=\text{LGB}} = \iint_{-\infty}^{\infty} \mathbf{N}\left(\mathbf{s}_L; \frac{K}{1 + e^{-r(\mathbf{L}-L_0)}}, \sigma_\epsilon^2\right) P(K)P(r)dKdr \quad (20)$$

where $P(K)$ and $P(r)$ denote predefined prior distributions of S-shaped function hyperparameters.

For LGB, we define the probability densities for Carrying Capacity K and Growth Rate r as:

$$\ln(K) \sim U(\ln(1), \ln(10)) \quad (21)$$

$$\ln(r) \sim U(\ln(0.001), \ln(0.1)) \quad (22)$$

Figure 12 shows examples of time series generated using LGB.

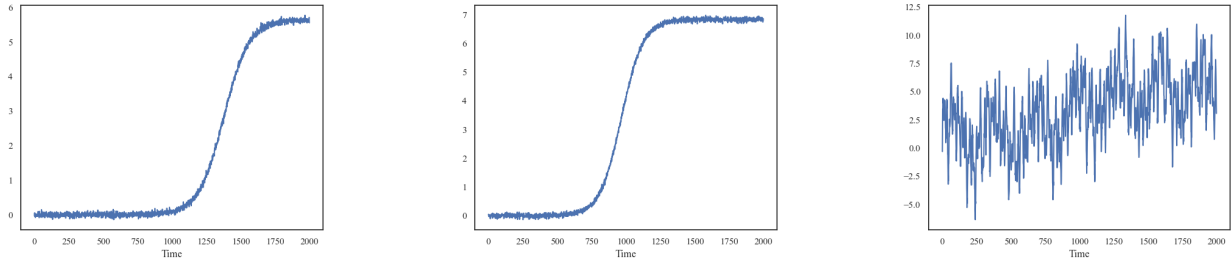


Figure 12: Examples of time series generated using LGB.

A.2.3. TREND WAVE DATA BEHAVIOR (TWDB)

TWDB combines linear trends with periodic fluctuations:

$$P(\mathbf{s}_L | L, B_p) |_{B_p=\text{TWDB}} = \iint_{-\infty}^{\infty} \mathbf{N}\left(\mathbf{s}_L; a\mathbf{L} + b + \sum_{i=1}^{k_{\text{TWDB}}} A_i f_{\text{periodic}}(\omega_i t), \sigma_\epsilon^2\right) \times P(a)P(b)\mathbf{P}(\mathbf{A})\mathbf{P}(\omega)dadbd\mathbf{A}d\omega \quad (23)$$

where $P(a)$, $P(b)$, $P(\mathbf{A})$ and $\mathbf{P}(\omega)$ are predefined prior distributions of hyperparameters.

In the TWDB, we define the probability densities for linear function random variables $P(a)$ and $P(b)$, as well as for the superimposed periodic wave components $P(\mathbf{A})$ and $P(\omega)$. The settings for $P(\mathbf{A})$, $P(\omega)$, and k_{TWDB} are consistent with those used in the PWB module. The probability densities for $P(a)$ and $P(b)$ are detailed below:

$$a \sim U(-1, 1) \quad (24)$$

$$b \sim U(-10, 10) \quad (25)$$

Figure 13 shows examples of time series generated using TWDB.

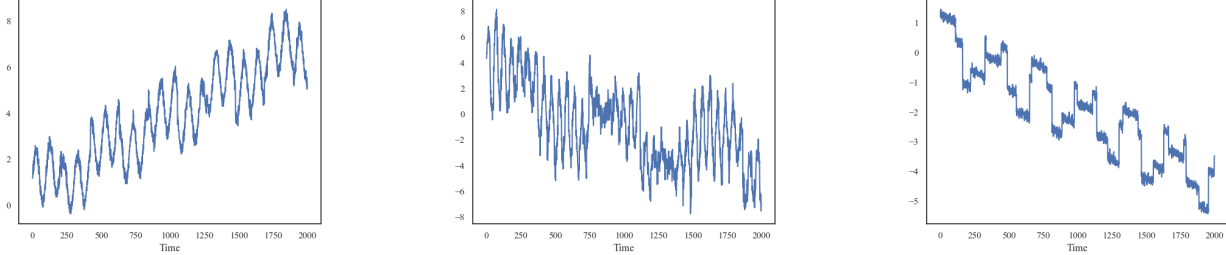


Figure 13: Examples of time series generated using TWDB.

A.3. Data Augmentation Techniques

To enhance the diversity and robustness of synthetic data, we employ various data augmentation techniques, including:

- Multiple period replication: Repeats the generated periodic data over multiple cycles to capture long-term periodic patterns.
- Data flipping: Reverses the time series to create new patterns while preserving underlying characteristics.
- Convolution smoothing and detrending: Removes underlying trends from the data to isolate periodic components, making it easier for the model to learn these patterns.
- Data perturbation: Introduces sudden changes or anomalies into the data, simulating real-world disturbances and improving the model’s ability to handle unexpected variations.

More details of RealTS are offered in the code part of the Supplementary Material.

B. Training configuration

B.1. ViTime model structure

Table 3: Details of model architecture

Module	Embed_dim	Depth	Patch size	Num_heads
Visual Time Tokenizer	768	9	(4,32)	12
Decoder	384	4	\	12

The detailed network configuration of the proposed ViTime is reported in Table 3.

B.2. Training details

We incorporate the following technique details during training.

Data normalization. To ensure ViTime can effectively capture patterns involving sudden changes, an in-sequence data normalization based on L2 normalization is implemented. By normalizing each sequence within the data sequence, the model can pay more attention to abrupt variations. The normalization process is defined as follows:

Table 4: Numerically Solved Optimal MS^*

Resolution h	Optimal MS^*		
	$k = 1$	$k = 1.5$	$k = 2$
32	2.1	2.62	3.03
64	2.38	2.95	3.41
128	2.64	3.26	3.76
256	2.88	3.53	4.08
512	3.09	3.79	4.38

$$\mathbf{S}_L = \frac{\mathbf{S}_L - \text{mean}(\|\mathbf{S}_{1:T}\|_2)}{\text{std}(\mathbf{S}_{1:T})} \quad (26)$$

Temporal resolution enhancement. To improve the temporal resolution in the binary image metric space, the input sequence is first linearly interpolated to twice its original length ($2L$), so that, the granularity of the temporal dimension is increase. Moreover, due to the inherent sparsity of the binary image space, most patches exhibit low-level information density. To improve the information density, Gaussian blurring is applied to the input binary images before being fed into ViTime, thereby reducing sparsity and enhancing information density.

C. Theoretical analysis and Proof

C.1. Theoretical analysis of MS

Proposition C.1 investigates how the upper bound of SE varies with a MS given a fixed value of h , which provides a theoretical guidance to choose the best MS under different computational budgets (h).

Proposition C.1 (Optimal MS Selection). *For fixed h , there exists a unique optimal threshold MS^* minimizing the SE upper bound, characterized by:*

$$\frac{1}{h} (\Phi(MS^*) - \Phi(-MS^*)) - 2 + 2\Phi(MS^*) + \frac{MS^*}{h} \sqrt{\frac{2}{\pi}} e^{-\frac{MS^{*2}}{2}} = 0 \quad (27)$$

C.1.1. GENERALIZATION TO SCALED VARIANCE CASE

The fidelity of predictions in binary image space \mathcal{V} heavily depends on the bidirectional mapping between discrete space \mathcal{V} and continuous latent space \mathcal{S} . A key challenge arises from the SE, which quantifies the discrepancy between the original continuous representation and its reconstructed version after discretization. While Assumption 3.2 assumes $\mathcal{S} \sim \mathcal{N}(0, \mathbf{I})$, real-world scenarios often exhibit larger variance in the latent space due to factors such as dataset shifts or model miscalibration. This motivates our analysis of SE under the generalized assumption $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$, where $k > 1$ captures the variance scaling.

Proposition C.2 (Optimal Threshold under Variance Scaling). *Under the assumption $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$ with $k > 1$, the optimal threshold MS^* that minimizes the SE upper bound is characterized by the following condition:*

$$\frac{1}{h} \left(\Phi\left(\frac{MS^*}{\sqrt{k}}\right) - \Phi\left(-\frac{MS^*}{\sqrt{k}}\right) \right) - 2 + 2\Phi\left(\frac{MS^*}{\sqrt{k}}\right) + \frac{MS^*}{h} \sqrt{\frac{2}{\pi k}} e^{-\frac{(MS^*)^2}{2k}} = 0 \quad (28)$$

Here, $\Phi(\cdot)$ is the CDF of the standard normal distribution, h is the spatial resolution, and k is the variance scaling factor. This result generalizes Proposition C.1 to scenarios where the latent space exhibits larger variability. In practice, it is challenging to find an analytic solution for Equation (32). Thus, the numerical method is employed to obtain solutions of Equation (32) in this work and the corresponding results are reported in Table 4.

C.2. Proofs

C.2.1. PROOF OF THEOREM 3.3

Theorem 3.3 (System Error Upper Bound). Given a tensor $\hat{s} \in \mathcal{S} \subset \mathbb{R}^{c \times t}$, the system error defined as $\|f^{-1}(\mathbf{f}(\hat{s})) - \hat{s}\|_1$ satisfies the following expectation bound:

$$SE := \mathbb{E} \|f^{-1}(\mathbf{f}(\hat{s})) - \hat{s}\|_1 \leq g(h, MS) = ct \left[MS \left(\frac{1}{h} (\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS) \right) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}} \right] \quad (29)$$

where Φ denotes the cumulative density function of $N(\mathbf{0}, \mathbf{I})$.

Proof. Step 1: Error Decomposition

The total error decomposes into quantization error ($|s| \leq MS$) and truncation error ($|s| > MS$):

$$\mathbb{E} \|f^{-1}(f(s)) - s\|_1 = ct \left[\underbrace{\int_{-MS}^{MS} |f^{-1}(f(s)) - s| P(s) ds}_{\text{Quantization}} + 2 \underbrace{\int_{MS}^{\infty} (s - MS) P(s) ds}_{\text{Truncation}} \right]$$

Step 2: Quantization Error Bound

When $|s| \leq MS$, the continuous space is quantized into h bins of width $\Delta = 2MS/h$. The maximum per-element error is $\Delta/2 = MS/h$:

$$\int_{-MS}^{MS} \frac{MS}{h} P(s) ds = \frac{MS}{h} [\Phi(MS) - \Phi(-MS)]$$

Step 3: Truncation Error Calculation

For $s > MS$, the reconstruction error is $s - MS$. Using Gaussian integrals:

$$\int_{MS}^{\infty} s P(s) ds = \frac{1}{\sqrt{2\pi}} e^{-MS^2/2}, \quad \int_{MS}^{\infty} P(s) ds = 1 - \Phi(MS)$$

Thus:

$$2 \int_{MS}^{\infty} (s - MS) P(s) ds = 2 \left(\frac{e^{-MS^2/2}}{\sqrt{2\pi}} - MS[1 - \Phi(MS)] \right)$$

Step 4: Combine Results

Summing both components:

$$SE \leq ct \left\{ \frac{MS}{h} [\Phi(MS) - \Phi(-MS)] + \sqrt{\frac{2}{\pi}} e^{-MS^2/2} - 2MS[1 - \Phi(MS)] \right\}$$

Rearranging terms yields Equation (29). □

C.2.2. PROOF OF PROPOSITION 3.4

Proposition 3.4 (Asymptotic Convergence with h). For any $\varepsilon > 0$, there exists $\delta > 0$ such that when $h \rightarrow +\infty$ and $MS \geq \delta$, the SE upper bound converges to zero:

$$\lim_{h \rightarrow +\infty} \left| MS \left(\frac{1}{h} (\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS) \right) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}} \right| = 0 \quad (30)$$

Proof. As $h \rightarrow \infty$, the $1/h$ term vanishes. Analyze the residual terms for $MS \rightarrow \infty$:

Lemma C.3 (Mill's Ratio). For $MS \gg 0$,

$$1 - \Phi(MS) \sim \frac{e^{-MS^2/2}}{MS\sqrt{2\pi}} \left(1 - \frac{1}{MS^2} + \dots \right)$$

Substituting into the remaining terms:

$$-2MS \left(\frac{e^{-MS^2/2}}{MS\sqrt{2\pi}} \right) + \sqrt{\frac{2}{\pi}} e^{-MS^2/2} = -\sqrt{\frac{2}{\pi}} e^{-MS^2/2} + \sqrt{\frac{2}{\pi}} e^{-MS^2/2} = 0$$

Thus $\forall \varepsilon > 0$, choose δ such that for $MS \geq \delta$, the residual $< \varepsilon$. \square

C.2.3. PROOF OF PROPOSITION C.1

Proposition C.1 (Optimal MS Selection). For fixed h , there exists a unique optimal threshold MS^* minimizing the SE upper bound, characterized by:

$$\frac{1}{h} (\Phi(MS^*) - \Phi(-MS^*)) - 2 + 2\Phi(MS^*) + \frac{MS^*}{h} \sqrt{\frac{2}{\pi}} e^{-\frac{MS^{*2}}{2}} = 0 \quad (31)$$

Proof. **First-Order Condition:** Differentiate the SE bound $\chi(MS, h)$ w.r.t. MS :

$$\frac{\partial \chi}{\partial MS} = \frac{1}{h} [\phi(MS) + \phi(-MS)] + 2\phi(MS) - 2 + \frac{MS}{h} \sqrt{\frac{2}{\pi}} e^{-MS^2/2} - \frac{MS^2}{h} \sqrt{\frac{2}{\pi}} e^{-MS^2/2}$$

Using $\phi(MS) = \frac{1}{\sqrt{2\pi}} e^{-MS^2/2}$ and $\phi(-MS) = \phi(MS)$, we obtain Equation (27).

Second-Order Condition:

$$\frac{\partial^2 \chi}{\partial MS^2} = \sqrt{\frac{2}{\pi}} e^{-MS^2/2} \left(\frac{2 + h - MS^2}{h} \right)$$

- Convex when $MS < \sqrt{h+2}$: $\frac{\partial^2 \chi}{\partial MS^2} > 0$
- Concave when $MS > \sqrt{h+2}$: $\frac{\partial^2 \chi}{\partial MS^2} < 0$

Boundary Behavior:

$$\lim_{MS \rightarrow 0^+} \frac{\partial \chi}{\partial MS} = -2 < 0, \quad \lim_{MS \rightarrow +\infty} \frac{\partial \chi}{\partial MS} = +\infty > 0$$

By Intermediate Value Theorem and convex-concave transition, there exists exactly one MS^* where $\partial \chi / \partial MS = 0$. \square

C.2.4. PROOF OF PROPOSITION C.2

Proposition C.2 (Optimal Threshold under Variance Scaling). Under the assumption $S \sim \mathcal{N}(0, k\mathbf{I})$ with $k > 1$, the optimal threshold MS^* that minimizes the SE upper bound is characterized by the following condition:

$$\frac{1}{h} \left(\Phi \left(\frac{MS^*}{\sqrt{k}} \right) - \Phi \left(-\frac{MS^*}{\sqrt{k}} \right) \right) - 2 + 2\Phi \left(\frac{MS^*}{\sqrt{k}} \right) + \frac{MS^*}{h} \sqrt{\frac{2}{\pi k}} e^{-\frac{(MS^*)^2}{2k}} = 0 \quad (32)$$

Proof. :

Step 1: Scaled Probability Measures

For $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$, the probability density function (PDF) and cumulative distribution function (CDF) are scaled as:

$$P_k(s) = \frac{1}{\sqrt{2\pi k}} e^{-s^2/(2k)}, \quad \Phi_k(x) = \Phi\left(\frac{x}{\sqrt{k}}\right) \quad (33)$$

This scaling reflects the increased spread of the latent space distribution.

Step 2: Modified Quantization Error

The quantization error term, which captures the discretization loss within the threshold $[-MS, MS]$, becomes:

$$\int_{-MS}^{MS} \frac{MS}{h} P_k(s) ds = \frac{MS}{h} [\Phi_k(MS) - \Phi_k(-MS)] \quad (34)$$

This term increases with MS due to the wider interval.

Step 3: Adjusted Truncation Error

The truncation error, representing the loss from discarding values outside $[-MS, MS]$, transforms as:

$$2 \int_{MS}^{\infty} (s - MS) P_k(s) ds = 2 \left[\sqrt{\frac{k}{2\pi}} e^{-MS^2/(2k)} - MS(1 - \Phi_k(MS)) \right] \quad (35)$$

This term decreases with MS as more probability mass is retained.

Step 4: Unified Error Bound

Combining the quantization and truncation errors yields the SE upper bound:

$$SE \leq ct \left\{ \frac{MS}{h} [\Phi_k(MS) - \Phi_k(-MS)] + \sqrt{\frac{2k}{\pi}} e^{-MS^2/(2k)} - 2MS[1 - \Phi_k(MS)] \right\} \quad (36)$$

This bound balances the trade-off between discretization and truncation losses.

Step 5: First-Order Optimality Condition

Differentiating (36) with respect to MS and setting the derivative to zero gives:

$$\begin{aligned} \frac{\partial SE}{\partial MS} &= \frac{1}{h} [\Phi_k(MS) - \Phi_k(-MS)] - 2 + 2\Phi_k(MS) \\ &\quad + \frac{MS}{h} \sqrt{\frac{2}{\pi k}} e^{-MS^2/(2k)} = 0 \end{aligned} \quad (37)$$

Substituting $\Phi_k(x) = \Phi(x/\sqrt{k})$ yields the condition in Equation (32).

Step 6: Uniqueness Guarantee

The second derivative test confirms the uniqueness of MS^* :

$$\frac{\partial^2 SE}{\partial MS^2} = \sqrt{\frac{2}{\pi k^3}} e^{-MS^2/(2k)} \left(2 + h - \frac{MS^2}{k} \right) \quad (38)$$

- Convex region: $MS < \sqrt{k(h+2)} \Rightarrow \frac{\partial^2 SE}{\partial MS^2} > 0$
- Concave region: $MS > \sqrt{k(h+2)} \Rightarrow \frac{\partial^2 SE}{\partial MS^2} < 0$

Boundary behavior ensures a unique solution:

$$\lim_{MS \rightarrow 0^+} \frac{\partial SE}{\partial MS} = -2 < 0, \quad \lim_{MS \rightarrow +\infty} \frac{\partial SE}{\partial MS} = +\infty > 0$$

By the Intermediate Value Theorem and convex-concave transition, exactly one MS^* exists. \square

The generalized Proposition C.2 provides a principled framework for selecting the optimal threshold MS^* when the latent space exhibits larger variance. Key insights include:

- **Scaling Law:** The optimal threshold scales as $MS^* \propto \sqrt{k}$, reflecting the need to accommodate a larger MS^* when facing the dramatically fluctuating application scenarios. We have offered further analysis in Appendix E.1 and Appendix F.3.
- **Practical Guidance:** For real-world applications where k is unknown, adaptive methods can estimate k and dynamically adjust MS^* .
- **Consistency Check:** When $k = 1$, Proposition C.2 reduces to the original Proposition C.1, validating the generalization.

D. Theoretical Advantages of Visual Intelligence for Time Series Forecasting

The integration of visual intelligence into TSF transcends a mere architectural decision, presenting a comprehensive framework rooted in cognitive science, mathematical rigor, and empirical robustness. This section delineates the theoretical underpinnings of ViTime, our proposed model, through three pivotal dimensions: cognitive alignment, pattern preservation, and geometric regularization. Each dimension addresses specific advantages inherent in visual intelligence approaches to TSF.

D.1. Cognitive Alignment

Analysts inherently interpret time series data by discerning visual patterns rather than processing raw numerical values. This cognitive inclination leverages the brain’s proficiency in recognizing spatial relationships and temporal trends visually. ViTime capitalizes on this natural predisposition by converting time series data into binary images, thereby enabling the model to emulate human-like analytical reasoning. This alignment is operationalized through **isometric spatiotemporal mapping**, which ensures that temporal patterns in the time series are preserved as spatial patterns in the visual representation.

Definition D.1 (Spatiotemporal Isometry). For a time series $s \in \mathcal{S}$ and its visual embedding $f(s) \in \{0, 1\}^{h \times T}$, the mapping satisfies:

$$\frac{1}{T} \sum_{t=1}^{T-1} |s_{t+1} - s_t| = \lambda \sum_{x=1}^{T-1} \|f(s)_x - f(s)_{x+1}\|_F$$

where λ is a scale invariance factor that controls the relationship between temporal changes in the time series and spatial changes in the visual representation.

This isometry ensures that slope changes in the time series manifest as proportional edge densities in the image. By aligning the data representation with human cognitive processes, ViTime not only enhances interpretability but also leverages the brain’s innate ability to recognize spatial relationships and temporal trends. This alignment is particularly important for applications where human-AI collaboration is essential, such as in healthcare or finance.

D.2. Pattern Preservation

A paramount challenge in TSF is the preservation of critical temporal structures—such as trends, seasonality, and periodicities—through data transformations. ViTime addresses this challenge by ensuring spectral fidelity, thereby maintaining the essential frequency components of the original time series within its visual representation.

Theorem D.2 (Spectral Fidelity). For any time series $s(t)$ with Fourier transform $\mathcal{F}_t(\omega)$, its visual representation $v(x, y)$ satisfies:

$$\exists C > 0 : \|\mathcal{F}_{x,y}(v) - C \cdot \mathcal{F}_t(s)\|_\infty \leq \epsilon(h),$$

where $\epsilon(h) \rightarrow 0$ as the image height $h \rightarrow \infty$.

Proof. Consider a discrete time series $s(t) \in \mathbb{R}$ of length T . The visual representation $v(x, y) \in \{0, 1\}^{h \times w}$ is constructed via thresholding as follows:

1. **Thresholding Process:** The value range of $s(t)$ is divided into h equal intervals. For each time step t , the pixel $v(t, y)$ is set to 1 if:

$$s(t) \in \left[\frac{y-1}{h} \Delta, \frac{y}{h} \Delta \right),$$

and 0 otherwise, where $\Delta = \max(s) - \min(s)$.

2. **2D Fourier Transform of $v(x, y)$:** The 2D Fourier transform is given by:

$$\mathcal{F}_{x,y}(v)(u, v) = \sum_{x=0}^{T-1} \sum_{y=0}^{h-1} v(x, y) e^{-i2\pi \left(\frac{ux}{T} + \frac{vy}{h} \right)}.$$

3. **Relation to 1D Fourier Transform:** Focusing on the $v = 0$ frequency component:

$$\mathcal{F}_{x,y}(v)(u, 0) = \sum_{x=0}^{T-1} \left(\sum_{y=0}^{h-1} v(x, y) \right) e^{-i2\pi \frac{ux}{T}}.$$

The inner sum $\sum_y v(x, y)$ approximates $s(x)$ scaled by h/Δ , yielding:

$$\sum_{y=0}^{h-1} v(x, y) \approx \frac{h}{\Delta} s(x) + \mathcal{O}\left(\frac{1}{h}\right).$$

Let $C = \frac{h}{\Delta}$, then:

$$\mathcal{F}_{x,y}(v)(u, 0) \approx C \cdot \mathcal{F}_t(s)(u) + \mathcal{O}\left(\frac{T}{h}\right).$$

4. **Error Bound Analysis:** The approximation error comprises:

- **Discretization Error:** $\mathcal{O}(1/h)$ per time step.
- **Boundary Effects:** Diminishing as $h \rightarrow \infty$.

Consequently, the supremum norm of the error over all frequencies is:

$$\epsilon(h) = \max_u |\mathcal{F}_{x,y}(v)(u, 0) - C \cdot \mathcal{F}_t(s)(u)| = \mathcal{O}\left(\frac{T}{h}\right),$$

which tends to zero as h increases.

5. **Extension to 2D Spectrum:** For non-zero v -frequencies, energy concentrates at $v = 0$ due to threshold averaging:

$$\|\mathcal{F}_{x,y}(v)\|_{\infty} = \|\mathcal{F}_{x,y}(v)(\cdot, 0)\|_{\infty},$$

thereby ensuring the bound holds across the entire 2D Fourier spectrum.

□

Theorem D.2 substantiates that ViTime's binary image transformation preserves the critical spectral characteristics of the original time series. This preservation ensures that both high-frequency dynamics, such as short-term fluctuations, and low-frequency components, like long-term trends, are accurately represented and hence effectively captured by the forecasting model.

D.3. Geometric Regularization through Visual Embedding

A key challenge in TSF is the sensitivity of models to noise and perturbations in the input data. Traditional numerical approaches often struggle to maintain robustness under such conditions, as they operate directly on raw time series values without explicit structural constraints. By mapping time series data to binary images, we introduce a form of geometric regularization that smooths the underlying data manifold, making it less sensitive to small perturbations in Theorem D.3. This is particularly important in real-world applications where data quality is often compromised by noise and outliers. The computational results are offered in Appendix E.4.

Theorem D.3 (Geometric Regularization through Visual Embedding). *Let $\mathcal{M}_s \subset \mathbb{R}^T$ denote the manifold of raw time series data, and $\mathcal{M}_v \subset \{0, 1\}^{h \times w}$ its corresponding visual embedding. Assume the time series $s(t)$ has a bounded amplitude range $\Delta = \max(s) - \min(s)$ and is Lipschitz continuous with constant $\text{Lip}(s)$. Then, the visual embedding process imposes the following constraints on \mathcal{M}_v :*

1. **Local Perturbation Stability:** *For any time series $s \in \mathcal{M}_s$ and its visual embedding $v \in \mathcal{M}_v$, small perturbations in the input are compressed into sparse changes in the output:*

$$\forall \epsilon_s > 0, \quad \|s' - s\|_\infty \leq \epsilon_s \implies \|v' - v\|_0 \leq \frac{hT\epsilon_s}{\Delta} + \mathcal{O}(1/h),$$

where $\|v' - v\|_0$ is the Hamming distance between the original and perturbed visual embeddings.

2. **Path Complexity Constraint:** *Any continuous path $\gamma : [0, 1] \rightarrow \mathcal{M}_v$ in the visual embedding space has a bounded number of pixel transitions:*

$$\text{Jump}(\gamma) \leq T \cdot \left\lceil \frac{h \cdot \text{Lip}(s)}{\Delta} \right\rceil,$$

where $\text{Jump}(\gamma)$ counts the number of pixel transitions along the path γ .

Proof. Part 1: Local Perturbation Stability Consider a time series $s(t)$ and its visual embedding $v(x, y)$. A perturbation ϵ_s in the input time series can cause the value $s(t)$ to shift by at most ϵ_s . In the visual embedding, this corresponds to a vertical shift of the active pixel within a column. The maximum vertical shift is bounded by:

$$\Delta y \leq \left\lceil \frac{h\epsilon_s}{\Delta} \right\rceil.$$

Since each column contains exactly one active pixel, the total Hamming distance between the original and perturbed embeddings is:

$$\|v' - v\|_0 \leq T \cdot \Delta y \leq T \left(\frac{h\epsilon_s}{\Delta} + 1 \right).$$

Ignoring the constant term, we obtain:

$$\|v' - v\|_0 \leq \frac{hT\epsilon_s}{\Delta} + \mathcal{O}(1/h).$$

This shows that small input perturbations are compressed into sparse output changes, with sparsity controlled by the bin height h .

Part 2: Path Complexity Constraint Consider a continuous path $\gamma(\tau)$ in \mathcal{M}_v , parameterized by $\tau \in [0, 1]$. The corresponding time series path $s(\tau, t)$ is Lipschitz continuous, so the difference between adjacent time steps is bounded by $\text{Lip}(s)$. In the visual embedding, this translates to a vertical shift of at most:

$$|y_{t+1} - y_t| \leq \left\lceil \frac{h \cdot \text{Lip}(s)}{\Delta} \right\rceil.$$

The total number of pixel transitions along the path is therefore bounded by:

$$\text{Jump}(\gamma) \leq T \cdot \left\lceil \frac{h \cdot \text{Lip}(s)}{\Delta} \right\rceil.$$

This demonstrates that the complexity of paths in \mathcal{M}_v is linearly controlled by the bin height h . □

Table 5: Empirical Forecasting Performance under Different MS Values

				MS			
	2.38	2.64	2.88	3.09	3.50	5.00	6.00
ReMSE	0.4423	0.4404	0.4400	0.4348	0.4178	0.4780	0.4724
ReMAE	0.3818	0.3812	0.3811	0.3788	0.3759	0.3990	0.3959

Theorem D.3 establishes that the visual embedding process introduces geometric regularization by compressing input perturbations and constraining the complexity of paths in the embedding space. This has two key implications:

- **Robustness to Noise:** The local perturbation stability property ensures that small input perturbations result in sparse changes in the visual representation. This makes the model less sensitive to noise, as minor fluctuations in the input are effectively filtered out. By adjusting h , practitioners can control the trade-off between robustness and model complexity. Larger values of h enhance robustness (see in Figure 14) but may increase computational cost.
- **Controlled Complexity:** The path complexity constraint limits the number of pixel transitions in the visual embedding, preventing the model from overfitting to complex patterns in the data. This regularization effect is controlled by the bin height h , which can be tuned to balance robustness and expressiveness.

The geometric regularization theorem provides a rigorous foundation for the robustness and stability of ViTime. By transforming time series data into binary images, ViTime introduces structural constraints that mitigate the effects of noise and prevent overfitting. This theoretical insight not only enhances the interpretability of the model but also guides practical implementation, ensuring reliable performance in real-world applications.

D.4. Summary

ViTime’s adoption of visual intelligence for time series forecasting is a theoretically grounded advancement that effectively addresses the limitations of conventional numerical approaches. By aligning with human cognitive processes, ensuring the preservation of essential temporal patterns, and embedding intrinsic robustness against noise, ViTime establishes a robust and interpretable framework for accurate and reliable forecasting. This triad of theoretical foundations not only enhances forecasting performance but also promotes greater trust and usability in practical applications.

E. Additional results of computational experiments

E.1. Ablation study

E.1.1. ABLATION OF MS

Proposition C.2 establishes the theoretical relationship between the optimal MS threshold and the variance scaling factor k in the latent space. For stationary data ($\mathcal{S} \sim \mathcal{N}(0, \mathbf{I})$, i.e., $k = 1$), Proposition C.2 reveals that with $h = 128$, the optimal MS should be 2.64. However, real-world time series often exhibit non-stationary characteristics. Our analysis of the target variable’s variance after input-based standardization (see KL divergence results in Section 4.2) demonstrates that the effective k value for the prediction horizon falls within $[1.5, 2]$ across all benchmark datasets.

Table 4 provides numerically solved optimal MS^* values under different k and h configurations. For $h = 128$ (our experimental setting) and $k \in [1.5, 2]$, the theoretical optimal MS ranges between 3.26-3.76. This motivates our selection of $MS = 3.5$ as a balanced configuration within this interval.

To validate this choice, Table 5 presents the average relative MSE and MAE across six benchmark datasets. The results demonstrate that $MS = 3.5$ achieves the minimum forecasting error, reducing ReMSE by 4.1% and ReMAE by 1.8% compared to the stationary optimal $MS = 2.64$. This strong alignment between theoretical predictions (Table 4) and empirical performance (Table 5) confirms that our MS selection strategy effectively minimizes system error while accommodating real-world data characteristics.

Table 6: Ablation study of loss function components on prediction performance.

Metric	Loss Configuration		
	EMD Only	JSD+EMD (Ours)	JSD Only
Average ReMAE	0.3941	0.3759	0.3956
Average ReMSE	0.4586	0.4178	0.4637

E.1.2. ABLATION OF LOSS FUNCTION

In this section, we conducted ablation studies on the loss function components using the ViTime architecture. Table 6 compares model performance under three configurations: (1) EMD alone, (2) our proposed loss function in Equation (10), where $\alpha = 0.2$ to balance the quantity level, and (3) JSD alone. The results demonstrate that our dual-objective loss achieves optimal performance on both ReMSE and ReMAE.

E.1.3. ABLATION OF OTHER CONFIGURATION

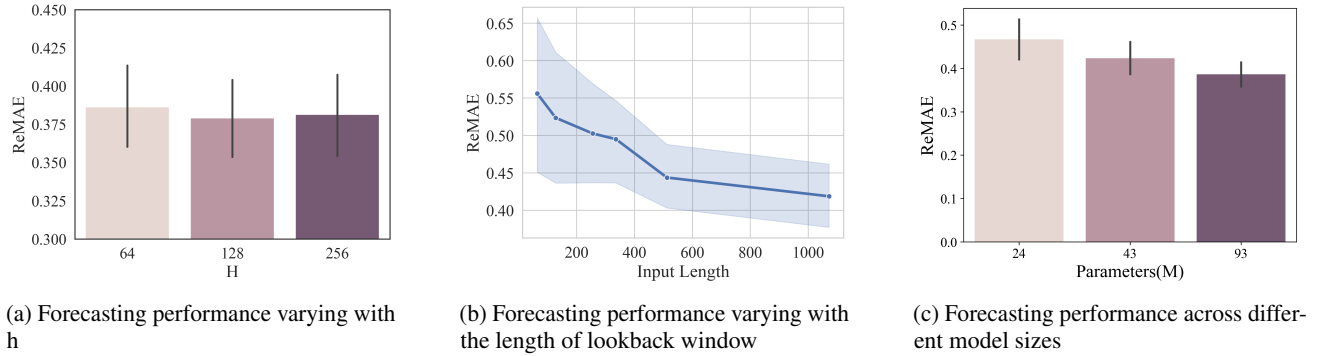


Figure 14: Ablation studies with zero-shot forecasting.

Note: The model size of ViTime used in computational experiments is 93M parameters version.

In this section, we perform several ablation studies to gain deeper insights into ViTime model configuration. The results are reported in Figure 14. Figure 14 a depicts the influence of varying spatial resolutions (h) on model accuracy. Although increasing h slightly improves the prediction results, the associated computational cost increases exponentially. Thus, setting h to 128 is more economical and efficient. Figure 14 b illustrates the effect of different lookback window lengths (T) on prediction accuracy. It is evident that a longer lookback window length significantly enhances the model’s prediction accuracy. Figure 14 c reports the prediction accuracy across different model sizes. The data shows that models with more parameters tend to perform better. Moreover, the proposed ViTime achieves superior performance with only **93M** parameters compared with TimesFM, which is over 200M parameters, further demonstrating the efficiency and effectiveness of ViTime.

E.2. Zero-shot study

The full results of the zero-shot study are reported in Table 7. We also illustrate zero-shot TSF examples with prediction length equals 720 of the proposed ViTime versus TimesFM in Figure 18 - Figure 23. It is observable that ViTime consistently demonstrates superior zero-shot prediction performance compared to TimesFM across a range of rescale factors.

E.3. Fine-tuning study

Complete results of the fine-tuning study are reported in Table 8.

E.4. Robust inference study

To thoroughly evaluate the robustness of ViTime against TimesFM under various data quality scenarios, we conducted extensive experiments with different types of data perturbations under zero-shot setting. Table 9 presents the comparative

Table 7: Full computational results of zero-shot study. The best zero-shot results are **bolded**, the second best are underlined and the result better than supervised PatchTST are marked with **purple**.

Method		ViTime-1072		ViTime		PatchTST-ZS		TimesFM		PatchTST	
Training data		RealTS		RealTS		RealTS		Real world data		Real world data	
Supervision		Zero-shot		Zero-shot		Zero-shot		Zero-shot		Supervised	
Metric		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
Electricity	96	0.216	0.297	0.219	0.308	1.336	0.896	0.269	0.338	0.121	0.216
	192	0.229	0.308	<u>0.234</u>	<u>0.320</u>	1.425	0.924	0.322	0.375	0.139	0.234
	336	0.244	0.320	<u>0.258</u>	<u>0.338</u>	1.443	0.929	0.381	0.414	0.156	0.251
	720	0.296	0.355	<u>0.344</u>	<u>0.395</u>	1.451	0.932	0.496	0.488	0.194	0.286
Average	-	0.246	0.320	<u>0.263</u>	<u>0.340</u>	1.414	0.920	0.367	0.404	<i>0.153</i>	<i>0.247</i>
Traffic	96	0.665	0.357	0.709	0.394	1.964	0.975	0.771	0.459	0.327	0.242
	192	0.658	0.354	<u>0.723</u>	<u>0.391</u>	2.073	1.008	0.818	0.495	0.342	0.249
	336	0.660	0.357	<u>0.747</u>	<u>0.401</u>	2.071	1.011	0.878	0.524	0.359	0.259
	720	0.725	0.391	<u>0.844</u>	<u>0.447</u>	2.105	1.012	1.033	0.595	0.387	0.274
Average	-	0.677	0.365	<u>0.760</u>	<u>0.408</u>	2.053	1.002	0.875	0.518	<i>0.353</i>	<i>0.256</i>
Weather	96	<u>0.179</u>	<u>0.210</u>	0.172	0.209	0.836	0.561	0.171	0.215	0.146	0.197
	192	<u>0.233</u>	0.258	0.227	0.258	0.922	0.588	0.236	0.273	0.195	0.244
	336	0.285	0.295	<u>0.287</u>	<u>0.300</u>	0.938	0.592	0.303	0.327	0.241	0.281
	720	0.352	0.341	<u>0.368</u>	<u>0.352</u>	0.945	0.594	0.427	0.411	0.307	0.328
Average	-	0.262	0.276	<u>0.263</u>	<u>0.279</u>	0.910	0.584	0.284	0.307	<i>0.222</i>	<i>0.263</i>
ETTh1	96	0.425	<u>0.416</u>	<u>0.432</u>	0.414	1.398	0.877	0.409	0.413	0.365	0.396
	192	0.438	0.428	<u>0.462</u>	<u>0.433</u>	1.490	0.908	0.486	0.458	0.414	0.431
	336	0.464	0.447	<u>0.496</u>	<u>0.457</u>	1.507	0.912	0.535	0.488	0.457	0.461
	720	0.530	0.493	<u>0.574</u>	<u>0.509</u>	1.511	0.914	0.623	0.545	0.550	0.506
Average	-	0.464	0.446	<u>0.490</u>	<u>0.453</u>	1.477	0.903	0.513	0.476	<i>0.447</i>	<i>0.449</i>
ETTh2	96	0.252	0.315	<u>0.258</u>	<u>0.320</u>	1.032	0.752	0.265	0.324	0.290	0.349
	192	0.285	0.346	<u>0.301</u>	<u>0.355</u>	1.115	0.782	0.316	0.362	0.347	0.390
	336	0.351	0.389	<u>0.346</u>	<u>0.390</u>	1.115	0.781	0.365	0.401	0.372	0.414
	720	0.431	0.443	<u>0.447</u>	<u>0.458</u>	1.122	0.783	0.472	0.476	0.451	0.469
Average	-	0.330	0.373	<u>0.338</u>	<u>0.380</u>	1.096	0.775	0.355	0.391	<i>0.365</i>	<i>0.406</i>
ETTh1	96	0.428	<u>0.394</u>	<u>0.416</u>	0.392	1.215	0.773	0.576	0.454	0.296	0.347
	192	0.451	0.411	<u>0.454</u>	<u>0.413</u>	1.309	0.801	0.608	0.479	0.338	0.374
	336	0.467	0.424	<u>0.503</u>	<u>0.437</u>	1.329	0.808	0.705	0.517	0.381	0.398
	720	0.508	0.449	<u>0.588</u>	<u>0.479</u>	1.327	0.807	0.792	0.563	0.421	0.423
Average	-	0.464	0.420	<u>0.490</u>	<u>0.430</u>	1.295	0.797	0.670	0.503	<i>0.359</i>	<i>0.386</i>
ETTh2	96	<u>0.211</u>	0.293	0.203	0.277	0.749	0.593	0.214	<u>0.284</u>	0.174	0.262
	192	0.263	<u>0.330</u>	<u>0.265</u>	0.319	0.823	0.618	0.295	0.330	0.227	0.301
	336	0.309	<u>0.360</u>	<u>0.322</u>	0.356	0.817	0.617	0.368	0.377	0.279	0.336
	720	0.388	<u>0.409</u>	<u>0.400</u>	0.406	0.829	0.621	0.465	0.443	0.348	0.382
Average	-	0.293	<u>0.348</u>	<u>0.297</u>	0.339	0.805	0.612	0.336	0.359	<i>0.257</i>	<i>0.320</i>

Table 8: MAE of different methods in fine-tuning study.

Method		VITIME (FT)		TimesFM (FT)	GPT4TS (FT)	TIME-LLM (FT)	PatchTST	SiMBA	TIMESNET	PatchTST
Data proportion		10%	100%	10%	10%	10%	10%	100%	100%	100%
ETTh1	96	0.397	0.383	0.398	0.485	0.460	0.485	0.395	0.402	0.400
	192	0.414	0.401	0.424	0.524	0.483	0.524	0.424	0.429	0.429
	336	0.427	0.410	0.436	0.550	0.540	0.550	0.443	0.469	0.440
	720	0.460	0.438	0.445	0.610	0.604	0.610	0.469	0.500	0.468
Average	-	0.424	0.408	0.426	0.542	0.522	0.542	0.433	0.450	0.434
ETTh2	96	0.324	0.302	0.356	0.389	0.326	0.389	0.339	0.374	0.337
	192	0.354	0.331	0.400	0.414	0.373	0.414	0.390	0.414	0.382
	336	0.377	0.352	0.428	0.441	0.429	0.441	0.406	0.452	0.384
	720	0.431	0.411	0.457	0.480	0.449	0.480	0.431	0.468	0.422
Average	-	0.372	0.349	0.410	0.431	0.394	0.431	0.393	0.427	0.381
ETTm1	96	0.341	0.333	0.345	0.419	0.388	0.419	0.360	0.375	0.346
	192	0.364	0.353	0.374	0.434	0.416	0.434	0.382	0.387	0.370
	336	0.386	0.373	0.397	0.454	0.426	0.454	0.405	0.411	0.392
	720	0.420	0.410	0.436	0.556	0.476	0.556	0.437	0.450	0.420
Average	-	0.378	0.367	0.388	0.466	0.427	0.466	0.396	0.406	0.382
ETTm2	96	0.260	0.237	0.263	0.274	0.261	0.274	0.263	0.267	0.256
	192	0.293	0.278	0.309	0.317	0.314	0.317	0.306	0.309	0.296
	336	0.325	0.313	0.349	0.353	0.327	0.353	0.343	0.351	0.329
	720	0.382	0.371	0.415	0.427	0.390	0.427	0.399	0.403	0.385
Average	-	0.316	0.300	0.334	0.343	0.323	0.343	0.327	0.332	0.317
Electricity	96	0.231	0.226	Not Reported		0.222	0.235	0.253	0.272	0.222
	192	0.239	0.235			0.240	0.250	0.262	0.289	0.240
	336	0.256	0.249			0.259	0.270	0.277	0.300	0.259
	720	0.289	0.279			0.290	0.315	0.305	0.320	0.290
Average	-	0.252	0.247			0.253	0.268	0.274	0.295	0.253
Traffic	96	0.247	0.237	Not Reported		0.249	0.268	0.268	0.321	0.249
	192	0.247	0.239			0.256	0.274	0.317	0.336	0.256
	336	0.251	0.246			0.264	0.282	0.284	0.336	0.264
	720	0.271	0.279			0.286	0.319	0.297	0.350	0.286
Average	-	0.254	0.250			0.264	0.286	0.291	0.336	0.264
Weather	96	0.188	0.186	Not Reported		0.198	0.221	0.219	0.220	0.198
	192	0.230	0.228			0.241	0.261	0.260	0.261	0.241
	336	0.271	0.270			0.282	0.300	0.297	0.306	0.282
	720	0.317	0.322			0.334	0.351	0.349	0.359	0.334
Average	-	0.252	0.251			0.264	0.283	0.281	0.267	0.264

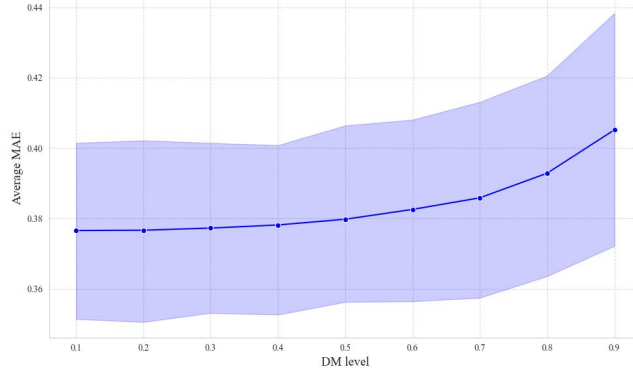


Figure 15: Average MAE of ViTime across different DM rates.

results under Gaussian noise (GN) with standard deviations of 0.1 and 0.3, as well as data missing (DM) with a probability of 0.3. As shown in Table 9, ViTime consistently outperforms TimesFM across all data quality scenarios. ViTime’s resilience to GN is particularly noteworthy, where it maintains superior performance even under higher noise levels (std=0.3). This robust performance can be attributed to the model’s visual representation learning mechanism, which effectively filters out noise through its vision feature extraction process.

Furthermore, we conducted an in-depth analysis of ViTime’s performance under varying degrees of missing data, ranging from 10% to 90% missing values. Figure 15 illustrates the model’s average ReMAE across different DM ratios. The results demonstrate that ViTime maintains remarkable prediction accuracy when the DM ratio remains below 50%, with only gradual performance degradation observed. This robustness can be attributed to the model’s ability to treat missing values as zero-valued pixels in its visual representation, effectively leveraging spatial relationships in the remaining data points to maintain prediction quality.

These comprehensive results further validate the superior generalizability and robustness of ViTime compared to traditional numerical fitting-based approaches, particularly in scenarios with compromised data quality. The model’s ability to maintain consistent performance under various perturbations makes it a promising solution for real-world applications where data quality cannot always be guaranteed.

E.5. Computational complexity analysis

We conduct extensive experiments to analyze the computational complexity and prediction accuracy of our proposed models. All experiments are performed with batch size 4, input sequence length 512, and prediction horizon 720 on a single Nvidia 3090 GPU.

The baseline TimesFM model requires substantial computational resources with 18.1GB GPU memory and 200M parameters, while achieving an average ReMAE of 0.423. In contrast, our proposed ViTime architecture demonstrates remarkable improvements in both efficiency and accuracy. The basic version without the refining module strikes an optimal balance between computational efficiency and performance - it requires only 667MB GPU memory ($27\times$ reduction), achieves faster inference at 0.082s per batch, uses 63% fewer parameters (74M), while maintaining competitive accuracy with an average ReMAE of 0.381.

For applications prioritizing prediction accuracy, the ViTime variant with refining module achieves the best performance with an average ReMAE of 0.376, representing an 11.1% improvement over TimesFM. This comes at the cost of increased computational overhead - 3.1GB GPU memory and 2.89s inference time per batch, though still maintaining a $5.8\times$ reduction in memory compared to TimesFM. Notably, the mapping & inverse mapping between image space and numerical space in ViTime variants consume only 0.0068s, representing 8.3% and 0.24% of the total inference time for the basic and refined versions, respectively.

These results demonstrate that our proposed architecture offers flexible deployment options: the basic version for resource-constrained scenarios requiring good accuracy and computational efficiency, and the refined version for applications where prediction accuracy is paramount. Both variants significantly outperform the baseline in terms of the computation-accuracy

Table 9: Performance comparison of TimesFM and ViTime across different robust inference scenarios

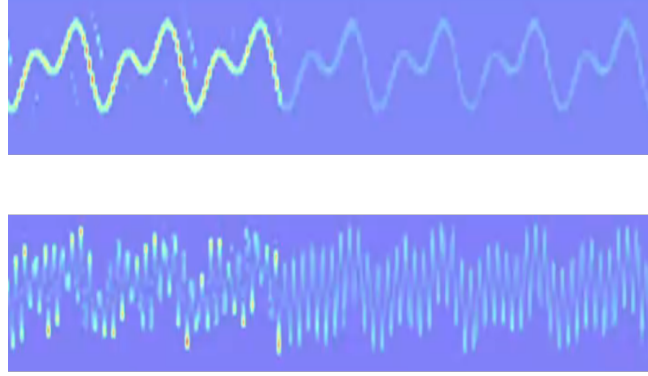
Dataset	PredLen	GN (0.1)				GN (0.3)				DM (0.3)			
		TimesFM		ViTime		TimesFM		ViTime		ViTime		TimesFM	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	-	-
ETTh1	96	0.410	0.414	0.431	0.416	0.427	0.423	0.444	0.436	0.433	0.414	-	-
	192	0.460	0.445	0.460	0.434	0.473	0.452	0.469	0.452	0.463	0.433	-	-
	336	0.513	0.478	0.493	0.458	0.525	0.485	0.501	0.474	0.496	0.456	-	-
	720	0.617	0.545	0.572	0.509	0.632	0.553	0.585	0.525	0.574	0.507	-	-
	Average	0.500	0.471	0.489	0.454	0.514	0.478	0.500	0.472	0.492	0.453	-	-
ETTTh2	96	0.269	0.326	0.256	0.321	0.265	0.329	0.268	0.335	0.255	0.319	-	-
	192	0.326	0.367	0.301	0.356	0.314	0.366	0.308	0.367	0.297	0.352	-	-
	336	0.378	0.406	0.347	0.391	0.366	0.404	0.354	0.401	0.343	0.387	-	-
	720	0.480	0.475	0.449	0.460	0.464	0.470	0.450	0.465	0.441	0.455	-	-
	Average	0.363	0.394	0.338	0.382	0.352	0.392	0.345	0.391	0.334	0.378	-	-
ETTM1	96	0.521	0.444	0.428	0.404	0.532	0.442	0.436	0.425	0.408	0.392	-	-
	192	0.572	0.476	0.468	0.425	0.575	0.469	0.467	0.440	0.451	0.413	-	-
	336	0.642	0.508	0.517	0.450	0.640	0.501	0.509	0.462	0.507	0.440	-	-
	720	0.719	0.551	0.594	0.490	0.688	0.538	0.587	0.501	0.595	0.484	-	-
	Average	0.614	0.495	0.502	0.442	0.609	0.488	0.500	0.457	0.490	0.432	-	-
ETTM2	96	0.212	0.282	0.200	0.277	0.200	0.280	0.209	0.290	0.199	0.274	-	-
	192	0.291	0.329	0.260	0.319	0.262	0.321	0.266	0.327	0.262	0.317	-	-
	336	0.358	0.370	0.322	0.358	0.323	0.360	0.320	0.360	0.320	0.354	-	-
	720	0.459	0.430	0.403	0.407	0.420	0.418	0.392	0.406	0.398	0.403	-	-
	Average	0.330	0.353	0.296	0.340	0.301	0.345	0.297	0.344	0.295	0.337	-	-
Electricity	96	0.266	0.338	0.225	0.317	0.285	0.363	0.252	0.354	0.222	0.311	-	-
	192	0.320	0.375	0.240	0.328	0.333	0.398	0.262	0.361	0.235	0.323	-	-
	336	0.377	0.413	0.263	0.345	0.390	0.440	0.285	0.377	0.260	0.341	-	-
	720	0.488	0.486	0.348	0.402	0.527	0.532	0.368	0.432	0.343	0.397	-	-
	Average	0.363	0.403	0.269	0.348	0.384	0.433	0.292	0.381	0.265	0.343	-	-
Traffic	96	0.770	0.457	0.724	0.398	0.765	0.478	0.778	0.463	0.782	0.406	-	-
	192	0.830	0.489	0.723	0.395	0.815	0.505	0.776	0.461	0.772	0.400	-	-
	336	0.850	0.516	0.743	0.404	0.873	0.534	0.799	0.470	0.792	0.409	-	-
	720	0.979	0.582	0.844	0.450	0.988	0.600	0.896	0.514	0.886	0.454	-	-
	Average	0.857	0.511	0.759	0.412	0.860	0.529	0.812	0.477	0.808	0.417	-	-
Weather	96	0.157	0.202	0.178	0.209	0.170	0.222	0.178	0.230	0.170	0.209	-	-
	192	0.214	0.253	0.229	0.257	0.228	0.272	0.232	0.273	0.231	0.261	-	-
	336	0.272	0.299	0.289	0.300	0.283	0.314	0.284	0.309	0.288	0.302	-	-
	720	0.378	0.369	0.372	0.353	0.380	0.376	0.361	0.355	0.366	0.353	-	-
	Average	0.255	0.281	0.267	0.280	0.265	0.296	0.264	0.292	0.264	0.281	-	-

Table 10: Model Performance and Computational Resource Requirements

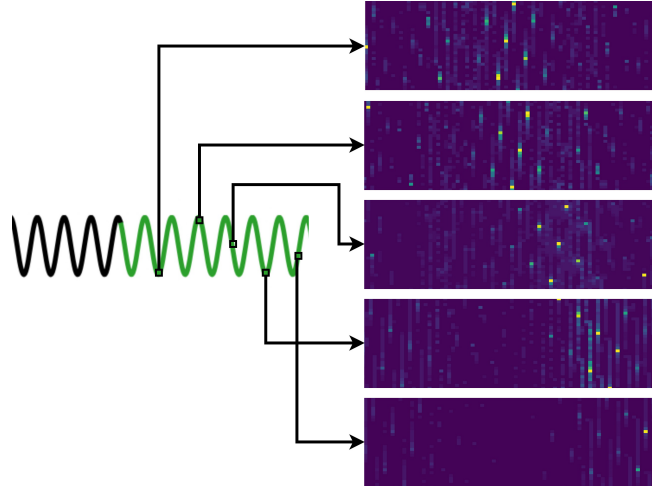
Model	GPU Memory (MB)	Inference Time (s/batch)		Parameters (M)	Avg. ReMAE
		Total	Mapping & Inverse Mapping		
TimesFM	18,154	0.130	-	200	0.423
ViTime w/ Refining Module	3,120	2.890	0.0068	95	0.376
ViTime w/o Refining Module	667	0.082	0.0068	74	0.381

trade-off.

E.6. Interpretation of ViTime



(a) Grad-CAM heatmap showing attention on key trend changes.



(b) Attention maps at different prediction positions demonstrating temporal dependencies.

Figure 16: Visualization of ViTime’s attention mechanism. Despite not using an autoregressive paradigm, ViTime exhibits sequential processing patterns through its multi-layer self-attention modules.

Figure 16 illustrates the attention mechanism of ViTime through grad-cam (Selvaraju et al., 2017) heatmaps and position-specific attention maps. The grad-cam results demonstrate that ViTime focuses strongly on periods of fundamental trend changes. Further analysis through attention maps at different prediction positions reveals an interesting pattern: despite not adopting an autoregressive paradigm, ViTime’s multi-layer self-attention modules process information in a temporal sequence. The input data and the predicted results from previous time steps determine the spatiotemporal distribution of predictions at each time step. This aligns with human cognitive patterns, where information is processed from the recent to the distant past while maintaining awareness of known information.

F. Discussion

F.1. Mechanistic Insights: Temporal Dynamics in ViTime

Despite eschewing autoregressive paradigms, ViTime’s multi-layer self-attention modules inherently maintain temporal ordering through spatiotemporal dependency learning. As shown in Figure 16(b), current predictions’ attention patterns demonstrate systematic dependence on both input data characteristics and previous prediction steps, mirroring human cognitive patterns in time series reasoning.

F.2. Why Using ReMSE and ReMAE as metrics?

The proposed ReMSE and ReMAE metrics address a critical challenge in evaluating time series foundation models: mitigating test set leakage caused by overlapping data distributions between training and testing phases. By rescaling the test set across multiple resolutions ($\beta \in \mathbf{U}$) via time series interpolation (TSI, Equation (11)), these metrics introduce synthetic scale variations that disrupt exact temporal patterns, thereby reducing the risk of evaluating models on memorized or overfitted data. This approach ensures a leakage-resistant evaluation framework, as models must generalize to unseen scales rather than relying on spurious correlations learned from the training set.

The empirical results in Figure 4 highlight the efficacy of this strategy. TimesFM, while achieving low error at $\beta = 1$, exhibits significant performance degradation at $\beta \neq 1$, suggesting its predictions at the original scale may exploit leaked patterns. In contrast, ViTime maintains consistent accuracy across all β values, indicating robust learning of intrinsic temporal dynamics rather than scale-specific memorization. This stark divergence underscores how ReMSE/ReMAE expose leakage by penalizing models that overfit to the training data’s implicit resolution.

A key implication of this work is the necessity of scale-agnostic evaluation in time series forecasting. Traditional single-scale metrics like MSE/MAE risk conflating memorization with true generalization, particularly when training data encompasses diverse real-world sources. By averaging errors across β , ReMSE/ReMAE incentivize models to capture invariant temporal structures—such as periodicity, trends, and noise resilience—that persist across resolutions. This aligns with recent theoretical insights in self-supervised learning, where augmentation-induced invariance improves out-of-distribution robustness (Yao et al., 2022).

The proposed ReMSE and ReMAE establish multi-scale evaluation as a paradigm for leakage-free benchmarking of time series models. The success of ViTime under ReMSE/ReMAE suggests that architectural inductive biases for scale invariance—such as decoupled frequency-time processing (Wu et al., 2022)—are critical for developing generalizable foundation models. This aligns with broader machine learning principles where invariance to augmentations correlates with robustness (Yao et al., 2022). Extending this framework to multi-variate and non-stationary settings remains an impactful direction for future research.

F.3. Limitations and Future Directions

While ViTime demonstrates state-of-the-art performance in accuracy and robustness, two key challenges warrant further investigation:

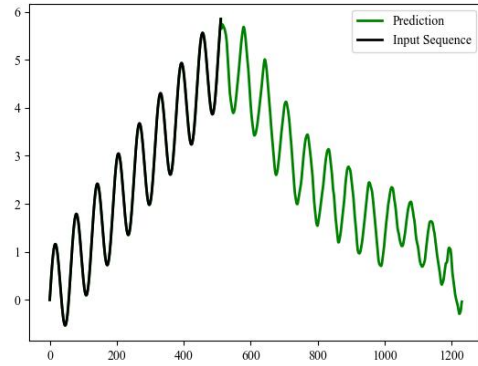
F.3.1. RESOLUTION CONSTRAINTS & ADAPTIVE ENHANCEMENT.

The mapping function’s truncation imposes resolution limits, particularly evident in explosive growth patterns (Figure 17 a-b). A key limitation of ViTime arises from its assumption of $\mathcal{S} \sim \mathcal{N}(0, \mathbf{I})$, which fails to capture the high-variance nature of explosive growth data that typically follows $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$ with $k \gg 1$. As shown in Proposition C.2, the optimal threshold MS^* scales as \sqrt{k} , implying that fixed thresholds (e.g., $MS = 3.5$ for $k = 1.5$) become suboptimal for high-variance scenarios, introducing significant system errors and degrading prediction accuracy.

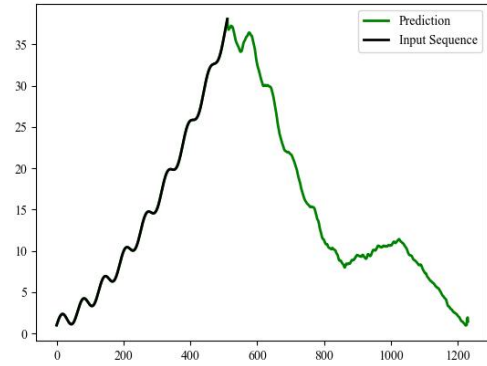
Our empirical analysis reveals that doubling the MS parameter from 3.5 to 7 significantly improves prediction fidelity for explosive growth patterns (Figure 17c-d). However, excessively large MS values increase spectral energy, as demonstrated in Theorem 3.3, leading to computational inefficiency. This trade-off suggests two complementary research directions:

- **Elastic Resolution Enhancement:** Techniques to dynamically adjust spatial resolution h based on data variance, ensuring sufficient granularity for high-variance regions without unnecessary computational overhead.
- **Adaptive MS Estimation:** Algorithms to estimate the variance scaling factor k and compute the optimal MS^* in real-time, balancing prediction fidelity with spectral efficiency.

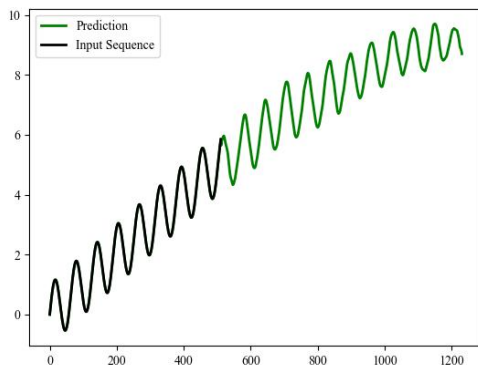
These enhancements would enable ViTime to handle explosive growth patterns more effectively while maintaining computational tractability.



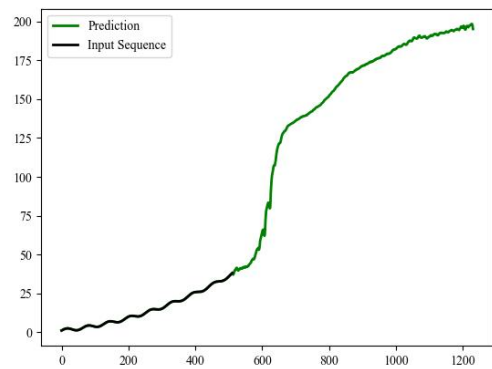
(a) MS=3.5 (Ground Truth)



(b) MS=3.5 (Prediction)



(c) MS=7 (Ground Truth)

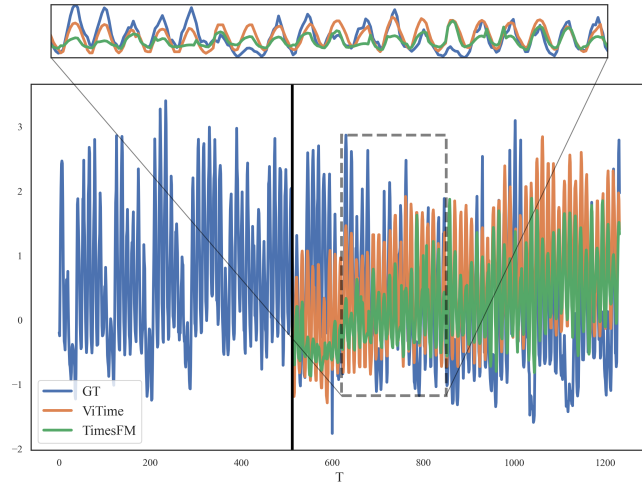


(d) MS=7 (Prediction)

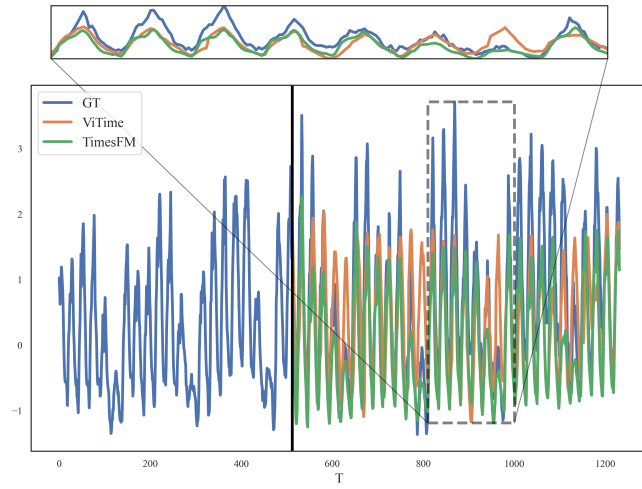
Figure 17: Resolution analysis for explosive growth patterns: (a-b) With MS=3.5, ViTime incorrectly predicts peak decline due to spatial constraints. (c-d) Doubling MS to 7 enables accurate growth trend capture. Comparison demonstrates the critical role of dynamic resolution adjustment.

F.3.2. ENHANCED DATA GENERATION.

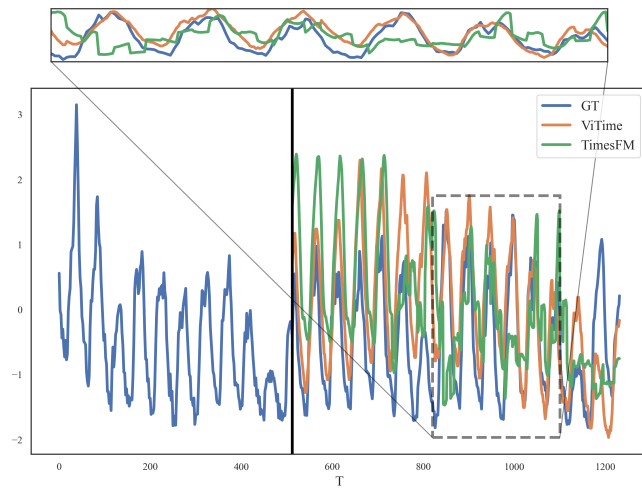
ViTime’s predictive quality fundamentally depends on RealTS’s synthetic data generation capabilities. The current methodology faces challenges in simulating complex real-world temporal dynamics, particularly for non-stationary processes and regime-switching scenarios. Future work should develop 1) Advanced pattern injection mechanisms for synthetic data generation, and 2) Quantitative metrics for simulation fidelity assessment across different temporal regimes.



(a) Rescale factor=0.5

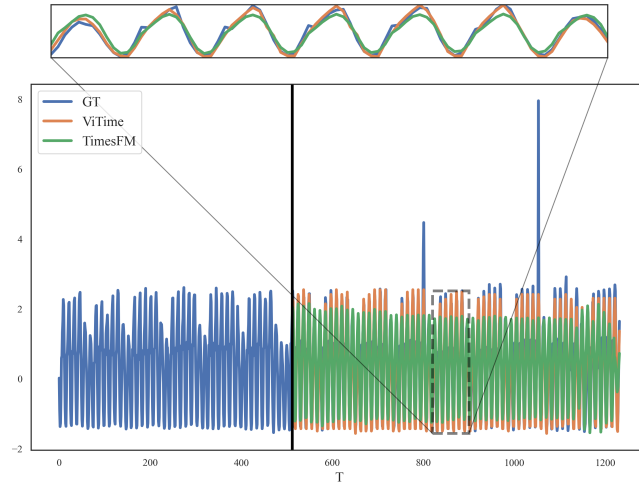


(b) Rescale factor=1

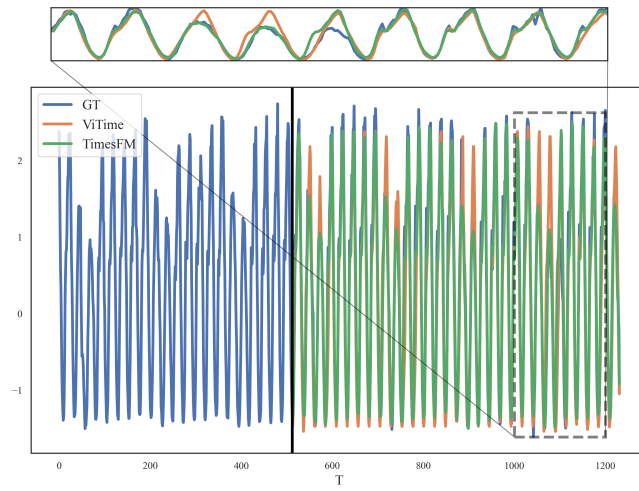


(c) Rescale factor=2

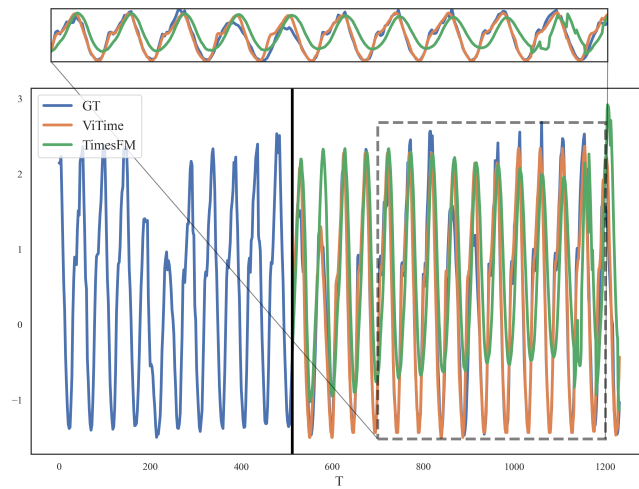
Figure 18: Illustrative example of Electricity dataset.



(a) Rescale factor=0.5

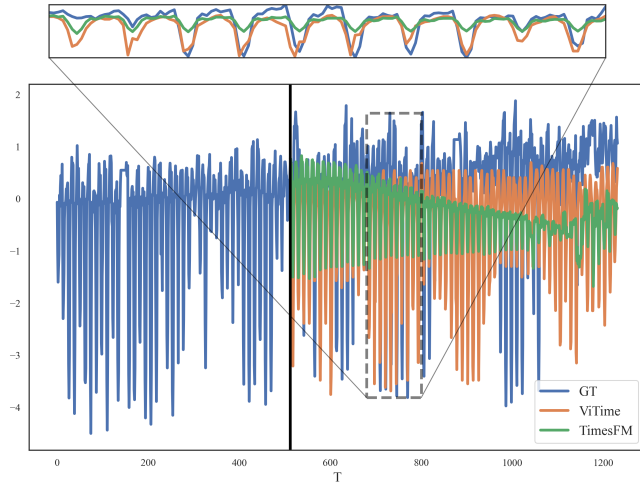


(b) Rescale factor=1

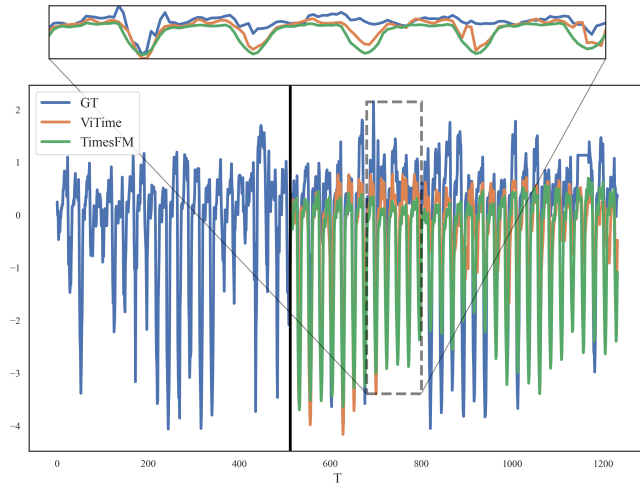


(c) Rescale factor=2

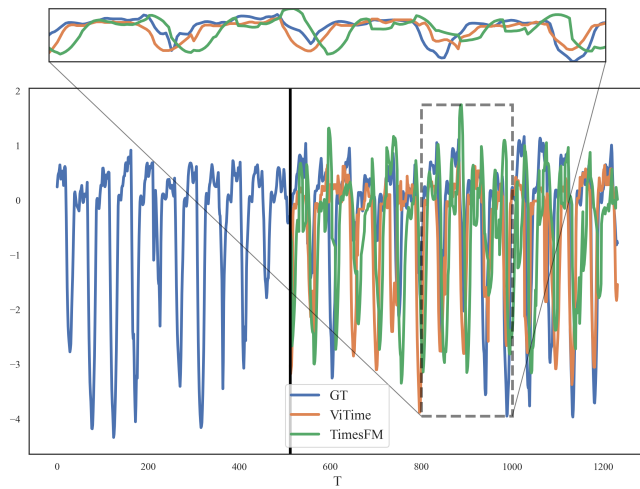
Figure 19: Illustrative example of Traffic dataset.



(a) Rescale factor=0.5

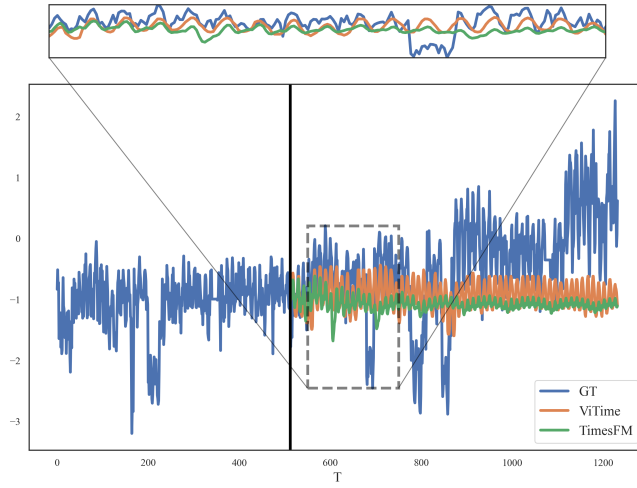


(b) Rescale factor=1

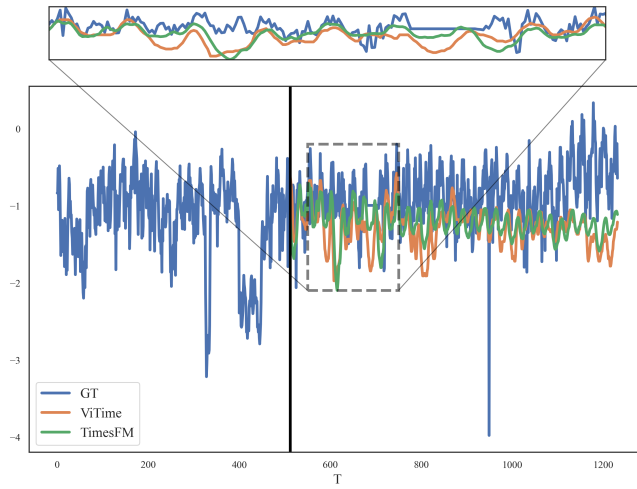


(c) Rescale factor=2

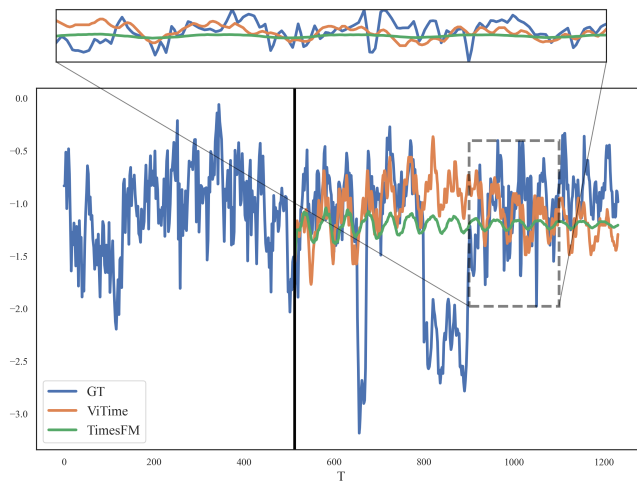
Figure 20: Illustrative example of ETTh1 dataset.



(a) Rescale factor=0.5

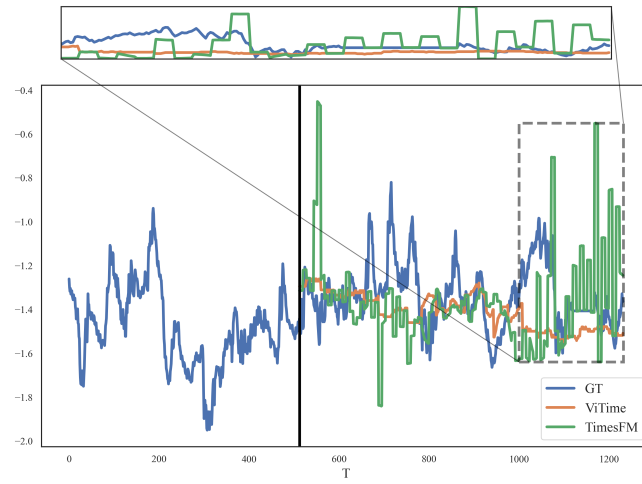


(b) Rescale factor=1

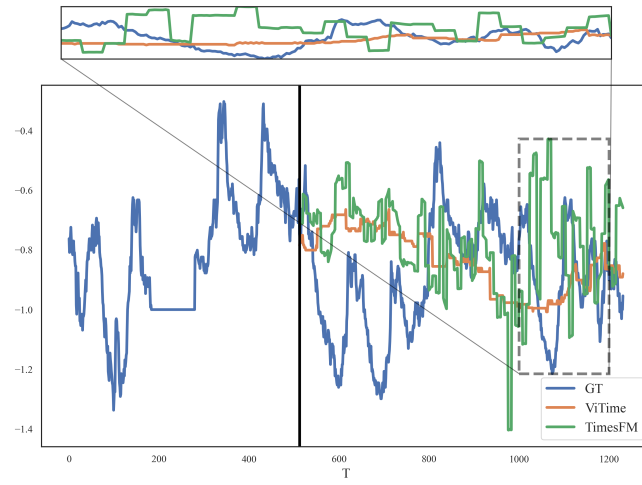


(c) Rescale factor=2

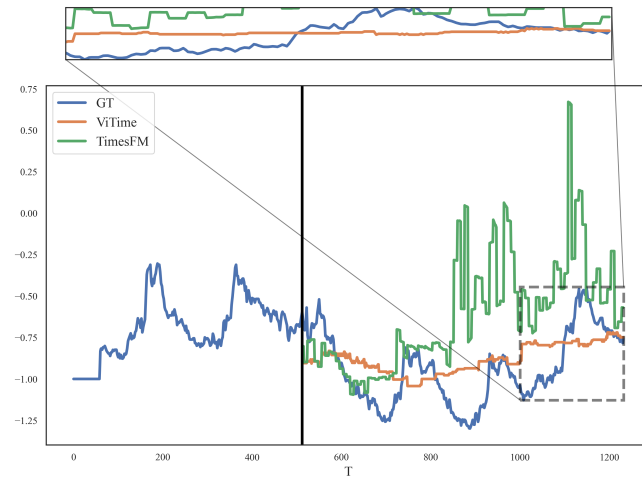
Figure 21: Illustrative example of ETTh2 dataset.



(a) Rescale factor=0.5

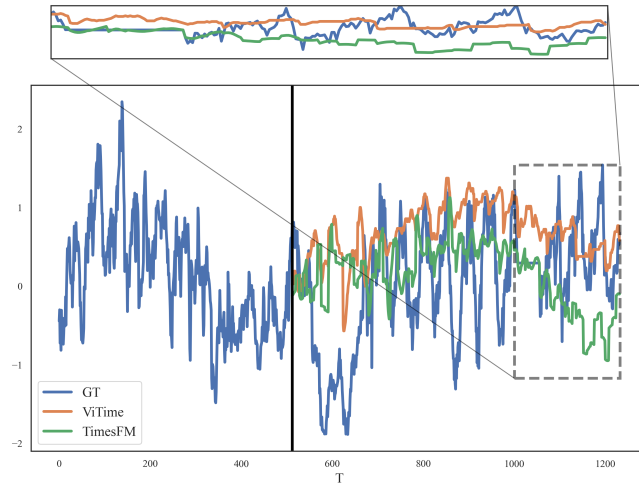


(b) Rescale factor=1

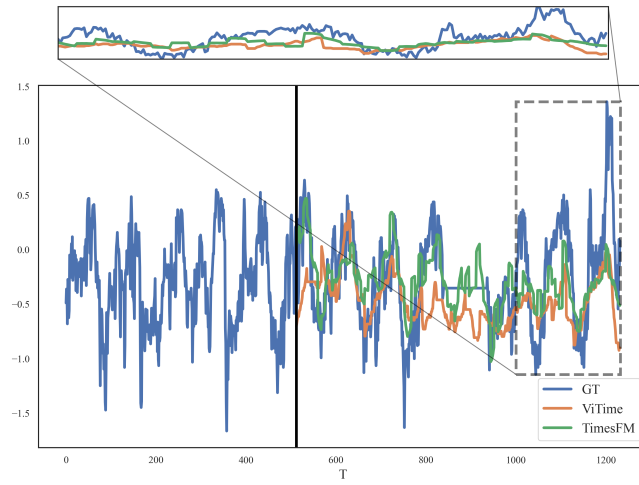


(c) Rescale factor=2

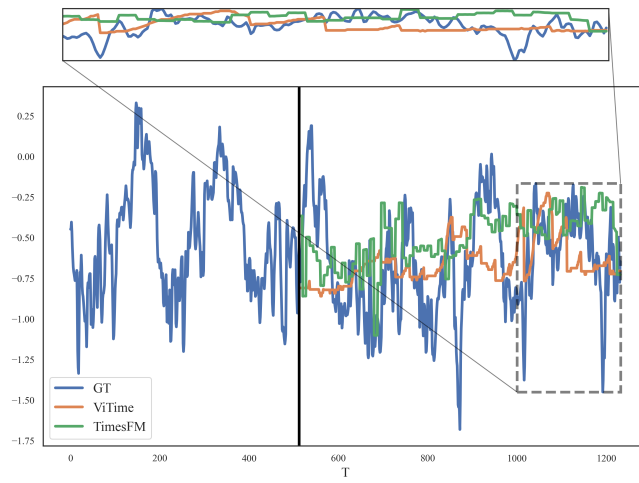
Figure 22: Illustrative example of ETTm1 dataset.



(a) Rescale factor=0.5



(b) Rescale factor=1



(c) Rescale factor=2

Figure 23: Illustrative example of ETTm2 dataset.