arXiv:2509.25826v3 [cs.LG] 14 May 2026

KAIROS: Toward Adaptive and Parameter-Efficient Time Series Foundation Models

Kun Feng[1*], **Shaocheng Lan**[1*], **Yuchen Fang**[2*], **Wenchao He**[1],
Sihan Lu[1], **Shuqi Gu**[1], **Lintao Ma**[2], **Xingyu Lu**[2], **Kan Ren**[1†]
[1]School of Information Science and Technology, ShanghaiTech University, Shanghai, China
[2]Ant Group, Shanghai, China

Abstract

Inherent temporal heterogeneity, such as varying sampling densities and periodic structures, has posed substantial challenges in zero-shot generalization for Time Series Foundation Models (TSFMs). Existing TSFMs predominantly rely on massive parameterization to absorb such heterogeneity, as their static tokenization and positional encoding schemes entangle diverse temporal patterns into a fixed representation space, encouraging memorization rather than adaptation. To address this limitation, we propose KAIROS, a flexible and parameter-efficient TSFM dedicated to forecasting tasks, which decouples temporal heterogeneity from model capacity through a novel tokenization perspective. KAIROS introduces a dynamic patching tokenizer and a mixture-of-size encoding that adapt observational granularity to local information density, enabling fine-grained temporal abstraction without increasing model width or depth. In addition, we design a multi-granularity positional embedding based on dynamic rotary encodings, which conditions on instance-level spectral features and temporal structure induced by dynamic patching tokenization, allowing robust modeling of diverse temporal dependencies. Trained on a novel Predictability-Stratified Time-Series (PreSTS) corpus, KAIROS achieves superior zero-shot performance with substantially fewer parameters on two mainstream benchmarks, GIFT-Eval and Time-Series-Library. The project page is at https://foundation-model-research.github.io/Kairos.

1 Introduction

Time series forecasting is a core component of many real-world applications [6, 39]. Although time series data are ubiquitous, many individual datasets suffer from scarcity, making it difficult to train models independently, particularly for deep neural networks that require large amounts of data. This challenge has long motivated the development of few-shot and zero-shot forecasting methods [30]. More recently, inspired by the success of large-scale foundation models [1, 21], researchers have turned to building Time Series Foundation Models (TSFMs) for forecasting. These models leverage vast and diverse time series corpora to acquire strong generalization capabilities, showing promising results across various downstream tasks [4, 40, 34].

Despite this progress, a key limitation persists: current TSFMs for forecasting rely on non-adaptive processing pipelines that fail to reflect the dynamic and heterogeneous nature of time series, severely compromising their parameter efficiency. As shown in Figure 1(b)(c), the statistical information density (details in Appendix K) varies not only across datasets but also across segments within the same series. By applying rigid architectures (exemplified in Figure 1(d)) to this inherently heterogeneous data, models are forced to use massive parameters to memorize specific patterns rather

*Equal contribution.
†Corresponding author: renkan@shanghaitech.edu.cn

Preprint.

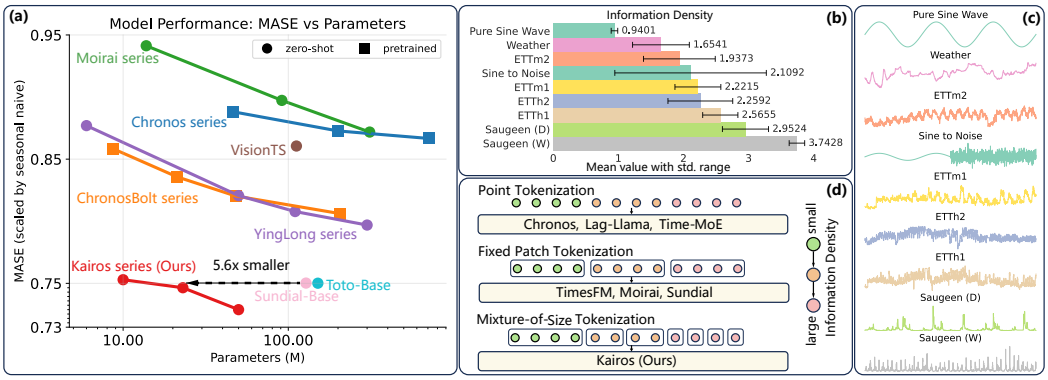


Figure 1: **(a)** Comparison on the GIFT-Eval benchmark [3] demonstrates that our KAIROS achieves superior zero-shot forecasting performance (lower normalized MASE) while requiring significantly fewer parameters than existing TSFMs. **(b) (c)** Significant variation exists in information density (i.e., signal complexity) across and within different time series datasets. **(d)** Existing TSFMs primarily use tokenization methods like point-wise or fixed-size patching, while our KAIROS uses Mixture-of-Size Tokenization to dynamically adapt to information density, enabling the learning of generalizable rules rather than memorization.

than acquiring adaptive, generalizable rules. Unfortunately, this strategy diminishes the overall utility of the model's capacity, rendering it less effective for real-world forecasting applications.

This deficiency manifests primarily in two fundamental aspects of model design: (i) *Representation learning*: TSFMs have yet to adopt dynamic *tokenization strategies* analogous to those in Large Language Models (LLMs). While LLMs utilize subword tokenization [14] to efficiently compress semantic units of varying lengths, TSFMs predominantly persist with static patching, whether through point-wise tokenization [4, 34] or fixed-size patches [11]. By coupling token generation to fixed time intervals rather than information content, these approaches compress diverse temporal variations into a rigid representation space. Consequently, the model wastes computational capacity on low-information segments while struggling to capture high-frequency details within the same fixed budget, leading to significant parameter inefficiency. (ii) *Temporal dependency modeling*: Conventional *positional encodings* in TSFMs impose a unified temporal scale across sequences, neglecting differences in periodicity, trend and seasonality. As a result, existing TSFMs struggle to adapt to heterogeneous domains, for example, power consumption data measured hourly versus retail sales data measured daily, each exhibiting distinct temporal dynamics, as we validate in Section 4.3. Consequently, the model requires a deeper, more complex architecture to compensate for this misalignment, further exacerbating parameter inefficiency and hindering practical utility.

To address these limitations, we propose KAIROS, a parameter-efficient model comprising a Mixture-of-Size Encoder, a Heterogeneity-Aware Transformer, and a Multi-Patch Decoder. KAIROS addresses time series heterogeneity through two primary mechanisms: (i) For intra-patch representation learning, the Mixture-of-Size Encoder adaptively models time series at multiple granularities based on local characteristics. Inspired by null experts in Mixture-of-Experts [43, 20], we introduce computation-free null experts to dynamically adjust the number of active granularities. This approach moves beyond static tokenization strategies, enabling more expressive modeling of nuanced, time-varying dynamics. (ii) For inter-patch dependency modeling, we enhance the Transformer with Dynamic Rotary Position Embedding (DRoPE). Unlike standard RoPE [35], which represents positional information with a fixed temporal scale, DRoPE modulates temporal scales using spectral features and granularity to generate instance-specific positional encodings. Flexible intra-patch and inter-patch modeling enable KAIROS to effectively characterize heterogeneous time series with a substantially reduced parameter count (see Figure 1(a)). Additionally, the Multi-Patch Decoder uses learnable tokens to predict future patches in parallel, which mitigates cumulative errors in autoregressive generation and offers greater flexibility for variable-length prediction horizons.

To complement our architecture with high-quality supervision, we construct the Predictability-Stratified Time Series (PreSTS) corpus, a large-scale and diverse pretraining dataset curated through a targeted sampling strategy. By prioritizing more predictable sequences while maintaining broad cover-

age, PreSTS provides high-quality supervision that supports efficient model scaling. As demonstrated by the results on the GIFT-Eval benchmark [3] in Figure 1(a), these combined innovations allow our KAIROS to achieve superior performance while using fewer parameters. This strong performance is consistently observed on Time-Series-Library (TSLib) [39] benchmarks as well.

In summary, our main contributions are as follows: (i) We present KAIROS, a parameter-efficient time series foundation model that introduces a novel architectural framework to handle the dynamic and heterogeneous nature of time series data. (ii) We curate the PreSTS corpus, a large-scale pre-training dataset of over 300 billion time points, featuring a predictability-based sampling strategy to provide high-quality and diverse supervision. (iii) Extensive empirical evaluations on the GIFT-Eval and Time-Series-Library benchmarks confirm the efficacy of KAIROS, highlighting its superior zero-shot forecasting performance.

2 Related Work

Time Series Foundation Models (TSFMs) for Forecasting. Inspired by the strong generalization of Large Language Models (LLMs), researchers have devoted increasing attention to TSFMs. Existing approaches typically involve adapting LLMs to time series [18], training scalable Transformers on large corpora [4, 11, 40, 10, 28], or exploring alternative architectures, including non-Transformer models [5, 17] and lightweight Multi-Layer Perceptrons [13]. However, these models often apply fixed tokenization strategies and position embedding to time series from diverse domains with distinct characteristics, limiting their cross-domain generalization. KAIROS, in contrast, employs a dynamic patching tokenizer and instance-adaptive, granularity-aware position embedding to enhance its generalizability for diverse time series data.

Time Series Tokenization. Transformer-based models require converting raw data into discrete or continuous tokens [32, 2]. Current strategies generally fall into two static paradigms: (i) *Fixed-size*: The dominant paradigm relies on a uniform patch size across the sequence. This ranges from fine-grained point-wise tokenization [4, 31, 34] to coarser uni-size patching [29], which has been adopted by recent foundation models [40, 28]. While the latter improves upon the computational inefficiency of point-wise methods, both remain rigid. (ii) *Multi-size*: To capture broader temporal dynamics, models like Pathformer [8] and TTM [13] incorporate multiple predefined patch sizes. However, these approaches apply selected granularities uniformly across the entire sequence based on global features rather than local information. This rigidity forces the model to allocate equal computational cost to segments of varying complexity. This imbalance leads to suboptimal parameter efficiency and limited generalization. In contrast, KAIROS employs a mixture-of-size strategy. By allocating fewer tokens to simple patterns and more to complex regions, KAIROS enhances parameter efficiency while maintaining robust generalization.

Position Embedding. Due to the position-unaware characteristic of self-attention mechanism, Transformers [36] rely on position embeddings (PEs) to model temporal information. However, most TSFMs adopt existing designs for PE in Natural Language Processing (NLP) domains [11, 34], which typically emphasize long-term decay through methods like sinusoidal calculations or by suppressing long-range positional information [36, 35]. By imposing a uniform temporal scale, these PEs struggle to model the heterogeneous temporal relationships of time series. Existing adaptive PE methods in NLP [45, 23] are also insufficient, as they target context extrapolation rather than the complex temporal structures of time series. Therefore, ElasTST [44] proposed a tunable RoPE to better adapt to time series data. However, its adaptation uses per-dataset training rather than dynamic adjustment. To address this limitation, we propose to dynamically modulate PE tailored to the intrinsic characteristics of each input time series, effectively decoupling temporal heterogeneity from model capacity, therefore enhancing the parameter efficiency and overall performance of TSFMs.

3 Methodology

3.1 Overall Framework

We formulate our task as learning a time series forecasting function $f_\phi : \mathbb{R}^T \to \mathbb{R}^H$ parameterized by ϕ. Given historical time series $\mathbf{X} \in \mathbb{R}^T$, time series forecasting is to predict the future horizon

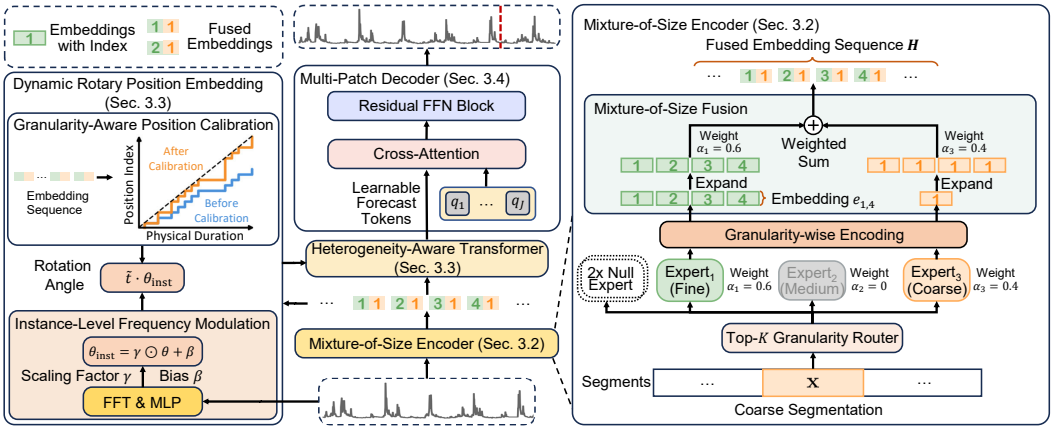


Figure 2: The architecture of KAIROS, which includes (i) The Mixture-of-Size Encoder (right) adaptively tokenizes the input series by routing segments to optimal granularity experts and fusing their embeddings into a unified sequence $\boldsymbol{H}$. (ii) The Heterogeneity-Aware Transformer (middle) processes these tokens using Dynamic Rotary Position Embedding (DRoPE) (left), which modulates frequencies based on instance-level spectral features and calibrates positions to account for varying patch sizes. (iii) The Multi-Patch Decoder (top-middle) employs learnable forecast tokens to predict multiple future patches in parallel via cross-attention.

$\mathbf{Y} \in \mathbb{R}^H$. The learning objective is to minimize the discrepancy between the ground truth $\mathbf{Y}$ and the prediction $\hat{\mathbf{Y}} = f_\phi(\mathbf{X})$. We detail it in Appendix H.1.2.

The architecture of KAIROS, shown in Figure 2, consists of three components. First, the input time series is tokenized by the Mixture-of-Size Encoder to extract multi-granularity local representations via dynamic granularity selection. These embeddings are then processed by a Heterogeneity-Aware Transformer equipped with Dynamic Rotary Position Embedding (DRoPE), a granularity-aware positional encoding that performs instance-wise modulation and adapts to dynamic patching tokenization. Finally, the Multi-Patch Decoder generates forecasts by predicting multiple future patches in parallel, mitigating cumulative errors. The details of these components are presented in the following sections.

3.2 Mixture-of-Size Encoder

Prior to the Heterogeneity-Aware Transformer (Section 3.3), the Mixture-of-Size Encoder adaptively tokenizes the raw time series $\mathbf{X}$ through a three-stage process. First, the series is partitioned into fixed-size *segments*, where a router dynamically selects optimal temporal granularities *for each segment*. Next, for each activated granularity, the segment is simultaneously patchified and projected into the latent space by the expert network corresponding to that granularity. Finally, these multi-granularity embeddings are fused into a unified token sequence representing the input series.

Notably, the Mixture-of-Size Encoder is both *sparse* and *dynamic*. A gating mechanism equipped with null experts (described below) adaptively selects relevant granularities *for each segment*, enabling sparse expert activation and segment-specific granularity adaptation.

Coarse Segmentation. As shown in Figure 1(b) in Section 1 where time series exhibits inherent temporal heterogeneity, we first apply coarse segmentation to the input time series $\mathbf{X}$ and process each segment independently in subsequent tokenization and encoding stages. Specifically, $\mathbf{X}$ is partitioned into a sequence of non-overlapping coarse segments $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, where each segment $\mathbf{x}$ has a fixed length P and the total number of segments is $N = \lceil T/P \rceil$. In the following, we operate within each segment $\mathbf{x}$ and therefore omit the segment index for notational simplicity.

Granularity. For a time series segment $\mathbf{x}$, we adopt a multi-granularity perspective. *Granularity* refers to the temporal resolution at which the segment is tokenized and encoded, as defined below.

Definition 3.1 (Granularity Level)**.** *A granularity level $i \in \{1, \ldots, G\}$ is defined by a specific patch size P_i and its corresponding encoding network π_i, under which the time series segment $\mathbf{x}$ is patchified and mapped into a latent representation.*

Intuitively, finer granularity levels capture high-frequency local variations, while coarser granularity levels emphasize low-frequency global trends. At each granularity level i, the patchification and embedding operations described below are applied independently and in parallel, producing representations that characterize the segment at the corresponding temporal resolution.

Step 1: Top-K Granularity Routing. Applying encoding at an appropriate granularity level for each time series segment is non-trivial due to the inherent temporal heterogeneity of time series data. To address this challenge, we introduce a dynamic routing mechanism equipped with a lightweight *router* that adaptively selects an optimal subset of granularity levels for subsequent encoding of a given segment. Inspired by the Mixture-of-Experts (MoE) paradigm [19, 33], we instantiate G granularity-specific experts, each corresponding to a distinct granularity level, together with $Z \in \{1, \ldots, K-1\}$ computation-free null experts. The inclusion of null experts allows the model to implicitly skip encoding at certain granularities, enabling adaptive control over the *effective number of active granularities for each input segment*. For each expert $i \in \{1, \ldots, G+Z\}$, the routing weights are computed via a learnable affinity score $s_i = \mathbf{w}_i^\top \mathbf{x}$, normalized with a bias-corrected softmax $\tilde{s}_i = (\exp(s_i + \boldsymbol{b}_i)) \,/\, \left(\sum_{j=1}^{G+Z} \exp(s_j + \boldsymbol{b}_j)\right)$, where $\mathbf{w}_i$ denotes learnable parameters and $\boldsymbol{b}_i$ is a bias for auxiliary-loss-free load balancing [24] (details in Appendix H.3.1). The router then activates the indices $\mathcal{I}_{\text{valid}} = \text{argtop-K}_{i \in [G+Z]}\ \tilde{s}_i \cap \{1, \ldots, G\}$. By allowing null experts to dynamically prune the number of active experts, KAIROS dynamically adapts the effective resolution to local information while maintaining a sparse, parameter-efficient architecture.

Step 2: Granularity-wise Encoding. For each selectively activated granularity expert at level i, we apply the corresponding patching and embedding operations. Specifically, a given time series *segment* $\mathbf{x}$ is re-partitioned into a sequence of patches at granularity i:

$$[\boldsymbol{p}_{i,1}, \ldots, \boldsymbol{p}_{i,M_i}] = \text{Patchify}(\mathbf{x}, i) \in \mathbb{R}^{M_i \times P_i}, \tag{1}$$

where P_i denotes the patch size at granularity i, and $M_i = \lceil P/P_i \rceil$ is the resulting number of patches obtained via non-overlapping segmentation, with zero-padding applied when necessary. Each patch is then projected into the latent space to obtain an embedding sequence $\boldsymbol{E}_i = [\boldsymbol{e}_{i,1}, \ldots, \boldsymbol{e}_{i,M_i}]$:

$$\boldsymbol{e}_{i,m} = \pi_i(\boldsymbol{p}_{i,m}) \in \mathbb{R}^{D_h}, m \in [1, M_i]\ . \tag{2}$$

where π_i denotes a granularity-specific projection function, implemented as a two-layer MLP.

Step 3: Mixture-of-Size Fusion. In the final encoder stage, we fuse the embeddings produced by the activated granularity experts into a unified token sequence, serving as the input to the Heterogeneity-Aware Transformer. First, for a time-series segment $\mathbf{x}$, we align all granularity-specific embeddings $\boldsymbol{E}_i$ to the finest activated resolution $P_{\min} = \min\{P_i \mid i \in \mathcal{I}_{\text{valid}}\}$, yielding a target sequence length $M = \lceil P/P_{\min} \rceil$. This alignment is crucial to eliminate representation bias arising from unequal sequence lengths induced by different patch sizes across granularities. To enable efficient alignment, we design the patch sizes to be nested, i.e., P_{i+1} is divisible by P_i. Under this constraint, each embedding sequence $\boldsymbol{E}_i \in \mathbb{R}^{M_i \times D_h}$ can be deterministically expanded to the target resolution via repetition (details and an example in Appendix H.3.2): $\tilde{\boldsymbol{E}}_i = \text{Expand}(\boldsymbol{E}_i) \in \mathbb{R}^{M \times D_h}$. The fused representation is obtained by a gated weighted summation over the activated granularities: $\bar{\boldsymbol{E}} = \sum_{i \in \mathcal{I}_{\text{valid}}} c_i\, \tilde{\boldsymbol{E}}_i, c_i = \tilde{s}_i / \sum_{j \in \mathcal{I}_{\text{valid}}} \tilde{s}_j$.

Sample-level Result. Finally, the representations from all coarse segments are concatenated to form the encoder output

$$\boldsymbol{H} = \text{Concat}(\bar{\boldsymbol{E}}_1, \ldots, \bar{\boldsymbol{E}}_N) \in \mathbb{R}^{\tilde{T} \times D_h}, \tag{3}$$

where $\tilde{T} = \sum_{n=1}^{N} M_n$ and M_n is the embedding sequence length of the n-th segment, forming the representation sequence for the given time series sample $\mathbf{X}$.

Unlike prior approaches such as Chen et al. [8], Zhang et al. [44], which impose a *static* multi-scale partitioning over the entire time series, our method (i) dynamically adapts the tokenization granularity at the *segment level*. (ii) decouples temporal heterogeneity from a fixed representational scale by

explicitly and dynamically assigning patch size via routing and multi-granularity encoding, enabling more effective *intra-patch representation learning*. This adaptive behavior cannot be achieved via globally predefined scales applied statically and is empirically validated in Section 4.4.1 and Appendix E. As a result, varying temporal patterns are modeled via the lightweight Mixture-of-Size Encoder, reducing the burden of subsequent model parameterization.

3.3 Heterogeneity-Aware Transformer

Following Mixture-of-Size Encoder, the fused representation $\boldsymbol{H} \in \mathbb{R}^{\tilde{T} \times D_h}$ is processed by a Heterogeneity-Aware Transformer, a stack of standard Transformer [36] encoder blocks, yielding representations $\boldsymbol{H}^{\text{en}} \in \mathbb{R}^{\tilde{T} \times D_h}$ that are subsequently fed to the prediction head. To inject temporal information into the self-attention mechanism, we propose Dynamic Rotary Position Embedding (DRoPE). Unlike static encodings such as Vaswani et al. [36], Su et al. [35], DRoPE adaptively calibrates temporal structure to account for instance-specific periodicities and the varying physical durations of dynamic patches.

Base Structure of DRoPE. Our dynamic position encoding is based on the standard Rotary Position Embedding (RoPE) [35], where a hidden vector $\boldsymbol{z} \in \mathbb{R}^{D_h}$ (a query or key) at token index $t \in [1, \tilde{T}]$ is treated as a set of $D_h/2$ complex numbers. Each pair is rotated by an angle proportional to the position t and a fixed frequency θ_d as below, where $d \in [0, D_h/2 - 1]$ indexes the dimension:

$$f_{\text{DRoPE}}(\boldsymbol{z}, t) = (\boldsymbol{z}_{2d} + i\boldsymbol{z}_{2d+1})e^{it\theta_d} \tag{4}$$

Here the frequencies are traditionally pre-defined as $\theta_d = b^{-2d/D_h}$. DRoPE generalizes this formulation by dynamically transforming both the frequency θ and the token position t. A detailed derivation of the RoPE calculation is provided in Appendix H.4.

Instance-Level Frequency Modulation. Standard RoPE uses a uniform frequency θ for all sequences, failing to capture the unique periodic structures of heterogeneous time series (analysis in Theorem H.1). We propose to modulate the base frequencies using instance-specific spectral features. Specifically, given a time series input $\mathbf{X}$, we extract the magnitudes of the first ω low-frequency components via Fast Fourier Transform, denoted as $\mathbf{X}_{\text{fft}}$. A lightweight MLP maps these features to a scaling factor γ and a bias β to produce *adaptive frequencies* θ_{inst}. To accommodate the exponentially decaying nature of RoPE's base frequencies, we perform this affine transformation in log-space:

$$\log \theta_{\text{inst},d} = \gamma_d \cdot \log \theta_d + \beta_d \text{ , where } \gamma, \beta = \text{MLP}(\mathbf{X}_{\text{fft}}) \tag{5}$$

This log-space adaptation allows KAIROS to robustly re-scale the temporal rotation based on the dominant information like trends and seasonalities in the specific time series instance (see Appendix H.4.2 for a detailed discussion).

Granularity-Aware Position Calibration. In our multi-granularity framework, the discrete token index t is no longer an accurate proxy for time because different tokens represent different physical durations, a direct consequence of the granularity routing and tokenization in our Mixture-of-Size Encoder, as described in Section 3.2. To maintain temporal consistency across mixed patch sizes, we replace the index t with a calibrated position $\tilde{t} = \sum_{k=1}^{t-1} \bar{P}_k / \bar{P}_{\text{g\_min}}$, where $\bar{P}_k$ is the patch size of the k-th token and $\bar{P}_{\text{g\_min}} = \min\{P_i\}_{i=1}^{G}$ is the globally minimum patch size of the finest granularity.

Unifying these two components, rotating the hidden vector $\boldsymbol{z}$ at position t as follows:

$$f_{\text{DRoPE}}(\boldsymbol{z}, t) = (\boldsymbol{z}_{2d} + i\boldsymbol{z}_{2d+1})e^{i \cdot \tilde{t} \cdot \theta_{\text{inst},d}} \tag{6}$$

By measuring relative distances in terms of absolute physical time and instance-specific frequencies, DRoPE enables the Transformer to learn consistent temporal dependencies considering the diverse encoding granularity and the intrinsic properties of the data sample.

3.4 Multi-Patch Decoder

To mitigate the error accumulation typical of patch-wise autoregressive decoding while maintaining structural flexibility, KAIROS utilizes a Multi-Patch Decoder. This component forecasts J future patches in parallel, effectively decoupling the prediction horizon from iterative dependencies. We

first introduce a sequence of J learnable forecast tokens, $\mathbf{Q}_{\text{forecast}} = \{\mathbf{q}_1, \ldots, \mathbf{q}_J\}$. Multi-Patch Decoder processes these token embeddings with the latent representations $\boldsymbol{H}^{\text{en}} \in \mathbb{R}^{\tilde{T} \times D_h}$ from the above Heterogeneity-Aware Transformer through cross-attention, to produce a sequence of latent forecasting embeddings $\boldsymbol{h}^{\text{de}} \in \mathbb{R}^{J \times D_h}$.

Each embedding $\boldsymbol{h}_j^{\text{de}}$ is then projected into the observation space via a shared prediction head, which is implemented as a residual FFN block [11], to generate a future patch of length $H_s = \lceil H/J \rceil$ as

$$\hat{\mathbf{Y}}_{(j-1)H_s+1:jH_s} = \text{ResidualFFN}(\boldsymbol{h}_j^{\text{de}}), \quad j \in [1, J] \tag{7}$$

This multi-token design serves two primary objectives: (i) *error mitigation and computational efficiency* via parallel patch prediction, and (ii) *horizon flexibility* for exact alignment with various target lengths. We provide a detailed comparative analysis of this design in Appendix D. By pairing this flexible Multi-Patch Decoder with the adaptive Mixture-of-Size Encoder, KAIROS achieves a unified and parameter-efficient framework for zero-shot time series forecasting.

3.5 Training Corpus

For pre-training, we curated the Predictability-Stratified Time Series (PreSTS) corpus, comprising over 300 billion time points. PreSTS integrates large-scale real-world time series from diverse domains and a complementary synthetic dataset to ensure broad coverage. All test sets from the GIFT-Eval [3] and TSLib [39] benchmarks are explicitly excluded to prevent data leakage. While dataset diversity is widely regarded as essential for TSFM training, recent studies indicate that low-predictability or anomalous sequences can substantially impair forecasting [9]. To address this, we stratify real-world datasets into five tiers based on predictability and assign higher sampling probabilities to more predictable data during training. This strategy promotes stable and high-quality pre-training without compromising data diversity. Further details on dataset construction, along with an experimental analysis decoupling the contribution of the corpus from our model architecture, are provided in Appendix I.1 and Appendix G, respectively.

4 Experiment

In this section, we assess the performance of KAIROS and its components by addressing three research questions. **RQ1:** Does KAIROS achieve superior zero-shot generalization capabilities while maintaining parameter efficiency compared to previous methods? (Section 4.2) **RQ2:** Does the Mixture-of-Size Encoder effectively handle time series heterogeneity by adapting its granularity based on local information density? (Section 4.3 and 4.4.1) **RQ3:** Does DRoPE effectively generate customized position embeddings tailored to the unique temporal structure of each time series instance? (Section 4.3 and 4.4.2)

Table 1: Performance evaluation on the GIFT-Eval. We evaluate using normalized MASE and CRPS (Section 4.1). Baselines encompass statistical, Deep Learning (DL), and TSFM methods, with TSFMs categorized into TestData Leakage and Zero-Shot following official protocols. TestData Leakage signifies that the model's training set included test data, whereas Zero-Shot indicates that the model was trained without any data from the test set or its corresponding training split. Baseline results are officially reported by GIFT-Eval.

Type	**Statistical**	**DL (Full-Shot)**		**TSFMs (TestData Leakage)**				**TSFMs (Zero-Shot)**						
Method	Seasonal Naïve	DLinear	PTST.	TTM	Chronos	Chronos Bolt	TimesFM	Moirai	VisionTS	Ying.	Toto	Sundial	KAIROS$_s$ **(ours)**	KAIROS$_b$ **(ours)**
#Params	-	-	-	5M	709M	205M	500M	311M	112M	300M	151M	128M	23M	53M
MASE ↓	1.000	1.061	0.849	1.020	0.870	0.808	0.758	0.875	0.863	0.798	0.750	0.750	<u>0.748</u>	**0.738**
CRPS ↓	1.000	0.846	0.587	0.873	0.574	0.574	0.550	0.599	0.755	<u>0.548</u>	**0.517**	0.559	0.554	<u>0.548</u>

4.1 Experiment Settings

To comprehensively evaluate our proposed method, we conducted zero-shot assessments on two well-known public benchmarks: GIFT-Eval [3] and Time-Series-Library (TSLib) [39]. We train KAIROS in three sizes: mini (10M), small (23M) and base (53M). The training details are provided in Appendix H.1.

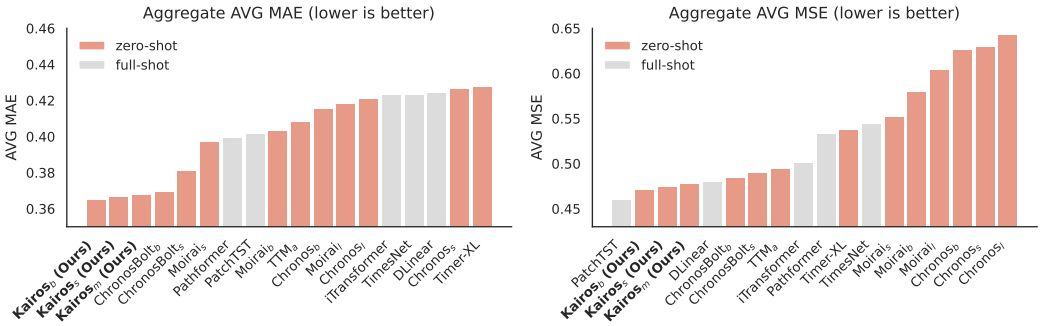


Figure 3: Zero-shot performance on TSLib averaged over prediction lengths {96, 192, 336, 720}. The subscripts l, b, s, m, and a represent model sizes of large, base, small, mini and advanced, respectively. The complete experimental results are presented in Appendix J.

Benchmarks. The GIFT-Eval benchmark comprises 97 tasks spanning short- (55), medium- (21), and long-term (21) horizons, facilitating a comprehensive assessment. For TSLib, we selected the ETT and Weather [46] datasets, consistent with Time-MoE [34]. To test adaptability to diverse frequencies, we included the Saugeen datasets [15], denoted as Saugeen (D) and Saugeen (W) for daily and weekly intervals, respectively. Details of the compared methods, evaluation datasets, splitting configurations, and evaluation length selection are provided in Appendices I.2, I.3, and I.4.

Metric. In our evaluation of GIFT-Eval, consistent with prior research [5, 28], we employ the official Mean Absolute Scaled Error (MASE) and Continuous Ranked Probability Score (CRPS) metrics to assess point and probabilistic forecasting performance, respectively. These metrics are first normalized by the Seasonal Naïve baseline for each individual task. The final score is then computed as the geometric mean of these normalized values. For the TSLib benchmark, we followed previous works [26, 27, 29] by adopting Mean Squared Error (MSE) and Mean Absolute Error (MAE) to evaluate time series forecasting performance. The metric calculations are detailed in Appendix H.2.1.

4.2 Zero-shot Evaluation

Finding 1: KAIROS *achieves superior zero-shot forecasting performance with significantly higher parameter efficiency and learns highly transferable representations.*. On the GIFT-Eval benchmark, as presented in Table 1, $\text{KAIROS}_{\text{base}}$ (53M) achieves the best MASE and the second-best CRPS among state-of-the-art task-specific deep learning methods and other TSFMs. Notably, $\text{KAIROS}_{\text{small}}$ (23M) surpasses both Toto and Sundial in MASE, despite having a parameter count that is 6.6 and 5.6 times smaller, respectively. On the TSLib benchmark, as shown in Figure 3, our lightweight $\text{KAIROS}_{\text{mini}}$ (10M) surpasses the performance of both recent advanced TSFMs and the majority of full-shot deep learning models. These results confirm that KAIROS delivers a significant improvement in predictive capability without relying on massive scale. Furthermore, we demonstrate in Appendix C that these learned representations are highly transferable to downstream classification tasks.

4.3 Ablation Study

Finding 2: *The synergistic integration of Mixture-of-Size Encoding, DRoPE, and the Multi-Patch Decoding is critical for adapting to heterogeneous time series and achieving superior forecasting performance.* Table 2 details the component-wise contributions. (i) *Encoder Design*: KAIROS outperforms variants using fixed patch sizes or excluding null experts, confirming that dynamic, sparse multi-granularity modeling is critical for

Table 2: Ablation study comparing encoder design, positional embedding and decoder strategy. Models were evaluated using the normalized MASE (detailed in Section 4.1) across prediction horizons and as an aggregate across all tasks.

Model Variants	Short	Medium	Long	AVG
Impact of Encoder Design				
w/ Fixed Patch Size ($P = 32$)	0.724	0.802	0.820	0.761
w/ Mixture-of-Size (w/o Null Experts)	0.720	0.770	0.800	0.748
Impact of Positional Embedding				
w/ Standard RoPE (Fixed θ)	0.729	0.807	0.835	0.767
w/ Only Instance-Level Frequency Modulation	0.719	0.767	0.797	0.746
w/ Only Granularity-Aware Position Calibration	0.727	0.796	0.807	0.758
Impact of Decoder Strategy				
w/ Single-Patch Autoregressive	**0.705**	0.794	0.848	0.753
KAIROS (Full Model)	0.709	**0.761**	**0.794**	**0.738**

capturing diverse temporal patterns. (ii) *Positional Embedding*: DRoPE proves superior to standard RoPE [35]. Our ablation further isolates the gains from *Instance-Level Frequency Modulation* and *Granularity-Aware Position Calibration*, verifying the benefit of adapting to both spectral heterogeneity and physical time distortion caused by dynamic patching. (iii) *Decoding Strategy*: The Multi-Patch Decoding shows clear superiority over standard single-patch autoregressive decoding. The integration yielded the best performance across 97 GIFT-Eval tasks, which validates the superiority of our designs. Furthermore, the performance gap widens as the forecast horizon increases, demonstrating KAIROS's superior long-term forecasting capability.

4.4 Model Analysis

4.4.1 Mixture-of-Size Tokenization Analysis

Finding 3: KAIROS *demonstrates adaptive capability to varying information densities, enabling it to model complex temporal dynamics with appropriate granularity.* Instead of enforcing a uniform resolution, KAIROS dynamically routes segments to optimal granularity based on local information. As visualized in Figure 4, KAIROS strategically allocates fine-grained tokens to regions with abrupt changes or high volatility, while employing coarse-grained tokens for stable trends. This adaptive allocation mechanism reduces computational redundancy on low-information segments, allowing the model to achieve superior performance with a compact parameter budget compared to rigid architectures. See Appendix F for a detailed inference speed analysis. Furthermore, to demonstrate that KAIROS learns genuine segment-level adaptation rather than merely relying on a generic local multi-scale inductive bias, we provide additional experimental evidence in Appendix E.

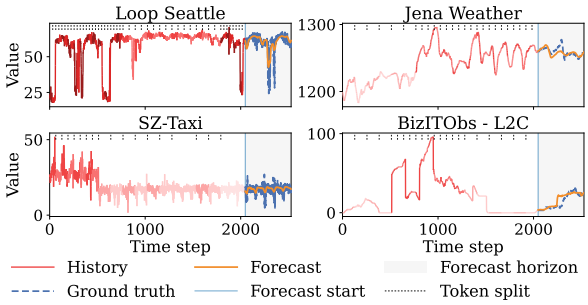


Figure 4: Adaptive tokenization visualization. Tokens are demarcated by dotted lines and varying colors, where **deeper red** curves indicate finer granularity. KAIROS adapts granularity to information density, applying finer processing to volatile regions.

4.4.2 DRoPE Analysis

Finding 4: *Modulating RoPE frequencies θ instance-wise can indeed better model time series temporal relationships.* To verify that DRoPE's performance gains stem from its instance-specific θ_{inst} modulation, we designed control experiments that disrupt this mechanism (Table 3; Appendix I.5): (i) Intra-Dataset Shuffle: randomly permute θ_{inst} across instances within the same dataset; (ii) Inter-Dataset Shuffle: assign θ_{inst} from instances of different datasets; (iii) Fixed RoPE: use standard RoPE without any modulation. The results reveal a clear performance gradient (DRoPE $\succ$ Intra-Shuffle $\succ$ Fixed RoPE $\succ$ Inter-Shuffle), indicating that DRoPE learns instance-specific representations to modulate θ. Intra-Dataset Shuffle outperforms Fixed RoPE, as samples within the same dataset share similar spectral characteristics, allowing shuffled θ_{inst} to remain partially compatible. However, Intra-Dataset Shuffle still incurs a 1.76% degradation compared to DRoPE, suggesting that instance-level adaptation contributes to the performance gains beyond dataset-level statistics. Inter-Dataset Shuffle yields the worst results (28.32% degradation), as θ_{inst} from different datasets cannot capture the temporal dynamics of the target series. These results collectively validate the effectiveness of instance-level θ_{inst} adaptation in DRoPE.

Table 3: Causal analysis of adaptive modulation by DRoPE. We evaluate the impact of disrupting or removing the instance-wise θ adjustment. Degradation indicates relative performance drop compared to DRoPE.

Method	DRoPE (Ours)	Intra-Dataset Shuffle	Standard RoPE (Fixed θ)	Inter-Dataset Shuffle
MASE ↓	**0.738**	0.751	0.767	0.947
Degradation	–	-1.76%	-3.93%	-28.32%

5 Conclusions

We introduce KAIROS, a parameter-efficient foundation model explicitly tailored for time series heterogeneity. Its architecture features three key innovations: a Mixture-of-Size Encoder for adaptive granularity, a Heterogeneity-Aware Transformer with Dynamic RoPE for instance-specific temporal calibration, and a Multi-Patch Decoder for efficient parallel forecasting. Extensive experiments demonstrate that KAIROS achieves superior performance, validating the necessity of aligning architectural inductive biases with the structural nuances of time series data. One limitation of our current approach is that KAIROS relies on channel-independent modeling [29] to support multivariate time series and does not explicitly capture inter-variable dependencies. Future work aims to model these interactions and extend KAIROS to tasks beyond forecasting.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

[3] Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Gift-eval: A benchmark for general time series forecasting model evaluation. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024.

[4] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Syndar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=gerNCVqqtR`.

[5] Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. Tirex: Zero-shot forecasting across long and short horizons with enhanced in-context learning. *arXiv preprint arXiv:2505.23719*, 2025.

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

[7] Mouxiang Chen, Lefei Shen, Zhuo Li, Xiaoyun Joy Wang, Jianling Sun, and Chenghao Liu. Visionts: Visual masked autoencoders are free-lunch zero-shot time series forecasters. In *Forty-second International Conference on Machine Learning*, 2025.

[8] Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. *arXiv preprint arXiv:2402.05956*, 2024.

[9] Hao Cheng, Qingsong Wen, Yang Liu, and Liang Sun. Robusttsf: Towards theory and design of robust time series forecasting with anomalies. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, et al. This time is different: An observability perspective on time series foundation models. *arXiv preprint arXiv:2505.14766*, 2025.

[11] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

[12] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[13] Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam H. Nguyen, Wesley M. Gifford, Chandra Reddy, and Jayant Kalagnanam. Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series, 2024.

[14] Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38, 1994.

[15] Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive. In *Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[16] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. In *International Conference on Machine Learning*, 2024.

[17] Lars Graf, Thomas Ortner, StanisĹ WoĹşniak, Angeliki Pantazi, et al. Flowstate: Sampling rate invariant time series forecasting. *arXiv preprint arXiv:2508.05287*, 2025.

[18] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36: 19622–19635, 2023.

[19] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[20] Peng Jin, Bo Zhu, Li Yuan, and Shuicheng YAN. Moe++: Accelerating mixture-of-experts methods with zero-computation experts. In *The Thirteenth International Conference on Learning Representations*, 2024.

[21] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[22] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International journal of forecasting*, 37(4): 1748–1764, 2021.

[23] Hongzhan Lin, Ang Lv, Yang Song, Hengshu Zhu, Rui Yan, et al. Mixture of in-context experts enhance llms' long context awareness. *Advances in Neural Information Processing Systems*, 37: 79573–79596, 2024.

[24] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

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

[26] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

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

[28] Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. *arXiv preprint arXiv:2502.00816*, 2025.

[29] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

[30] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1ecqn4YwB.

[31] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Hassen, Anderson Schneider, et al. Lag-llama: Towards foundation models for time series forecasting. In *R0-FoMo: Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023.

[32] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

[33] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

[34] Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts, 2024. URL https://arxiv.org/abs/2409.16040.

[35] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[37] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

[38] Xue Wang, Tian Zhou, Jinyang Gao, Bolin Ding, and Jingren Zhou. Output scaling: Yinglong-delayed chain of thought in a large pretrained time series forecasting model. *arXiv preprint arXiv:2506.11029*, 2025.

[39] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Chen Wang, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. *arXiv preprint arXiv:2407.13278*, 2024.

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

[41] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

[42] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

[43] Zihao Zeng, Yibo Miao, Hongcheng Gao, Hao Zhang, and Zhijie Deng. Adamoe: Token-adaptive routing with null experts for mixture-of-experts language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6223–6235, 2024.

[44] Jiawen Zhang, Shun Zheng, Xumeng Wen, Xiaofang Zhou, Jiang Bian, and Jia Li. Elastst: Towards robust varied-horizon forecasting with elastic time-series transformer. *arXiv preprint arXiv:2411.01842*, 2024.

[45] Chuanyang Zheng, Yihang Gao, Han Shi, Minbin Huang, Jingyao Li, Jing Xiong, Xiaozhe Ren, Michael Ng, Xin Jiang, Zhenguo Li, et al. Dape: Data-adaptive positional encoding for length extrapolation. *Advances in Neural Information Processing Systems*, 37:26659–26700, 2024.

[46] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

[47] Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023.

A Limitations and Future Work

While KAIROS demonstrates superior parameter efficiency and zero-shot performance, we acknowledge several limitations that provide avenues for future research.

Channel-Independent Modeling. Currently, KAIROS adopts a channel-independent modeling approach [29] to handle multivariate time series, treating each variable as an individual sequence without explicitly modeling the relationships between variables. However, our architectural innovations are orthogonal to the channel-independence assumption. It is easy to incorporate channel dependency modeling for further enhancement. For instance, a channel-mixer block can be easily integrated to explicitly capture cross-channel correlations. We plan to explore these extensions in future work.

Evaluation Scope. Our empirical evaluation primarily focuses on zero-shot forecasting, which is the primary evaluation setting for TSFMs [28, 10]. As a generative task, forecasting directly reflects whether a model can capture temporal structures such as trend, seasonality, and local dynamics. Furthermore, we provide evaluation results on classification tasks in Appendix C, which demonstrate that KAIROS successfully learns transferable representations. We plan to further extend KAIROS to tasks such as anomaly detection and imputation in future versions.

B Etymology of KAIROS

KAIROS was chosen for its mythological significance as the Greek god of the "right or critical moment", reflecting our model's ability to select the ideal set of granularities and positional encoding for time series characterized by varying information densities and frequencies. Furthermore, this continues a thematic tradition seen in other foundational models we benchmark against, such as Chronos [4] (the personification of time) and Moirai [40] (the personifications of destiny).

C Transferability on Classification Tasks

To verify whether KAIROS learns generalizable features that transfer beyond forecasting tasks, we evaluate its performance on downstream time series classification.

Experimental Setup. Following MOMENT [16], we evaluate on 91 datasets from the UCR archive [12]. To isolate representation quality, KAIROS-Base and MOMENT use a frozen encoder with a lightweight classifier, comparing them against fully fine-tuned baselines (GPT4TS [47], TimesNet [41]).

Results. The classification results are summarized in Table 4. Despite relying solely on frozen representations, KAIROS consistently outperforms both MOMENT and the fully fine-tuned baselines. These results provide evidence that our architectural innovations do not merely overfit to forecasting dynamics. Instead, by addressing temporal heterogeneity, KAIROS extracts high-level semantic features that are highly transferable to adaptation-heavy settings like classification. This solidifies KAIROS's capacity as a general-purpose foundation model capable of supporting diverse downstream time series applications.

Table 4: Representation transfer results on 91 UCR classification datasets. Baseline results follow MOMENT [16].

Model	Accuracy ↑	Rank Mean ↓	Rank Std ↓
MOMENT (Frozen)	0.794	1.90	0.80
GPT4TS (Fine-tuned)	0.567	3.27	0.81
TimesNet (Fine-tuned)	0.573	3.32	0.74
KAIROS-Base (Ours, Frozen)	**0.818**	**1.51**	**0.71**

D Analysis of Multi-Patch Prediction

Unlike task-specific, non-autoregressive models such as TFT [22] and Informer [46], which predict all future time points simultaneously, their architecture confines them to forecasting a fixed length

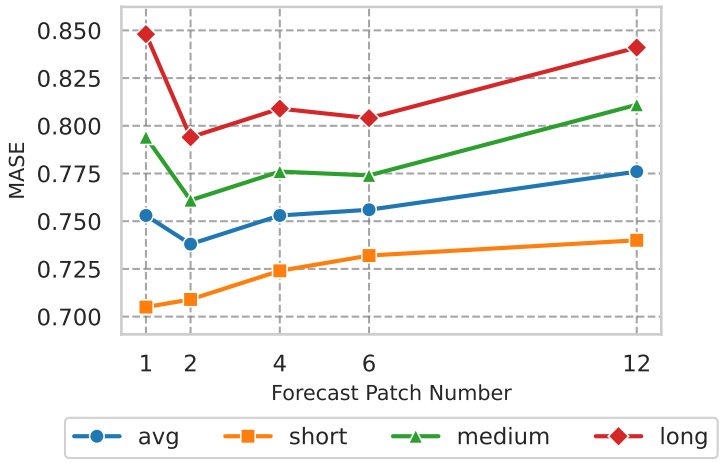


Figure 5: Performance analysis of multi-patch prediction on the GIFT-Eval benchmark across short, medium, and long horizons. We vary the number of forecast patches and find that a forecast patch number of 2 achieves the optimal trade-off, resulting in the best overall performance (lowest normalized MASE).

identical to the predefined prediction horizon used during training, precluding variable-length predictions during inference. Conversely, TSFMs typically employ an iterative process: they predict a segment of the sequence, append it to the historical data, and then forecast the subsequent segment, thereby enabling autoregressive predictions of arbitrary length. To mitigate the cumulative errors inherent in this autoregressive methodology, KAIROS introduces forecast tokens designed to predict multiple patches within each autoregressive step.

As briefly mentioned in Section 3.4, the multi-token design of our Multi-Patch Decoder is strategically motivated by two limitations in predominant forecasting paradigms: (i) *Error mitigation and efficiency*: Compared to conventional single-step autoregressive models [4, 27], predicting J patches in parallel improves computational efficiency and mitigates cumulative error by reducing iterative decoding steps. (ii) *Horizon flexibility*: Unlike models that generate a single larger patch [11], our approach offers superior granularity for variable prediction horizons. By assembling the forecast from the specific subset of tokens required, KAIROS ensures high-fidelity alignment with the target length and avoids the imprecise truncation associated with oversized outputs.

To empirically validate this design and determine the optimal setting, we present a detailed analysis of the multi-patch prediction strategy. During the training phase, we experimented with a range of forecast patch numbers, specifically $J = \{1, 2, 4, 6, 12\}$. The corresponding evaluation results on the GIFT-Eval benchmark are illustrated in Figure 5. Our observations reveal that when $J = 1$, which corresponds to the conventional approach of predicting a single patch [4, 27, 11], the model achieves optimal performance in short-term forecasting. However, this method necessitates multiple iterations of autoregressive prediction, leading to a significant degradation in performance for medium- and long-term forecasting.

Conversely, as we increase the forecast patch number J, the number of required autoregressive steps is markedly reduced. For instance, when $J = 2$, the autoregression frequency for medium- and long-term predictions is halved compared to the $J = 1$ case. This reduction yields a substantial improvement in forecasting accuracy over these longer horizons, demonstrating that our proposed multi-patch prediction strategy effectively mitigates the cumulative error inherent in autoregressive processes for medium- and long-term forecasting.

Nevertheless, we noted that the forecasting performance does not improve indefinitely with an increasing the forecast patch number J. We attribute this phenomenon to the escalated difficulty of the prediction task, which hinders the model's ability to optimize effectively. Ultimately, by evaluating the mean normalized MASE, we identified the optimal trade-off, selecting a forecast patch number of $J = 2$ for KAIROS.

E Validation of Learned Segment-Level Adaptation

To explicitly demonstrate that the performance gains of the Mixture-of-Size Encoder stem from genuine segment-level adaptation rather than merely a generic local multi-scale inductive bias, we conduct two dedicated analyses: a structural comparison with a sequence-level multi-scale baseline, and a direct causal intervention on the learned routing decisions.

Advantage of Intra-Sequence Adaptation. We separate the benefits of local intra-sequence adaptation from global multi-scale availability. To achieve this, we replaced our Mixture-of-Size Encoder with the sequence-level multi-scale routing mechanism proposed in Pathformer [8]. While Pathformer adaptively selects patch sizes for each sequence, it applies these selected granularities uniformly across the entire input, ignoring intra-sequence heterogeneity. As shown in Table 5, this replacement degrades GIFT-Eval MASE from 0.738 to 0.759, underscoring the advantage of local intra-sequence adaptation over global sequence-level multi-scale modeling.

Causal Intervention on Routing Decisions. To more directly test whether the router makes meaningful local decisions, we performed causal interventions on the routing mechanism at inference time. We kept all model weights and architectural components unchanged, but actively manipulated the router's outputs:

- *Uniform Weights:* Assigning uniform weights across all granularities within each segment, which removes adaptive selection while preserving the multi-scale structure.
- *Shuffled Routing:* Randomly shuffling the router-produced granularity weights within each segment, which disrupts the learned routing decisions.

Assigning uniform weights across all granularities degrades performance from 0.738 to 0.831, indicating that the gain cannot be explained by multi-scale modeling alone. When we further disrupt the learned routing policy by randomly shuffling the granularity weights, performance deteriorates much more sharply to 1.205. Taken together, these results provide direct intervention evidence that KAIROS benefits not simply from a generic multi-scale gain, but from assigning granularities appropriately to each segment.

Table 5: Empirical validation of segment-level adaptation on GIFT-Eval. Results report the normalized MASE across prediction horizons and as an aggregate across all tasks.

Setting	Short	Medium	Long	AVG
KAIROS (Full Model)	**0.709**	**0.761**	**0.794**	**0.738**
Structural Replacement				
w/ Sequence-Level Routing (Pathformer)	0.730	0.782	0.819	0.759
Causal Interventions at Inference				
w/ Uniform Granularity Weights	0.856	0.780	0.819	0.831
w/ Shuffled Routing Decisions	1.154	1.282	1.271	1.205

F Analysis of Inference Speed

To provide a comprehensive view of our model's efficiency, we benchmarked the single-batch inference speed of KAIROS against several state-of-the-art models. The experimental setup involved an input sequence length of 2048 and a prediction horizon of 96 on a single NVIDIA TITAN RTX GPU. For TTM-Advanced [13], its maximum supported input length of 1536 was used.

As shown in Table 6, KAIROS-Base occupies a superior position on the efficiency-performance frontier. Its inference time (0.061s) is on the same order of magnitude as other highly efficient models like ChronosBolt and Moirai, yet it delivers significantly better zero-shot accuracy (0.738 MASE). Compared to high-capacity models like Toto and Chronos, KAIROS is several orders of magnitude faster while maintaining higher forecasting precision.

G Decoupling Architecture and Data Contributions

In this section, we provide a dedicated experimental analysis to explicitly decouple the performance contributions of our proposed PreSTS corpus from the architectural innovations of KAIROS.

Table 6: Comparison of single-batch inference speeds and zero-shot performance on GIFT-Eval. All models were tested with an input length of 2048 and an output length of 96, except for TTM-Advanced (*), which used its maximum input length of 1536. Inference times are averaged per batch on a single GPU.

Model	Inference Time (s) ↓	MASE ↓
TTM-Advanced*	**0.009**	0.812
Timer-XL	0.014	0.795
ChronosBolt-Base	0.055	0.781
KAIROS-Base	0.061	**0.738**
Moirai-Large	0.070	0.765
Sundial-Base	0.099	0.750
Time-MoE-Large	0.148	0.755
TimesFM-2.0	0.202	0.760
Chronos-Large	4.901	0.876
Toto-Base	13.717	0.750

Ablation on Data Curation. We isolate the impact of our data curation components, specifically the predictability-based tier-stratified sampling and the inclusion of synthetic data. As shown in Table 7, removing the data curation components degrades the performance on the GIFT-Eval benchmark, confirming their utility in providing high-quality supervision. However, this degradation is notably smaller than the performance drop observed when removing the main architectural components. This indicates that while the PreSTS data curation strategy provides a complementary gain, the architectural design serves as the primary contributor to the overall performance of KAIROS.

Table 7: Ablation study decoupling data curation and architectural components. Models were evaluated using the normalized MASE on GIFT-Eval.

Variant	MASE ↓
Data Curation Ablations	
w/o tier-stratified sampling	0.748
w/o tier-stratified sampling & synthetic data	0.755
Architectural Ablations	
w/o Multi-Patch Decoder	0.753
w/o Mixture-of-Size Encoder	0.761
w/o DRoPE	0.767
KAIROS (Full Model)	**0.738**

Matched-Data Comparison. To further disentangle the architecture from the pre-training data, we perform a matched-data comparison. Specifically, we train KAIROS on the original Chronos corpus [4] and train the advanced baseline ChronosBolt on our proposed PreSTS corpus. The results are summarized in Table 8.

Table 8: Matched-data comparison on the GIFT-Eval benchmark separating architecture and training data contributions.

Model	Params	Training Data	MASE ↓
KAIROS	53M	PreSTS	**0.738**
KAIROS	53M	Chronos corpus	0.761
ChronosBolt	205M	PreSTS	0.781
ChronosBolt	205M	Chronos corpus	0.808
Chronos	709M	Chronos corpus	0.870

These matched-data results demonstrate that the architectural contribution significantly outweighs the data-curation contribution in our setting:

- KAIROS trained on the Chronos corpus (0.761) still outperforms ChronosBolt trained on PreSTS (0.781), despite using about 4× fewer parameters. This indicates that KAIROS retains a clear advantage even without the proposed PreSTS corpus.
- Under the same Chronos corpus, the gap between KAIROS and ChronosBolt is 0.047 (0.761 vs. 0.808), whereas for ChronosBolt the gain from replacing the Chronos corpus with PreSTS is 0.027 (0.808 to 0.781). This suggests that, in our experiments, architectural innovation is the primary source of improvement, while PreSTS provides an additional but smaller gain.

H Implementation Details

H.1 Training Details

H.1.1 Model Configurations

We train KAIROS in three sizes: mini (10M), small (23M) and base (53M) with the detailed model configurations are in Table 9. The base model are trained for 300,000 steps with batch sizes of 512. We employ the AdamW optimizer, a linear decay learning rate adjustment strategy for model optimization. The learning rate for parameters related to DRoPE is set to 1e-5, while the learning rate for others is set to 1e-3. Training is conducted on 4 × NVIDIA A100 GPUs using TF32 precision, which takes only 15 hours for base size.

Table 9: Details of KAIROS model configurations.

	Layers	Heads	d_{model}	d_{ff}	d_{expert}	P	K	G	Z	τ	η_b	w	θ	Params
$\text{KAIROS}_{\text{mini}}$	4	4	256	1024	1408	$\{32, 64, 128\}$	3	3	2	$\{0.55, 0.1, 0.05, 0.15, 0.15\}$	0.01	128	$10000^{-2j/64}$	10M
$\text{KAIROS}_{\text{small}}$	4	8	384	1536	1408	$\{32, 64, 128\}$	3	3	2	$\{0.55, 0.1, 0.05, 0.15, 0.15\}$	0.01	128	$10000^{-2j/64}$	23M
$\text{KAIROS}_{\text{base}}$	6	8	512	2048	1408	$\{32, 64, 128, 256\}$	4	4	2	$\{0.55, 0.1, 0.05, 0.15, 0.15\}$	0.01	128	$10000^{-2j/64}$	53M

H.1.2 Loss Function

To better accommodate varied forecast horizons, and following the methodology of ElasTST [44], we build upon the standard quantile loss by assigning distinct weights to each timestep, such that earlier predictions are given greater importance. The model's parameters are optimized by minimizing the quantile loss with weight decay, formulated as:

$$\mathcal{L} = \frac{1}{B}\sum_{b=1}^{B}\sum_{t=1}^{H}\frac{1}{Q}\sum_{k=1}^{Q}\omega(t)L_{\alpha_k}(y_{b,t}, q_{b,t}(\alpha_k)), \tag{8}$$

$$\omega(t) = \frac{1}{H}(\ln(H) - \ln(t')), \quad L_\alpha(y, q) = (\alpha - \mathbf{1}_{\{y<q\}})(y - q), \tag{9}$$

where B, H, Q are the batch size, forecast horizon, and total number of quantiles, respectively; $y_{b,t}$ is the ground truth value, $q_{b,t}(\alpha_k)$ is the k-th quantile forecast, $\omega(t)$ is the weight at each time step t, and L_{α_k} denotes the pinball loss function. To prevent the loss weight from evaluating to zero at the final prediction step ($t = H$), we evaluate the logarithmic decay using t', which maps the discrete timesteps $t \in \{1, \ldots, H\}$ to an evenly spaced continuous sequence from $1 + \epsilon_1$ to $H - \epsilon_2$ (specifically, $\epsilon_1 = 10^{-5}$ and $\epsilon_2 = 10^{-3}$ in our implementation). Empirically, we set $Q = 9$ and use quantile levels of $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$.

H.2 Evaluation Details

H.2.1 Metric

GIFT-Eval. GIFT-Eval employs the Mean Absolute Scaled Error (MASE) and the Continuous Ranked Probability Score (CRPS) to assess the performance of point and probabilistic forecasts, respectively. Following the official evaluation protocol, we normalize the metrics for each task using a seasonal naïve baseline and subsequently aggregate the scores across all tasks via the geometric mean.

TSLib. We adopt mean square error (MSE) and mean absolute error (MAE) as evaluation metrics. These metrics are calculated as follows:

$$\text{MSE} = \frac{1}{H}\sum_{i=1}^{H}(x_i - \hat{x}_i)^2, \quad \text{MAE} = \frac{1}{H}\sum_{i=1}^{H}|x_i - \hat{x}_i| \tag{10}$$

where x_i is the ground truth and $\hat{x}_i$ is the prediction for the i-th future time point.

H.2.2 Stride

Due to the significant inference latency of our baseline, Chronos [4], we set the evaluation stride to 96 to enhance evaluation efficiency without compromising fairness. This setting is consistent with the protocols of Moirai [40], Moirai-MoE [25], and the GIFT-Eval benchmark [3], all of which set the stride equal to the prediction length. Additionally, the original Chronos paper evaluates the model on only a single window per dataset. Therefore, we wish to emphasize a critical point: while the specific numerical results would likely vary with a different stride, our chosen protocol is applied uniformly to all models under evaluation, ensuring a fair and equitable comparison.

H.3 Mixture-of-Size Encoder

H.3.1 Auxiliary-Loss-Free Load Balancing

In this section, we elaborate on the details of bias previously introduced in Section 3.2. In order to ensure that different experts are adequately trained and to control the distribution ratios of various patch sizes, we employ an Auxiliary-Loss-Free Load Balancing method similar to that used in DeepSeek-V3 [24].

Specifically, we compute the empirical load L_i for the i-th expert by summing its normalized routing weights across all N segments of all B sequences within a training batch:

$$L_i = \sum_{b=1}^{B}\sum_{n=1}^{N}\tilde{s}_{b,n,i}, \tag{11}$$

where $\tilde{s}_{b,n,i}$ denotes the routing weights for the i-th expert on the n-th segment of the b-th sequence, which directly corresponds to the definition of $\tilde{s}_i$ in Section 3.2.

We define a target load distribution $\tau = (\tau_1, \ldots, \tau_{G+Z})$, where τ_i is the desired proportion of the total load for expert i, satisfying $\sum_{j=1}^{G+Z}\tau_j = 1$. At the end of each training step, we update the bias term $\boldsymbol{b}_i$ using the empirical load L_i and the target τ:

$$\boldsymbol{b}_i \leftarrow \boldsymbol{b}_i + \eta_b \cdot \frac{\tau_i \cdot \sum_{j=1}^{G+Z} L_j - L_i}{\sum_{j=1}^{G+Z} L_j}, \tag{12}$$

where η_b is a hyper-parameter governing the magnitude of this adjustment, referred to as the bias update speed.

This dynamic adjustment of $\boldsymbol{b}_i$ aims to balance the workload across the experts according to the desired distribution by influencing future top-K selections, while also steering the patches-to-expert affinity scores $s'_{n,i}$ in subsequent batches, ensuring that the actual load distribution L_i progressively aligns with the target distribution τ_i, thereby promoting balanced expert utilization over time.

H.3.2 Definition of the Expansion Function

In this section, we provide the formal definition for the function $\mathrm{Expand}(\cdot)$ within the Mixture-of-Size Fusion stage, as introduced in Section 3.2.

Consider a single time series segment $\mathbf{x}$ of length P. During the Top-K Granularity Routing, a set of valid experts $\mathcal{I}_{\text{valid}}$ is activated. Let $P_{\min}$ denote the finest granularity among these activated experts:

$$P_{\min} = \min\{P_j \mid j \in \mathcal{I}_{\text{valid}}\}. \tag{13}$$

Consequently, the target sequence length for alignment is defined as $M = \lceil P/P_{\min} \rceil$.

For an activated granularity level i with patch size P_i, the encoding process yields a sequence of embeddings $\boldsymbol{E}_i = [\boldsymbol{e}_{i,1}, \ldots, \boldsymbol{e}_{i,M_i}]$, where $M_i = \lceil P/P_i \rceil$. Due to the nested design of patch sizes, the ratio $R_i = P_i/P_{\min}$ is strictly a positive integer.

The function $\mathrm{Expand}(\boldsymbol{E}_i)$ maps the coarser embedding sequence to the finest temporal resolution by repeating each embedding vector R_i times. Formally, the k-th element of the expanded sequence $\tilde{\boldsymbol{E}}_i \in \mathbb{R}^{M \times D_h}$ is derived from the original sequence via index mapping:

$$\tilde{\boldsymbol{e}}_{i,m} = \boldsymbol{e}_{i,\lceil m/R_i \rceil}, \quad \text{for } m = 1, \ldots, M. \tag{14}$$

In vector notation, this operation can be expressed as:

$$\mathrm{Expand}(\boldsymbol{E}_i) = [\underbrace{\boldsymbol{e}_{i,1}, \ldots, \boldsymbol{e}_{i,1}}_{R_i \text{ times}}, \ldots, \underbrace{\boldsymbol{e}_{i,M_i}, \ldots, \boldsymbol{e}_{i,M_i}}_{R_i \text{ times}}]. \tag{15}$$

We now illustrate the computational process with the following example. Let the setup be as follows:

- Segment length: $P = 128$.
- Set of available granularity patch sizes: $\{P_1, P_2, P_3\} = \{32, 64, 128\}$.
- Number of null experts: $Z = 2$.
- Top-K selection: $K = 3$.

Suppose for a specific segment $\mathbf{x}$, the router activates experts corresponding to granularity levels 1 and 3, alongside one null expert (i.e., the selected set is $\{1, 3, \text{null}\}$ while $\mathcal{I}_{\text{valid}} = \{1, 3\}$). The finest activated resolution is $P_{\min} = \min(32, 128) = 32$. The target aligned sequence length is $M = 128/32 = 4$.

The expansion process for each expert is computed as follows:

- **For Expert 1 ($P_1 = 32$):**
  The number of original patches is $M_1 = 128/32 = 4$. The embedding sequence is $\boldsymbol{E}_1 = [\boldsymbol{e}_{1,1}, \boldsymbol{e}_{1,2}, \boldsymbol{e}_{1,3}, \boldsymbol{e}_{1,4}]$. The repetition factor is $R_1 = 32/32 = 1$.

  $$\tilde{\boldsymbol{E}}_1 = \mathrm{Expand}(\boldsymbol{E}_1) = [\boldsymbol{e}_{1,1}, \boldsymbol{e}_{1,2}, \boldsymbol{e}_{1,3}, \boldsymbol{e}_{1,4}]. \tag{16}$$

  Since this expert operates at the finest activated resolution, the expansion is an identity mapping.
- **For Expert 3 ($P_3 = 128$):**
  The number of original patches is $M_3 = 128/128 = 1$. The embedding sequence consists of a single vector $\boldsymbol{E}_3 = [\boldsymbol{e}_{3,1}]$. The repetition factor is $R_3 = 128/32 = 4$.

  $$\tilde{\boldsymbol{E}}_3 = \mathrm{Expand}(\boldsymbol{E}_3) = [\boldsymbol{e}_{3,1}, \boldsymbol{e}_{3,1}, \boldsymbol{e}_{3,1}, \boldsymbol{e}_{3,1}]. \tag{17}$$

  The single coarse embedding is broadcast across all 4 time steps to align with the target resolution.
- **Hypothetical Case for Expert 2 ($P_2 = 64$):**
  Although Expert 2 was not activated in this example, if it were processed, $M_2 = 128/64 = 2$ and $R_2 = 64/32 = 2$. The expansion would be:

  $$\tilde{\boldsymbol{E}}_2 = [\boldsymbol{e}_{2,1}, \boldsymbol{e}_{2,1}, \boldsymbol{e}_{2,2}, \boldsymbol{e}_{2,2}]. \tag{18}$$

By explicitly aligning all representations to the finest resolution $P_{\min}$ via $\mathrm{Expand}(\cdot)$, KAIROS ensures that the subsequent weighted summation $\bar{\boldsymbol{E}} = \sum_{i \in \mathcal{I}_{\text{valid}}} \alpha_i \tilde{\boldsymbol{E}}_i$ is performed on element-wise compatible tensors, effectively fusing global context with local details.

H.4 Dynamic Rotary Position Embedding (DRoPE)

In this section, we provide a more detailed description of the DRoPE implementation discussed in Section 3.3. We first introduce the principles of Rotary Position Embedding (RoPE) [35] in detail, and then provide corresponding supplementary information on the DRoPE implementation.

H.4.1 Details of RoPE

The core idea of RoPE is to rotate segments of the query and key vectors by an angle proportional according to their absolute position in the sequence. This allows the model to discern relative positions through the geometry of these rotations, without needing explicit calculation of relative distances. Throughout this subsection, we use m and n to denote generic token positions.

Specifically, RoPE operates on vectors of an even dimension, denoted as D_h. For a vector $\boldsymbol{e}$ (representing a query $\mathbf{q}$ or a key $\mathbf{k}$ in self-attention) at position m, it is transformed by a rotation matrix $\mathbf{R}_m$. This matrix is block-diagonal, composed of $D_h/2$ individual 2×2 rotation blocks. Each block $\mathbf{R}_{m,d}$ acts on a pair of dimensions (e_{2d}, e_{2d+1}) of the vector:

As introduced in Section 3.3, RoPE applies a rotation to input vector. This rotation is applied to pairs of dimensions (e_{2d}, e_{2d+1}), and can be concisely expressed using complex form:

$$f_{\text{RoPE}}(\boldsymbol{z}, m) = (z_{2d} + iz_{2d+1})e^{im\theta_d}, \tag{19}$$

where θ_d is the angular frequency. Here, i denotes the imaginary unit. This multiplication by $e^{im\theta_d}$ corresponds to a rotation in the complex plane. For the real-valued components z_{2d} and z_{2d+1}, this operation is equivalent to applying the following 2×2 rotation matrix $\mathbf{R}_{m,d}$:

$$\mathbf{R}_{m,d} = \begin{bmatrix} \cos(m\theta_d) & -\sin(m\theta_d) \\ \sin(m\theta_d) & \cos(m\theta_d) \end{bmatrix}, \tag{20}$$

Here, m is the absolute position of the token. The term θ_d represents a predefined angular frequency for the d-th pair of dimensions ($d \in [0, D_h/2 - 1]$), typically defined as $\theta_d = b^{-2d/D_h}$

The full rotation matrix $\mathbf{R}_m$ for position m is thus:

$$\mathbf{R}_m = \text{diag}(\mathbf{R}_{m,0}, \mathbf{R}_{m,1}, \ldots, \mathbf{R}_{m,D_h/2-1}) \tag{21}$$

$$= \begin{pmatrix} \cos m\theta_0 & -\sin m\theta_0 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_0 & \cos m\theta_0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_1 & -\sin m\theta_1 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_1 & \cos m\theta_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{D_h/2-1} & -\sin m\theta_{D_h/2-1} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{D_h/2-1} & \cos m\theta_{D_h/2-1} \end{pmatrix}. \tag{22}$$

Let $\mathbf{q}_m$ and $\mathbf{k}_n$ be the original query and key vectors for tokens at positions m and n respectively. After applying RoPE, their new representations become $\mathbf{q}'_m = \mathbf{R}_m\mathbf{q}_m$ and $\mathbf{k}'_n = \mathbf{R}_n\mathbf{k}_n$. The dot product for attention is then:

$$(\mathbf{q}'_m)^\top(\mathbf{k}'_n) = (\mathbf{R}_m\mathbf{q}_m)^\top(\mathbf{R}_n\mathbf{k}_n) = \mathbf{q}_m^\top\mathbf{R}_m^\top\mathbf{R}_n\mathbf{k}_n, \tag{23}$$

due to the properties of rotation matrices, $\mathbf{R}_m^\top\mathbf{R}_n = \mathbf{R}_{n-m}$ we can get:

$$(\mathbf{q}'_m)^\top(\mathbf{k}'_n) = \mathbf{q}_m^\top\mathbf{R}_{n-m}\mathbf{k}_n. \tag{24}$$

This equation shows that the dot product between a query and key vector after rotation inherently depends on their original values $(\mathbf{q}_m, \mathbf{k}_n)$ and their relative positions $(n - m)$. This allows RoPE to integrate relative positional information into the self-attention without any additional learnable parameters or explicit relative position computations.

Equation 24 confirms that the interaction relies on the relative position $n - m$ through the full rotation matrix $\mathbf{R}_{n-m}$. Since this matrix is block-diagonal, the total interaction is actually composed of independent rotations on disjoint 2D subspaces. The following theorem demonstrates the periodicity of the attention score within each subspace.

Theorem H.1 (Periodic dependence of RoPE attention)**.** *Consider the d-th 2D subspace of the hidden state under RoPE, with angular frequency θ_d, and let $\text{atten}_d(m, n)$ denote this subspace's*

contribution to the dot-product attention between positions m and n. Then there exist an amplitude $A_d(m,n) \geq 0$ and a phase $\varphi_d(m,n)$, depending only on the content vectors at m and n, such that

$$\begin{aligned} \mathrm{atten}_d(m,n) &= A_d(m,n)\,\cos\big(\theta_d(m-n)+\varphi_d(m,n)\big) \\ &\propto\ \cos\big(\theta_d(m-n)+\varphi_d(m,n)\big). \end{aligned} \tag{25}$$

In particular, for fixed content, $\mathrm{atten}_d(m,n)$ is a periodic function of the relative distance $(m-n)$ with period $2\pi/\theta_d$.

Proof. Let $q_{m,d}, k_{n,d} \in \mathbb{C}$ be the complex-valued representations of the d-th 2D subspace for the query and key at positions m and n, respectively, as defined above. Applying RoPE multiplies these components by phase factors $e^{im\theta_d}$ and $e^{in\theta_d}$, yielding

$$q'_{m,d} = q_{m,d}e^{im\theta_d}, \qquad k'_{n,d} = k_{n,d}e^{in\theta_d}. \tag{26}$$

The contribution of this subspace to the dot-product attention is the real part of $q'_{m,d}(k'_{n,d})^*$:

$$\mathrm{atten}_d(m,n) = \mathrm{Re}\big[q'_{m,d}(k'_{n,d})^*\big] = \mathrm{Re}\big[q_{m,d}k^*_{n,d}e^{i\theta_d(m-n)}\big]. \tag{27}$$

Writing $q_{m,d}k^*_{n,d}$ in polar form as $q_{m,d}k^*_{n,d} = A_d(m,n)e^{i\varphi_d(m,n)}$ with $A_d(m,n) \geq 0$ and $\varphi_d(m,n) \in \mathbb{R}$, we obtain

$$\begin{aligned} \mathrm{atten}_d(m,n) &= \mathrm{Re}\big[A_d(m,n)e^{i(\theta_d(m-n)+\varphi_d(m,n))}\big] \\ &= A_d(m,n)\,\cos\big(\theta_d(m-n)+\varphi_d(m,n)\big), \end{aligned} \tag{28}$$

which shows that $\mathrm{atten}_d(m,n)$ is proportional to a cosine function of the relative distance $(m-n)$, with angular frequency θ_d. The periodicity with period $2\pi/\theta_d$ follows immediately from the periodicity of the cosine function. □

H.4.2 Details of DRoPE

The initial RoPE frequencies θ_d typically exhibit a wide numerical range. For instance, in our setting $\theta_0 = 1.0$ ($d = 0$), while $\theta_{31} \approx 1.33 \times 10^{-4}$ ($d = 31$, corresponding to $D_h/2 - 1$ and $D_h = 64$). Directly applying affine modulation to these values could lead to numerical instability or disproportionate adjustments due to the vast scale differences.

To address this and ensure stable and effective modulation across the entire range of base frequencies, our DRoPE performs the adaptation in log-space. The layer-specific modulation parameters, $\gamma_d^{(l)}$ and $\beta_d^{(l)}$, predicted by the Multilayer Perceptron (MLP) for layer l following Algorithm 1, are applied to the log-transformed base frequencies $\log\theta_d$ via an element-wise affine transformation as

$$\log\theta'^{(l)}_{\text{inst},d} = \gamma_d^{(l)} \cdot \log\theta_d + \beta_d^{(l)}, \tag{29}$$

where $\log\theta'^{(l)}_{\text{inst},d}$ represents the modulated log-frequencies for layer l and dimension pair d. Then, these modulated log-frequencies are transformed back to their original scale by exponentiation to obtain the adaptive rotation frequencies $\theta'^{(l)}_{\text{inst},d}$ used in the DRoPE calculation for layer l as

$$\theta'^{(l)}_{\text{inst},d} = \exp(\log\theta'^{(l)}_{\text{inst},d}). \tag{30}$$

Algorithm 1 Generating Instance Adaptive Parameters ($\gamma^{(l)}, \beta^{(l)}$)

Require: Time series instance $X \in \mathbb{R}^{B\times L}$, mask $M \in \{0,1\}^{B\times L}$, batch size B, sequence length L, FFT feature dimension ω, MLP network f_{MLP}
Ensure: Adaptive parameters $\gamma^{(l)} \in \mathbb{R}^{B\times D_h/2}$, $\beta^{(l)} \in \mathbb{R}^{B\times D_h/2}$

$X_{\text{masked}} \leftarrow X \odot M$ {Apply mask (element-wise product)}
$F_{\text{result}} \leftarrow \text{RFFT}(X_{\text{masked}}), F_{\text{result}} \in \mathbb{C}^{B\times(\frac{L}{2}+1)}$ {Real FFT along sequence dim}
$F_{\text{amp}} \leftarrow |F_{\text{result}}|, F_{\text{amp}} \in \mathbb{R}^{B\times(\frac{L}{2}+1)}$ {Amplitude spectrum}
$X_{\text{FFT}} \leftarrow F_{\text{amp}}[\ldots,:\omega], X_{\text{FFT}} \in \mathbb{R}^{B\times\omega}$ {Extract low frequency}
$X'_{\text{FFT}} \leftarrow \text{LayerNorm}_{\text{FFT}}(X_{\text{FFT}})$ {Normalize FFT features}
$\gamma^{(l)}, \beta^{(l)} \leftarrow f_{\text{MLP}}(X'_{\text{FFT}})$ {Get instance-specific parameters}
return $\gamma^{(l)}, \beta^{(l)}$

H.4.3 Interpretation of DRoPE

From the theorem H.1, the base RoPE parameters $\{\theta_d\}_{d=0}^{D_h/2-1}$ define a fixed bank of sinusoidal kernels over relative positions. Each θ_d controls the frequency of one cosine kernel. This fixed frequencies works reasonably well when sequences share a homogeneous notion of temporal scale, but it can become suboptimal for a TSFM that must handle highly heterogeneous sampling intervals and spectral patterns across domains.

DRoPE addresses this by making these frequencies instance-adaptive. The FFT-based module in Algorithm 1 extracts simple spectral statistics X'_{FFT} from each input sequence and maps them, via an MLP, to instance- and layer-specific parameters $\gamma_d^{(l)}$ and $\beta_d^{(l)}$. These parameters modulate the base frequencies in log-space:

$$\log \theta'^{(l)}_{\text{inst},d} = \gamma_d^{(l)} \cdot \log \theta_d + \beta_d^{(l)}, \quad \theta'^{(l)}_{\text{inst},d} = \exp\big(\log \theta'^{(l)}_{\text{inst},d}\big), \tag{31}$$

yielding an adapted frequency grid $\{\theta'^{(l)}_d\}$ for each layer and sequence.

Intuitively, this log-space affine transformation stretches or compresses the original RoPE frequency grid in a sequence-dependent way, while preserving the relative-position nature of RoPE. The resulting $\theta'^{(l)}_{\text{inst},d}$ still define sinusoidal kernels over relative lags, but their effective frequencies are now gently steered by the spectrum of the current input. Crucially, DRoPE does not aim to recover a single true period for each series; real-world time series are often multi-periodic and non-stationary. Instead, we interpret $\{\theta'^{(l)}_{\text{inst},d}\}$ as a sequence-dependent frequency profile that shapes how attention depends on relative lags, providing a more flexible and data-driven positional bias than fixed RoPE.

I Additional Details of Experiment Setting

I.1 Pre-training Datasets

We trained KAIROS on the Predictability-Stratified Time Series (PreSTS) corpus, which consists of over 300B real-world time series observations from Chronos [4] and Moirai [40] in conjunction with 15B synthetic time points. Following [11], the training loader samples 80% real data and 20% synthetic data.

Real-world data. The real-world datasets were stratified into five tiers based on their predictability. This hierarchical structure dictates the sampling probability during model training, assigning a higher likelihood of selection to datasets with greater predictability. Such a strategy ensures that the model is preferentially trained on high-quality data while preserving its capacity to predict corner cases. Specifically, Tier 1 comprises datasets characterized by pronounced periodicity and trends with low noise. Tier 2 contains datasets with similarly distinct patterns but high noise, whereas Tier 3 includes those with subtle trends and considerable noise. The remaining datasets were classified into Tiers 4 and 5, based on a composite assessment of their size and pattern regularity. The specific details of these datasets, categorized by their respective sampling frequencies, are presented in Tables 10-13.

Synthetic data. We build a synthetic data generator that produces two distinct types of time series, each with a length of 4096. The first type consists of composite series, created by the additive combination of seasonal, trend, and noise components. For seasonality, we sample one or two components, where the primary period is drawn from {24, 48, 288, 360}, and a potential second period is a fixed seven-fold multiple of the first; the seasonal patterns manifest as either spike trains or smooth non-sinusoidal templates generated via interpolation. An optional trend is chosen from linear, exponential, or an ARIMA-like process derived from cumulating a stationary ARMA model's output. High-probability white Gaussian noise is also added, with each series guaranteed to contain at least one seasonal or trend component and all magnitudes bounded for stability. The generation process for these composite series is formally detailed in Algorithm 2. Complementing these are idealized industrial signals, which simulate perfectly regular machine cycles. These feature a constant baseline with repeating events like trapezoidal spikes or inverted-U shaped dips, where the period, amplitude, and width of the events remain fixed across the entire series with no random jitter, as outlined in Algorithm 3. Synthetic dataset cases are provided in Appendix L.1.

Distribution Analysis. To investigate whether the synthetic dataset supplements distributions absent in the real-world data, we adopted the statistical methodology of Toto [10]. We employed the ARCH-LM Statistic and Spectral Entropy to quantify time-varying volatility and information density (complexity), respectively. These metrics align closely with the design motivation for the Mixture-of-Size Encoding and DRoPE components of KAIROS. As illustrated in Figures 6 (a) and (b), we observed that a significant portion of the real-world dataset exhibits ARCH-LM Statistic values approaching zero and Spectral Entropy values nearing one, indicating high time-varying volatility and complexity. In contrast, the synthetic dataset exhibits the opposite characteristics, exposing the model during training to time series with more constant volatility and greater regularity. This exposure enhances the model's generalization capabilities. Furthermore, we statistically compared the periodicity of the real-world and synthetic datasets by plotting their respective Cumulative Distribution Functions (CDFs). As shown in Figure 6 (c), the dominant periods in the real-world dataset are heavily concentrated. This can cause the model to overfit to these specific periodicities and fail to learn generalizable periodic patterns. Conversely, the synthetic dataset presents a significantly smoother distribution of periods, enabling the model to generalize effectively across arbitrary periodicities. Consequently, the synthetic dataset effectively mitigates these biases present in the real-world data, thereby exposing the model to a more comprehensive and diverse set of distributions.

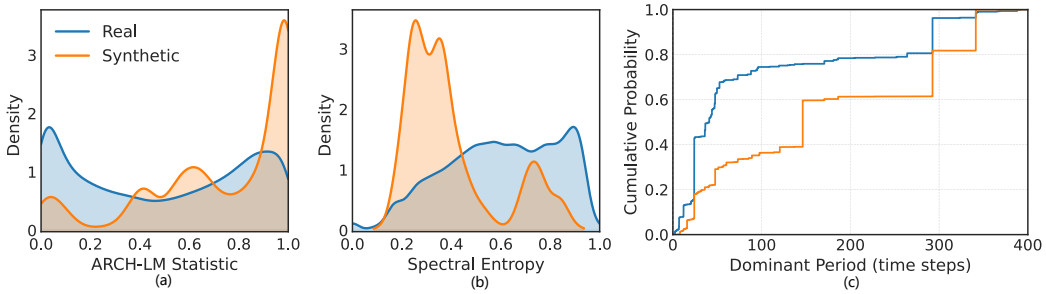


Figure 6: Comparison of real and synthetic datasets across **(a)** ARCH-LM statistic, **(b)** spectral entropy, and **(c)** dominant period distributions, illustrating complementary volatility, complexity, and periodicity characteristics.

I.2 Compared Methods

We compare KAIROS with several state-of-the-art models, including TSFMs such as Sundial [28], Toto [10], YingLong [38], VisionTS [7], Time-MoE [34], ChronosBolt [4], TTM [13], Moirai [40], Timer-XL [27], TimesFM-2.0 [11], Chronos [4], and advanced full-shot deep learning models, including PatchTST [29], DLinear [42], iTransformer [26], Pathformer [8], and TimesNet [41].

I.3 Evaluation Datasets

We select datasets from diverse domains and with varying sampling frequencies as evaluation datasets. The details are summarized in Table 14. For the evaluation on the TSLib benchmark, the datasets were split into training, validation, and test sets. The split for the ETT and Weather datasets follows the configuration adopted by iTransformer [26]. All other datasets use a 70%/10%/20% ratio for the training, validation, and test sets, respectively.

I.4 Evaluation Length Selection

In this section, we provide further details regarding the selection of historical sequence lengths for the various models evaluated in the TSLib benchmark, supplementing the discussion in Section 4.1.

To accommodate diverse application scenarios, an increasing number of TSFMs [28, 11, 27, 40] have devoted attention to predicting over long contexts. Consequently, we evaluate KAIROS and other TSFMs under a long-context setting. Specifically, we adopt a context length of 2048 time steps and examine four prediction horizons, which are {96,192,336,720}. For TSFMs incapable of processing this context length, we instead employ the context length at which each model achieves its best performance.

Table 10: Detailed descriptions of second-level, minute-level, and hourly datasets.

Dataset	Domain	Frequency	# Time Series	# Time points
Wind Power	Energy	4S	1	7,397,147
Residential Load Power	Energy	T	813	437,983,677
Residential PV Power	Energy	T	699	376,016,850
Los-Loop	Transport	5T	207	7,094,304
PEMS03	Transport	5T	358	9,382,464
PEMS04	Transport	5T	921	15,649,632
PEMS07	Transport	5T	883	24,921,792
PEMS08	Transport	5T	510	9,106,560
PEMS Bay	Transport	5T	325	16,941,600
Alibaba Cluster Trace 2018	CloudOps	5T	116,818	190,385,060
Azure VM Traces 2017	CloudOps	5T	159,472	885,522,908
Borg Cluster Data 2011	CloudOps	5T	286,772	1,075,105,708
LargeST	Transport	5T	42,333	4,452,510,528
KDD Cup 2022	Energy	10T	134	4,727,519
HZMetro	Transport	15T	160	380,320
Q-Traffic	Transport	15T	45,148	264,386,688
SHMetro	Transport	15T	576	5,073,984
Beijing Subway	Transport	30T	552	867,744
Elecdemand	Energy	30T	1	17,520
Australian Electricity Demand	Energy	30T	5	1,155,264
London Smart Meters	Energy	30T	5,560	166,528,896
Taxi	Transport	30T	2428	3,589,798
BDG-2 Bear	Energy	H	91	1,482,312
BDG-2 Fox	Energy	H	135	2,324,568
BDG-2 Panther	Energy	H	105	919,800
BDG-2 Rat	Energy	H	280	4,728,288
Borealis	Energy	H	15	83,269
BDG-2 Bull	Energy	H	41	719,304
China Air Quality	Nature	H	2,622	34,435,404
BDG-2 Cockatoo	Energy	H	1	17,544
Covid19 Energy	Energy	H	1	31,912
ELF	Energy	H	1	21,792
GEF12	Energy	H	20	788,280
GEF14	Energy	H	1	17,520
GEF17	Energy	H	8	140,352
BDG-2 Hog	Energy	H	24	421,056
IDEAL	Energy	H	217	1,255,253
Low Carbon London	Energy	H	713	9,543,553
Oikolab Weather	Climate	H	8	800,456
PDB	Energy	H	1	17,520
Sceaux	Energy	H	1	34,223
SMART	Energy	H	5	95,709
Spanish Energy and Weather	Energy	H	1	35,064
ERCOT Load	Energy	H	8	1,238,976
Mexico City Bikes	Transport	H	494	38,687,004
Beijing Air Quality	Nature	H	132	4,628,448
Pedestrian Counts	Transport	H	66	3,132,346
Rideshare	Transport	H	2,304	859,392
Traffic	Transport	H	862	15,122,928
Taxi (Hourly)	Transport	H	2,428	1,794,292
Uber TLC (Hourly)	Transport	H	262	1,138,128
Wind Farms (Hourly)	Energy	H	337	2,869,414
Weatherbench (Hourly)	Nature	H	225,280	78,992,150,528
Buildings900K	Energy	H	1,795,256	15,728,237,816
ERA5	Climate	H	11,059,200	96,613,171,200
CMIP6	Climate	6H	14,327,808	104,592,998,400

Table 11: Detailed descriptions of daily datasets.

Dataset	Domain	Frequency	# Time Series	# Time points
Bitcoin	Econ/Fin	D	18	81,918
Covid Mobility	Transport	D	362	148,602
Extended Web Traffic	Web	D	145,063	370,926,091
Favorita Sales	Sales	D	111,840	139,179,538
Favorita Transactions	Sales	D	54	84,408
Subseasonal	Climate	D	3,448	56,788,560
Subseasonal Precipitation	Climate	D	862	9,760,426
Sunspot	Nature	D	1	73,894
Vehicle Trips	Transport	D	329	32,512
Wiki-Rolling	Web	D	47,675	40,619,100
Dominick	Retail	D	100,014	29,652,492
M5	Sales	D	30,490	47,649,940
Monash Weather	Climate	D	3,010	43,032,000
NN5 Daily	Econ/Fin	D	111	87,801
Uber TLC Daily	Transport	D	262	47,422
Weatherbench (Daily)	Nature	D	225,280	3,291,336,704
Wiki Daily (100k)	Web	D	100,000	274,100,000
Wind Farms (Daily)	Energy	D	337	119,549
Exchange Rate	Finance	D	8	60,704

Table 12: Detailed descriptions of weekly datasets.

Dataset	Domain	Frequency	# Time Series	# Time points
CDC Fluview ILINet	Healthcare	W	375	319,515
CDC Fluview WHO NREVSS	Healthcare	W	296	167,040
Kaggle Web Traffic Weekly	Web	W	145,063	16,537,182
Project Tycho	Healthcare	W	1,258	1,377,707
Traffic Weekly	Transport	W	862	82,752
NN5 Weekly	Econ/Fin	W	111	12,543
Weatherbench (Weekly)	Nature	W	225,280	470,159,360

Table 13: Detailed descriptions of monthly, quarterly, and yearly datasets.

Dataset	Domain	Frequency	# Time Series	# Time points
GoDaddy	Econ/Fin	M	6,270	257,070
CIF 2016	Econ/Fin	M	72	7,108
FRED MD	Econ/Fin	M	107	77,896
M1 Monthly	Econ/Fin	M	617	55,998
M3 Monthly	Econ/Fin	M	1,428	167,562
Tourism Monthly	Econ/Fin	M	366	109,280
M3 Other	Econ/Fin	Q	174	11,933
M1 Quarterly	Econ/Fin	Q	203	9,944
M3 Quarterly	Econ/Fin	Q	756	37,004
Tourism Quarterly	Econ/Fin	Q	427	42,544
M1 Yearly	Econ/Fin	Y	181	4,515
M3 Yearly	Econ/Fin	Y	645	18,319
Tourism Yearly	Econ/Fin	Y	518	12,757

Table 14: Detailed descriptions of evaluation datasets.

Dataset	Domain	Frequency	# Time Series	# Target	# Time points
ETTh1	Energy	H	1	7	17,420
ETTh2	Energy	H	1	7	17,420
ETTm1	Energy	15T	1	7	69,680
ETTm2	Energy	15T	1	7	69,680
Weather	Nature	10T	1	21	52,696
Saugeen (D)	Nature	D	1	1	23,741
Saugeen (W)	Nature	W	1	1	3,391

Algorithm 2 Composite Time Series Generation

Require: Time series length $L = 4096$, Primary period set $\mathcal{P}_1 = \{24, 48, 288, 360\}$, Harmonic multiplier $n = 7$, Trend types $\mathcal{T}_{\text{types}} = \{\text{linear, exp, ARMA}\}$, Seasonal patterns $\mathcal{S}_{\text{patterns}} = \{\text{spike, interpolated segment}\}$.
Ensure: A synthetic time series $\boldsymbol{x}_{1:L}$.

$\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{n} \leftarrow \mathbf{0}_{1:L}$ {Initialize total series and components (seasonal, trend, noise)}
Sample flags f_s, f_t, f_n {Determine component inclusion, ensuring $f_s \vee f_t$ is true}
if f_s is true **then**
  $k \sim \text{Bernoulli}(0.2)$ {$k = 1$ for double period (20% prob), $k = 0$ for single}
  $p_1 \sim \mathcal{U}(\mathcal{P}_1)$ {Sample primary period}
  $\mathcal{P}_{\text{active}} \leftarrow \{p_1\}$
  if $k = 1$ **then**
    $\mathcal{P}_{\text{active}} \leftarrow \mathcal{P}_{\text{active}} \cup \{n \cdot p_1\}$
  end if
  for all p in $\mathcal{P}_{\text{active}}$ **do**
    $a \sim \mathcal{U}(1.0, 3.0)$ {Sample amplitude for this component}
    $pattern \sim \mathcal{U}(\mathcal{S}_{\text{patterns}})$
    $\boldsymbol{c} \leftarrow \text{GeneratePattern}(pattern, p, a)$ {Create a single cycle of the pattern}
    $\boldsymbol{s} \leftarrow \boldsymbol{s} + \text{Tile}(\boldsymbol{c}, L)$ {Tile the cycle to length L and add to seasonal component}
  end for
end if
if f_t is true **then**
  $type \sim \mathcal{U}(\mathcal{T}_{\text{types}})$
  $\boldsymbol{t} \leftarrow \text{GenerateTrend}(type, L)$
  if f_s is true **then**
    $\lambda \sim \mathcal{U}(0.1, 0.3)$ {Reduce trend strength when seasonality is present}
    $\boldsymbol{t} \leftarrow \lambda \cdot \boldsymbol{t}$
  end if
end if
if f_n is true **then**
  $\sigma_n \sim \mathcal{U}(0.01, 0.1)$
  $\boldsymbol{n} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$ {Generate white Gaussian noise}
end if
$\boldsymbol{x} \leftarrow \boldsymbol{s} + \boldsymbol{t} + \boldsymbol{n}$
return $\boldsymbol{x}_{1:L}$

Algorithm 3 Idealized Industrial Signal Generation

Require: Time series length $L = 4096$, Pattern types $\mathcal{P}_{\text{types}} = \{\text{inverted\_u, spikes}\}$.
Ensure: A synthetic time series $\boldsymbol{x}_{1:L}$.

$type \sim \mathcal{U}(\mathcal{P}_{\text{types}})$
$(b, p, a, w, \sigma_n) \leftarrow \text{SampleParams}(type)$ {Sample baseline, period, amplitude, width, noise}
$\boldsymbol{x}_{1:L} \leftarrow b$ {Initialize series with baseline value}
$\boldsymbol{e} \leftarrow \text{TrapezoidShape}(w, a)$ {Create the event shape of width w and amplitude a}
if $type =$ inverted_u **then**
  $sign \leftarrow -1$
else
  $sign \leftarrow +1$
end if
for all $i \leftarrow 0, p, 2p, \ldots$ up to $L - 1$ **do**
  $start \leftarrow i$, $end \leftarrow \min(i + w, L)$
  $\boldsymbol{x}_{start:end} \leftarrow \boldsymbol{x}_{start:end} + sign \cdot \boldsymbol{e}_{1:end-start}$ {Add or subtract event shape at periodic intervals}
end for
if $\sigma_n > 0$ **then**
  $\boldsymbol{x} \leftarrow \boldsymbol{x} + \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$ {Add global Gaussian noise}
end if
return $\boldsymbol{x}_{1:L}$

For the full-shot deep learning baselines, **a hyperparameter search was conducted independently for each dataset to ensure a fair evaluation**. We compared the performance of each model using the context length from its original publication against a length of 2048, selecting the superior of the two. To be specific, lengths of 96 and 2048 were compared for DLinear [42], iTransformer [26], TimesNet [41], and Pathformer, while lengths of 336, 512, and 2048 were compared for PatchTST [29].

The definitive context lengths adopted for each model are systematically tabulated in Table 15.

Table 15: Context Lengths for Models on the TSLib Benchmark.

Method	DLinear	iTrans.	TimesNet	PatchTST	Path.	Chronos	Moirai	TimesFM-2.0	Timer-XL	TTM_a	ChronosBolt	KAIROS
Context length	{96, 2048}	{96, 2048}	{96, 2048}	{336, 512, 2048}	{96, 2048}	512	2048	2048	2048	1536	2048	2048

I.5 Details of DRoPE Analysis

In this section, we explain in more detail the setting of the DRoPE analysis experiment discussed in Section 4.4.2. To more definitively ascertain whether these instance-specific modulations truly capture and leverage beneficial positional information derived from an instance's FFT features, we designed a series of "shuffle" experiments. These experiments function as a form of causal intervention analysis. The underlying hypothesis is: if the instance-derived θ_{inst} modulations are crucial for DRoPE's performance, then disrupting this linkage by applying θ_{inst} modulations from one instance to another (shuffling) should lead to a noticeable degradation in forecasting accuracy.

The details of each group in the experiment are as follows:

- DRoPE: Our proposed method, where each time series instance utilizes its own FFT-derived features to independently modulate its RoPE θ_{inst} parameters at each layer.
- Intra-Dataset Shuffle: During inference, the learned layer-specific θ_{inst} modulation parameters $(\gamma^{(l)}, \beta^{(l)})$ are randomly permuted among instances within the same batch and originating from the same dataset. The modulated θ_{inst} are always shuffled in the same layer.
- Inter-Dataset Shuffle: For instances of a target dataset, layer-specific θ_{inst} modulation parameters $(\gamma^{(l)}, \beta^{(l)})$ are randomly sampled from a pre-computed collection derived from a different source dataset and applied to instances of the target dataset. This process also ensures layer-wise correspondence of the applied modulations.
- Fixed RoPE: To evaluate the architectural contribution of our module, this setting serves as a structural baseline trained from scratch. We replace the DRoPE module with standard RoPE, entirely removing the parameter-predicting MLP. All instances use the static, predefined RoPE θ across all layers during both training and inference.

By comparing DRoPE's performance against the inference-time causal interventions (shuffled configurations) and the fixed θ baseline, we can definitively attribute the performance gains to the effectiveness of the instance-adaptive θ_{inst} modulation process.

J Full Evaluation Results

In this section, we present the detailed results in Section 4.2. Table 16 presents the full results of the zero-shot forecasting experiments conducted at each forecast horizon. For Time-MoE, we report the results as presented in the original paper [34].

K Calculation of information density

In this section, we present the formula for the information density illustrated in Figure 1(b). To effectively characterize and compare different time series datasets, we introduce a method to quantify their information density and the variation of this density over time. For this, similar to TimeMixer [37], we utilize spectral entropy, a metric that measures the complexity and compressibility of a signal. In this context, a signal with low spectral entropy, such as one with a few dominant periodic components, is considered to have low information density due to its redundant and predictable nature. Conversely,

Table 16: Full results of zero-shot forecasting experiments. Best results are highlighted in **bold**, and second best results are underlined. For fair comparison, the full-shot model is excluded from the ranking and is presented for reference only. The dash (-) indicates that the dataset was utilized by the model during its pretraining phase.

Method	Metric	ETTh1					ETTh2					ETTm1					ETTm2					Weather					Saugeen(D)					Saugeen(W)				**AVG**	Total 1st
		96	192	336	720	AVG	96	192	336	720	AVG	96	192	336	720	AVG	96	192	336	720	AVG	96	192	336	720	AVG	96	192	336	720	AVG	96	192	336	AVG		
DLinear	MSE	0.377	0.416	0.446	0.482	0.430	0.280	0.398	0.505	0.796	0.495	0.312	0.339	0.364	0.401	0.354	0.174	0.228	0.287	0.396	0.271	0.174	0.214	0.256	0.309	0.238	0.829	0.832	0.826	0.858	0.836	0.820	0.849	0.781	0.817	0.480	-
	MAE	0.391	0.416	0.440	0.495	0.436	0.358	0.434	0.495	0.634	0.480	0.359	0.375	0.392	0.415	0.385	0.272	0.312	0.355	0.428	0.342	0.238	0.272	0.304	0.346	0.290	0.502	0.503	0.506	0.513	0.506	0.569	0.571	0.551	0.564	0.424	
iTransformer	MSE	0.383	0.441	0.478	0.470	0.443	0.282	0.367	0.402	0.406	0.364	0.327	0.363	0.390	0.424	0.376	0.336	0.369	0.398	0.428	0.383	0.171	0.218	0.278	0.357	0.256	0.778	0.795	0.800	0.839	0.803	0.988	1.041	1.007	1.012	0.501	-
	MAE	0.396	0.429	0.449	0.469	0.436	0.344	0.392	0.422	0.433	0.398	0.355	0.381	0.411	0.442	0.397	0.355	0.381	0.411	0.442	0.397	0.210	0.256	0.299	0.351	0.279	0.457	0.463	0.482	0.490	0.473	0.626	0.640	0.642	0.636	0.423	
TimesNet	MSE	0.408	0.456	0.508	0.567	0.485	0.315	0.435	0.481	0.455	0.422	0.358	0.428	0.494	0.550	0.458	0.175	0.251	0.307	0.412	0.286	0.174	0.229	0.282	0.360	0.261	1.177	1.268	1.129	1.136	1.178	0.817	0.777	0.751	0.782	0.544	-
	MAE	0.426	0.451	0.496	0.521	0.469	0.359	0.422	0.456	0.461	0.425	0.373	0.412	0.447	0.480	0.428	0.255	0.305	0.343	0.407	0.328	0.222	0.265	0.304	0.352	0.286	0.546	0.561	0.534	0.531	0.543	0.517	0.494	0.501	0.504	0.423	
PatchTST	MSE	0.382	0.416	0.443	0.466	0.427	0.280	0.351	0.388	0.424	0.361	0.278	0.327	0.363	0.414	0.346	0.162	0.221	0.270	0.369	0.256	0.149	0.195	0.254	0.346	0.236	0.827	0.83	0.819	0.848	0.831	0.849	0.847	0.781	0.826	0.456	-
	MAE	0.403	0.426	0.444	0.474	0.437	0.349	0.393	0.420	0.445	0.402	0.330	0.361	0.388	0.423	0.376	0.249	0.291	0.323	0.386	0.312	0.205	0.248	0.293	0.354	0.275	0.474	0.482	0.488	0.491	0.484	0.545	0.526	0.500	0.524	0.397	
Pathformer	MSE	0.405	0.466	0.506	0.494	0.468	0.276	0.365	0.420	0.429	0.373	0.302	0.356	0.395	0.463	0.379	0.162	0.225	0.290	0.414	0.273	0.152	0.200	0.258	0.341	0.238	1.168	1.266	1.142	1.149	1.181	0.932	0.949	0.877	0.919	0.533	-
	MAE	0.398	0.427	0.447	0.463	0.434	0.334	0.386	0.428	0.443	0.398	0.325	0.361	0.388	0.429	0.376	0.244	0.288	0.333	0.403	0.317	0.192	0.237	0.281	0.335	0.261	0.501	0.535	0.508	0.499	0.511	0.532	0.532	0.533	0.532	0.399	
Chronos_s	MSE	0.516	0.599	0.613	0.603	0.583	0.304	0.407	0.427	0.450	0.397	0.504	0.601	0.667	0.728	0.625	0.207	0.277	0.343	0.468	0.324	0.209	0.267	0.328	0.397	0.300	1.108	1.142	1.105	1.143	1.125	1.183	1.227	1.165	1.192	0.629	0
	MAE	0.416	0.464	0.487	0.512	0.470	0.347	0.401	0.428	0.457	0.408	0.398	0.457	0.498	0.538	0.473	0.277	0.323	0.364	0.437	0.350	0.213	0.265	0.307	0.356	0.285	0.477	0.503	0.506	0.512	0.500	0.520	0.524	0.526	0.523	0.426	
Chronos_b	MSE	0.526	0.585	0.593	0.608	0.578	0.322	0.411	0.440	0.443	0.404	0.422	0.523	0.600	0.686	0.558	0.206	0.272	0.329	0.436	0.311	0.184	0.241	0.296	0.375	0.274	1.172	1.224	1.202	1.249	1.212	1.166	1.222	1.157	1.182	0.626	0
	MAE	0.413	0.447	0.461	0.491	0.453	0.343	0.392	0.422	0.440	0.399	0.374	0.434	0.477	0.520	0.451	0.270	0.316	0.354	0.420	0.340	0.206	0.256	0.298	0.353	0.278	0.473	0.505	0.502	0.505	0.496	0.515	0.517	0.508	0.513	0.415	
Chronos_l	MSE	0.493	0.601	0.636	0.653	0.596	0.328	0.434	0.480	0.522	0.442	0.452	0.536	0.613	0.671	0.568	0.203	0.268	0.316	0.424	0.303	0.195	0.254	0.311	0.395	0.289	1.178	1.229	1.194	1.232	1.208	1.213	1.269	1.241	1.241	0.642	0
	MAE	0.400	0.449	0.475	0.503	0.457	0.348	0.407	0.448	0.482	0.421	0.379	0.430	0.470	0.512	0.448	0.263	0.309	0.342	0.408	0.331	0.211	0.264	0.305	0.360	0.285	0.481	0.508	0.499	0.499	0.497	0.523	0.534	0.544	0.534	0.420	
Moirai_s	MSE	0.395	0.430	0.445	0.430	0.425	0.274	0.341	0.364	0.378	0.339	0.362	0.385	0.410	0.450	0.402	0.192	0.258	0.307	0.391	0.287	0.160	0.207	0.258	**0.327**	0.238	1.082	1.112	1.085	1.105	1.096	1.272	1.284	1.177	1.244	0.551	4
	MAE	0.404	0.427	0.436	0.445	0.428	0.324	0.369	**0.386**	**0.405**	0.371	0.360	0.379	0.399	0.422	0.390	0.263	0.308	0.340	0.397	0.327	0.194	0.238	0.275	**0.324**	0.258	0.479	0.493	0.491	0.493	0.489	0.560	0.562	0.535	0.552	0.397	
Moirai_b	MSE	0.429	0.494	0.531	0.544	0.500	0.275	0.347	0.379	0.388	0.347	0.456	0.459	0.482	0.518	0.479	0.194	0.251	0.292	**0.354**	0.273	0.170	0.218	0.270	0.344	0.251	1.129	1.175	1.142	1.177	1.156	1.208	1.243	1.171	1.207	0.579	2
	MAE	0.412	0.447	0.467	0.501	0.457	0.319	0.365	0.388	0.408	0.370	0.385	0.400	0.415	0.435	0.409	0.266	0.306	0.333	**0.375**	0.320	0.203	0.249	0.288	0.339	0.270	0.472	0.495	0.485	0.485	0.484	0.557	0.550	0.538	0.548	0.403	
Moirai_l	MSE	0.464	0.523	0.546	0.549	0.521	0.298	0.403	0.464	0.530	0.424	0.522	0.541	0.563	0.611	0.559	0.195	0.257	0.308	0.395	0.289	0.170	0.222	0.277	0.361	0.258	1.102	1.140	1.102	1.122	1.117	1.213	1.252	1.179	1.215	0.604	0
	MAE	0.433	0.470	0.486	0.511	0.475	0.346	0.412	0.448	0.498	0.426	0.401	0.423	0.442	0.465	0.433	0.267	0.311	0.342	0.395	0.329	0.203	0.253	0.293	0.347	0.274	0.464	0.486	0.484	0.486	0.480	0.541	0.543	0.534	0.539	0.418	
TimesFM-2.0	MSE	0.424	0.444	0.443	0.438	0.437	0.256	0.328	0.366	0.391	0.335	0.362	0.410	0.429	0.470	0.418	0.175	0.239	0.294	0.369	0.269	-	-	-	-	-	1.099	1.106	1.058	1.088	1.088	1.277	1.448	3.971	2.232	-	0
	MAE	0.397	0.417	0.424	0.442	0.420	0.308	0.361	0.389	0.413	0.368	0.340	0.373	0.395	0.424	0.383	0.245	0.294	0.332	0.385	0.314	-	-	-	-	-	0.449	0.483	0.492	0.522	0.487	0.649	0.724	1.362	0.912	-	
Timer-XL	MSE	0.394	0.436	0.469	0.436	0.434	0.273	0.342	0.377	0.382	0.344	0.325	0.374	0.414	0.510	0.406	0.187	0.240	0.285	0.374	0.272	0.167	0.213	0.266	0.340	0.247	1.043	1.126	1.127	1.127	1.106	1.112	1.128	1.043	1.094	0.537	0
	MAE	0.395	0.419	0.437	0.448	0.425	0.336	0.380	0.402	0.416	0.384	0.354	0.385	0.409	0.455	0.401	0.272	0.310	0.338	0.392	0.328	0.221	0.264	0.304	0.354	0.286	0.574	0.612	0.615	0.606	0.602	0.626	0.620	0.594	0.613	0.427	
Time-MoE_b	MSE	0.357	**0.384**	0.411	0.449	0.400	0.305	0.351	0.391	0.419	0.367	0.338	0.353	0.381	0.504	0.394	0.201	0.258	0.324	0.488	0.318	0.160	0.210	0.274	0.418	0.266	-	-	-	-	-	-	-	-	-	-	3
	MAE	**0.381**	**0.404**	0.434	0.477	0.424	0.359	0.386	0.418	0.454	0.404	0.368	0.388	0.413	0.493	0.416	0.291	0.334	0.373	0.464	0.366	0.214	0.260	0.309	0.405	0.297	-	-	-	-	-	-	-	-	-	-	
Time-MoE_l	MSE	**0.350**	0.388	0.411	0.427	**0.394**	0.302	0.364	0.417	0.537	0.405	0.309	0.346	0.373	0.475	0.376	0.197	0.250	0.337	0.480	0.316	0.159	0.215	0.291	0.415	0.270	-	-	-	-	-	-	-	-	-	-	2
	MAE	0.382	0.412	0.430	0.455	0.420	0.354	0.385	0.425	0.496	0.415	0.357	0.381	0.408	0.477	0.406	0.286	0.322	0.375	0.461	0.361	0.213	0.266	0.322	0.400	0.300	-	-	-	-	-	-	-	-	-	-	
TTM_a	MSE	0.373	0.396	**0.406**	**0.411**	0.397	0.262	**0.326**	**0.354**	**0.376**	**0.330**	0.329	0.364	0.381	0.413	0.372	0.167	0.227	0.277	0.360	0.258	0.146	0.193	0.249	0.328	**0.229**	0.929	**0.926**	**0.904**	**0.939**	**0.925**	1.114	1.134	1.051	1.156	0.494	11
	MAE	0.397	0.415	0.424	0.441	0.419	0.334	0.378	0.400	0.421	0.383	0.350	0.374	0.390	0.410	0.381	0.254	0.296	0.329	0.382	0.315	0.197	0.242	0.284	0.336	0.265	0.511	0.518	0.523	0.541	0.523	0.628	0.628	0.608	0.621	0.408	
ChronosBolt_a	MSE	0.405	0.475	0.514	0.516	0.478	**0.255**	0.339	0.388	0.400	0.346	0.309	0.375	0.420	0.512	0.404	0.166	0.231	0.289	0.400	0.272	0.152	0.205	0.264	0.357	0.245	**0.863**	0.956	1.001	1.103	0.981	0.762	0.802	0.749	0.771	0.489	2
	MAE	0.388	0.426	0.453	0.485	0.438	0.307	0.362	0.398	0.418	0.371	0.317	0.357	0.389	0.441	0.376	0.237	0.284	0.324	0.393	0.310	0.186	0.237	0.279	0.338	0.260	0.410	0.458	0.492	0.551	0.478	0.433	0.446	0.473	0.451	0.381	
ChronosBolt_b	MSE	0.420	0.486	0.517	0.493	0.479	**0.255**	0.333	0.381	0.393	0.341	0.303	0.370	0.410	0.497	0.395	0.164	0.233	0.299	0.414	0.278	0.150	0.197	0.255	0.344	0.237	0.905	0.959	0.957	1.049	0.968	**0.752**	**0.791**	**0.725**	**0.756**	0.483	14
	MAE	0.388	0.424	0.445	0.457	0.429	**0.305**	**0.355**	0.389	0.408	**0.364**	**0.311**	**0.352**	0.382	0.428	0.368	**0.232**	**0.280**	0.323	0.391	0.307	0.183	0.230	0.272	0.329	0.254	0.417	0.439	0.441	0.467	0.441	**0.423**	**0.437**	0.462	0.441	0.369	
KAIROS_m	MSE	0.386	0.436	0.460	0.426	0.427	0.257	0.332	0.369	0.382	0.335	0.304	0.358	0.383	0.408	0.363	0.168	0.228	0.279	0.366	0.260	**0.144**	0.193	0.254	0.353	0.236	0.976	0.997	0.982	1.015	0.993	0.789	0.840	0.788	0.806	0.477	4
	MAE	0.384	0.411	0.422	0.425	0.411	0.314	0.364	0.391	0.407	0.369	0.329	0.362	0.384	0.407	0.371	0.246	0.290	0.323	0.378	0.309	**0.181**	0.229	0.273	0.331	0.254	0.424	0.439	0.443	0.450	0.439	0.437	0.444	**0.426**	**0.436**	0.367	
KAIROS_s	MSE	0.385	0.429	0.446	0.422	0.421	0.269	0.340	0.380	0.394	0.346	0.305	0.356	0.381	**0.404**	0.362	**0.163**	0.226	0.278	0.382	0.262	0.151	0.197	0.253	0.340	0.235	0.920	0.943	0.942	1.002	0.952	0.808	0.851	0.815	0.825	0.473	6
	MAE	0.383	0.408	**0.418**	**0.424**	**0.408**	0.320	0.368	0.397	0.416	0.375	0.326	0.360	0.382	**0.404**	0.368	0.238	0.284	0.320	0.384	0.307	0.186	0.232	0.274	0.329	0.255	0.410	0.426	0.429	0.442	0.427	0.447	0.446	0.429	0.441	0.366	
KAIROS_l	MSE	0.396	0.446	0.472	0.434	0.437	0.270	0.348	0.367	**0.376**	0.340	**0.291**	**0.337**	**0.368**	0.408	**0.351**	**0.163**	**0.224**	**0.274**	0.361	**0.255**	0.147	**0.191**	**0.248**	0.333	0.230	0.936	0.972	0.963	1.014	0.971	0.786	0.817	0.766	0.790	**0.471**	**26**
	MAE	0.385	0.414	0.429	0.432	0.415	0.325	0.378	0.395	0.410	0.377	0.321	0.355	**0.378**	0.405	**0.365**	0.239	0.283	**0.317**	0.376	**0.304**	**0.181**	**0.227**	**0.270**	0.326	**0.251**	**0.406**	**0.424**	**0.428**	**0.437**	**0.424**	0.445	0.446	0.429	0.440	**0.365**	

a signal with high spectral entropy, resembling white noise, has its power spread broadly across the frequency spectrum, indicating a high degree of randomness and thus a higher information density.

Given the observations $\mathbf{x}_{1:T} = (x_1, \ldots, x_T) \in \mathbb{R}^T$ of a specific dataset, we analyze the time series using a sliding window approach. The series is segmented into N windows, $\mathbf{w}_i$, each of size M with a step size of S. In our analysis, we use a non-overlapping configuration where both the window size and the step size are set to 128 (i.e., $M = 128$, $S = 128$). The total number of windows is $N = \lfloor (T - M)/S \rfloor + 1$. To mitigate spectral leakage from windowing, we apply a Hann window to each segment $\mathbf{w}_i$:

$$h[n] = 0.5 - 0.5 \cos\left(\frac{2\pi n}{M-1}\right), \quad 0 \leq n \leq M-1. \tag{32}$$

For each resulting windowed segment $\mathbf{w}'_i = \mathbf{w}_i \cdot h$, we compute its normalized power spectral density, $p_i[k]$, which describes how the signal's power is distributed over different frequencies. The spectral entropy for the i-th window, $H_{SE}(\mathbf{w}_i)$, is then calculated using the Shannon entropy formula:

$$H_{SE}(\mathbf{w}_i) = -\sum_{k=0}^{M-1} p_i[k] \log_2(p_i[k]). \tag{33}$$

This process yields a sequence of entropy values, $(H_{SE}(\mathbf{w}_1), \ldots, H_{SE}(\mathbf{w}_N))$, where each value represents the localized information density of its corresponding time segment.

To obtain a holistic view of the dataset's characteristics, we compute two key statistics from this entropy sequence.

First, the mean spectral entropy (μ_{SE}) serves as a measure of the average information density of the entire dataset. A higher μ_{SE} suggests that the dataset, on average, contains more complex and less predictable patterns. This allows for a direct comparison of the overall information content between different datasets.

$$\mu_{SE} = \frac{1}{N} \sum_{i=1}^{N} H_{SE}(\mathbf{w}_i). \tag{34}$$

Second, the standard deviation of spectral entropy (σ_{SE}) quantifies the variability of information density within the dataset. A small σ_{SE} indicates that the dataset is stationary in its complexity, with a consistent level of information density throughout. A large σ_{SE}, however, reveals a non-stationary character, signifying substantial fluctuations in the signal's complexity and information content over time.

$$\sigma_{SE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (H_{SE}(\mathbf{w}_i) - \mu_{SE})^2}. \tag{35}$$

Together, μ_{SE} and σ_{SE} provide a concise yet powerful summary of a dataset's informational characteristics, enabling a quantitative assessment of its average complexity and internal dynamics.

L Showcases

L.1 Showcases of Synthetic Data

Figure 7 showcases representative examples generated by the algorithm using the synthetic dataset. As detailed in Appendix I.1, the generated synthetic dataset is classified into two categories. The first is a composite type, formed from a combination of seasonal, trend, and noise components, which is designated Custom in the figures. The second consists of idealized industrial signals, designated Perfect periodic.

L.2 Showcases of KAIROS

We present several prediction examples generated by KAIROS during testing, as illustrated in Figure 8.

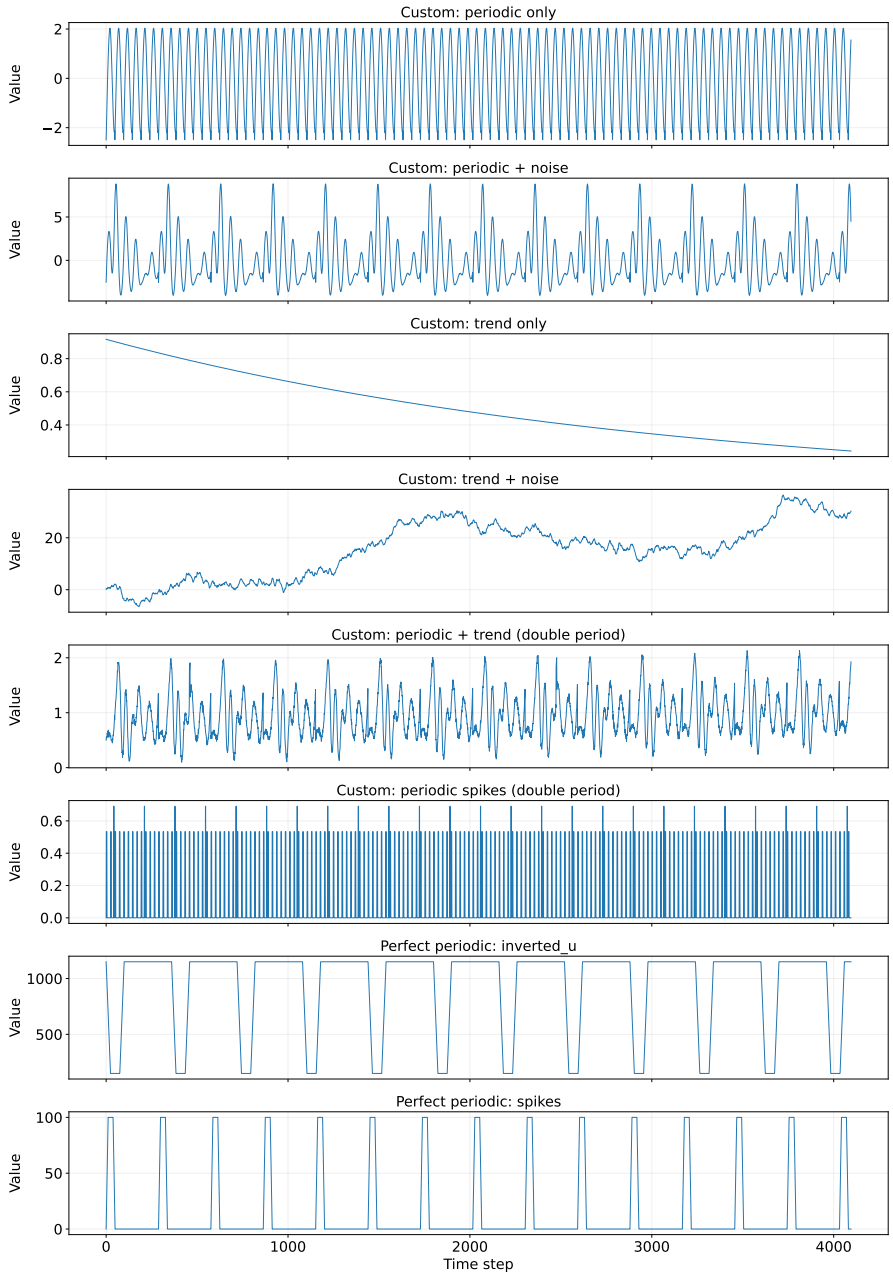
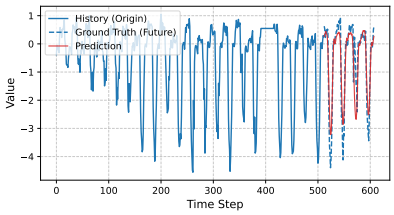
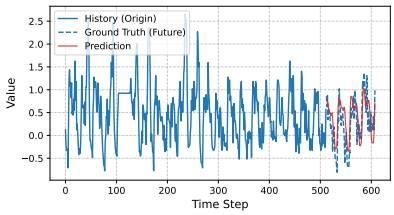


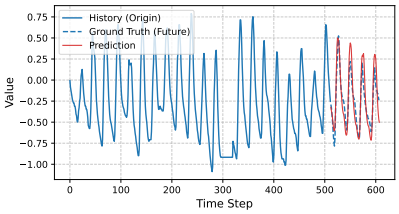
Figure 7: Several distinct case types are generated by the synthetic dataset algorithm. Specifically, the Custom designation refers to a composite signal type, which is systematically constructed by combining seasonal, trend, and noise components. Concurrently, the Perfect periodic designation denotes synthesized, idealized industrial signals.



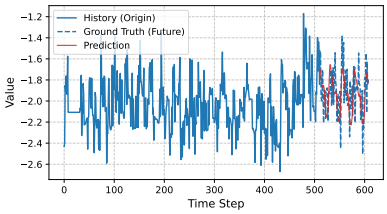
(a) ETTh1-1



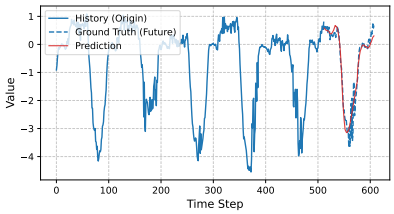
(b) ETTh1-2



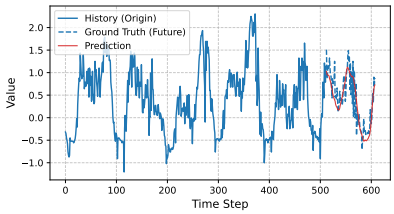
(c) ETTh2-1



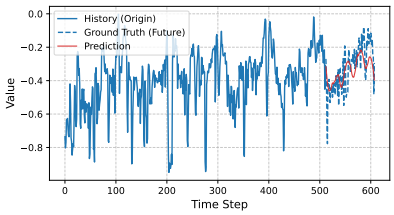
(d) ETTh2-2



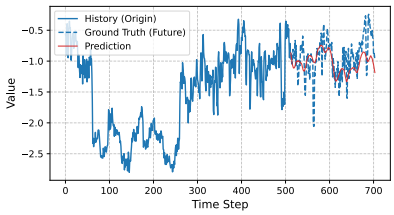
(e) ETTm1-1



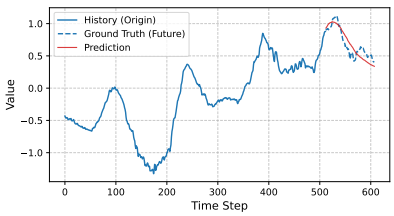
(f) ETTm1-2



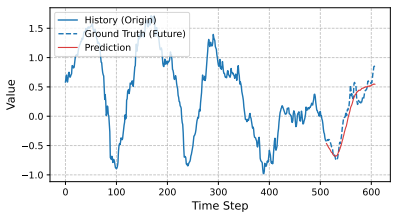
(g) ETTm2-1



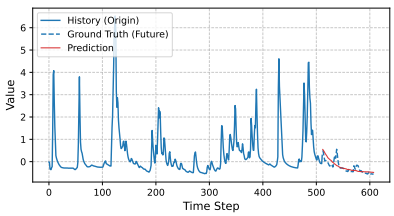
(h) ETTm2-2



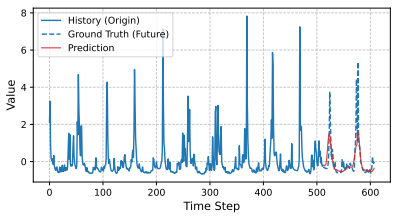
(i) Weather-1



(j) Weather-2



(k) Saugeen-1



(l) Saugeen-2

Figure 8: Example of forecasts from KAIROS$_b$ on the test datasets used in experiments.

M Broader impacts

Our work on KAIROS is foundational research focused on advancing parameter-efficient and adaptive time series modeling. We anticipate several positive societal impacts: (i) promoting sustainable AI by significantly reducing the computational resources and energy required for large-scale deployment; (ii) enhancing decision-making in dynamic environments (e.g., energy, retail) through improved modeling of heterogeneous temporal patterns; and (iii) providing a high-quality data curation methodology via the PreSTS corpus to improve model performance.

While we do not foresee immediate negative societal consequences, we recognize that advanced predictive technologies carry inherent risks of misuse. Consequently, rigorous ethical oversight is therefore essential to guarantee that such predictive capabilities serve the public interest.