HORA: LEARNING A UNIVERSAL 128-WORD VOCABULARY FROM BILLIONS OF CROSS-DOMAIN TIME POINTS FOR ZERO-SHOT FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series foundation models (TSFMs) are pretrained on large cross-domain corpora to forecast unseen series in a zero-shot manner. They inherit many designs from large language models (LLMs). An LLM is built on a vocabulary, but time series have no native vocabulary. This raises a fundamental question: *what should the vocabulary of time series be?* Existing approaches either have no vocabulary, borrow one from language, or learn one that collapses similar sequences. To answer this question, we propose **Hora**, a TSFM built on a **temporal vocabulary** of 128 words learned from billions of cross-domain time points. Hora *reads* each patch as a mixture of the words, whose weights constitute its *temporal grammar*. It *consults* the vocabulary in every layer as a global memory, and *writes* the forecast from the same words. Experiments on GIFT-Eval show that Hora achieves state-of-the-art zero-shot forecasting performance. Visualizations further demonstrate that the interpretable temporal grammar expresses patterns shared across domains. Our code is available at `https://anonymous.4open.science/r/Hora`.

1 INTRODUCTION

Time series forecasting is essential for a wide range of real-world systems and applications (Lim & Zohren, 2021; Benidis et al., 2022). Despite the abundance of time series data across domains, individual datasets often contain limited observations, making it difficult to train generalizable forecasting models from a single source. Inspired by large language models (LLMs), time series foundation models (TSFMs) have rapidly emerged. They are pretrained on large cross-domain corpora to achieve strong zero-shot forecasting performance (Liang et al., 2024; Jin et al., 2026).

TSFMs inherit many designs of LLMs. Notably, an LLM is built on a vocabulary, and every input token is taken from it. Language provides this vocabulary natively, but time series do not. This raises a fundamental question: *what should the vocabulary of time series be?* A vocabulary is a set of words shared by all inputs. It has to decide what it encodes and how different inputs use it. Two properties of time series constrain both. First, values are specific to individual series, while temporal patterns recur across domains. Therefore, a vocabulary of time series should encode patterns rather than values. Second, time series vary continuously, so no vocabulary can hold a dedicated word

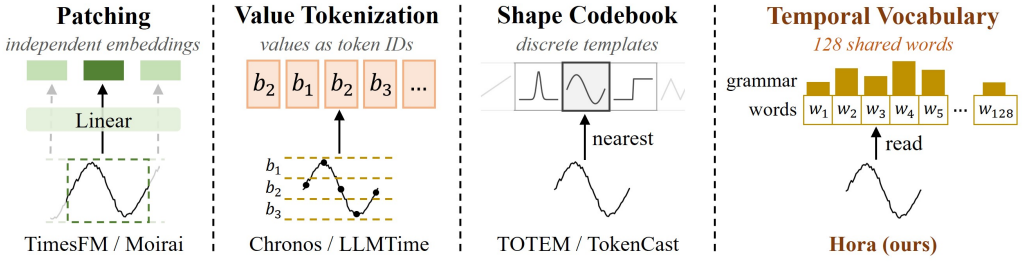


Figure 1: Different approaches to the vocabulary of time series.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

for every possible sequence. Therefore, sequences have to share the same words, and still need to be distinguished, or they would receive the same forecast. As shown in Figure 1, existing methods have explored this question in three ways. (1) Since Nie et al. (2023) proposed "a time series is worth 64 words", patches have become the standard input of most TSFMs (Das et al., 2024; Woo et al., 2024). A patch carries local semantics, but it is taken from one sequence **instead of a shared set**. Therefore, patching produces no vocabulary, and the patterns that recur across domains remain implicit in the model weights. (2) Value tokenization (Ansari et al., 2024; Gruver et al., 2023) borrows the design of language vocabularies, where each value becomes a token ID or a digit string. Such a vocabulary **encodes isolated values rather than patterns**, so the temporal structure formed by neighboring values is lost. (3) A shape codebook (Talukder et al., 2024; Tao et al., 2025) learns its templates from time series through vector quantization. The templates encode temporal patterns and can be reused across inputs. However, each token is replaced by the nearest template. Therefore, no matter how many templates the codebook holds, the tokens replaced by the same one become **indistinguishable**. In summary, patching produces no vocabulary, value tokenization takes one from language, and a shape codebook learns one from time series but collapses similar sequences.

To answer this question, we propose **Hora**, a TSFM built on a **temporal vocabulary** of 128 learnable words. The words are shared across domains, sequences, and positions. They are learned jointly with the model from billions of cross-domain time points. The vocabulary is thus designed to **encode patterns rather than values**. In short, "all time series are worth these 128 words". In detail, Hora *reads* each input patch as a mixture of the words, and the weights of the mixture form its *temporal grammar*, which describes the input in an interpretable way. Different patches share the same words but have different grammars, so they can remain **distinguished**. In the backbone, every layer *consults* the vocabulary, so it serves as a global memory during reasoning. Finally, Hora *writes* the forecast as a composition of the same words. In this way, the vocabulary is leveraged throughout the forecasting process. Our contributions are summarized as follows:

- Conceptually, we study what the vocabulary of time series should be, and propose a novel **128-word temporal vocabulary** learned from billions of cross-domain time points.
- Methodologically, we propose **Hora**, a powerful forecasting TSFM centered on the temporal vocabulary, which is used throughout reading, consulting, and writing. This design enables Hora to forecast any time series with the same 128 words.
- Empirically, we validate Hora on GIFT-Eval, where it achieves state-of-the-art zero-shot forecasting performance. Ablation studies and visualized case analyses further demonstrate the effectiveness of the temporal vocabulary and the interpretability of the temporal grammar.

2 Related Work

2.1 Time Series Forecasting

Early approaches to time series forecasting rely on statistical assumptions or hand-crafted features, including statistical methods (Stock & Watson, 2001; Hyndman & Khandakar, 2008) and machine learning models (Müller et al., 1997; Elsayed et al., 2021). Deep learning later enabled models to learn representations directly from data, with RNNs (Salinas et al., 2020; Lai et al., 2018), CNNs (Liu et al., 2022; Wu et al., 2023), and Transformers (Zhou et al., 2021; Wu et al., 2021; Nie et al., 2023; Liu et al., 2024a) becoming widely used for forecasting. Linear models (Zeng et al., 2023) and state space models (Wang et al., 2025b) have also achieved competitive performance with simpler architectures. *However, these approaches are typically trained for individual datasets, limiting their ability to exploit shared patterns across domains.*

2.2 Time Series Foundation Models

TSFMs are pretrained on large-scale corpora and can forecast unseen series in a zero-shot manner. Early attempts adapt pretrained LLMs to time series forecasting by aligning time series with the input space of language models (Zhou et al., 2023; Jin et al., 2024; Gruver et al., 2023). More recent works primarily focus on pretraining large Transformer-based models from scratch on diverse time series corpora (Das et al., 2024; Liu et al., 2024b; Shi et al., 2025; Liu et al., 2025b; Cohen et al., 2025; Woo et al., 2024; Goswami et al., 2024; Ansari et al., 2024; 2025). These models demonstrate

strong zero-shot forecasting performance across a wide range of datasets and domains, showing the potential of large-scale pretraining to improve the generalization of time series models. Recent works further explore non-Transformer backbones (Auer et al., 2025; Graf et al., 2026; Moroshan et al., 2025; Ekambaram et al., 2024), as well as pretraining with synthetic data (Dooley et al., 2023; Hoo et al., 2025). *However, most of these models take patches as input tokens, so the patterns that recur across domains remain implicit in the model weights.*

2.3 Time Series Tokenization and Bases

Representing time series with reusable units has a long history. Symbolic methods (Lin et al., 2007; Schäfer, 2015) discretize a series into predefined words and describe the series with these words. Recent works adopt tokenization schemes from language models, where values are quantized into bins (Ansari et al., 2024) or converted into digit strings (Gruver et al., 2023). Another line of work learns discrete codebooks from time series through quantization (Wen et al., 2024; Talukder et al., 2024; Tao et al., 2025), where each token corresponds to a single code. Beyond discrete units, several methods express time series as combinations of shared components. N-BEATS (Oreshkin et al., 2020) expresses both the backcast and the forecast as combinations of basis functions, while BasisFormer (Ni et al., 2023) computes coefficients by attending the input to the historical part of learned bases and forecasts with their future part. FlowState (Graf et al., 2026) decodes forecasts as combinations of predefined basis functions. Time-LLM (Jin et al., 2024) reprograms each patch as an attention-weighted combination of text prototypes, which are derived from the word embeddings of a frozen LLM. *However, these components are predefined, learned on a single dataset, or derived from language. In contrast, the words of Hora are learned from time series across domains, and the same words are used to read the input, consulted by every layer, and composed into the forecast.*

3 Methodology

As illustrated in Figure 2, we present **Hora**, a Transformer-based time series foundation model built on a temporal vocabulary of N learnable words shared across domains. The vocabulary is used throughout the forecasting process: we *read* inputs through the vocabulary, *consult* the vocabulary in every Transformer layer, and *write* forecasts from the vocabulary.

Problem definition. Given a time series $X_{1:L} \in \mathbb{R}^L$ over the past L time steps, the task of time series forecasting is to predict the next L_f time steps in the future. Rather than point estimates, Hora generates probabilistic predictions with Q quantile levels:

$$\widehat{X}_{L+1:L+L_f} = F_\theta(X_{1:L}), \quad F_\theta : \mathbb{R}^L \to \mathbb{R}^{L_f \times Q}, \tag{1}$$

where F_θ is a learnable mapping function parameterized by θ. As a foundation model, Hora learns F_θ once via pretraining and can forecast unseen time series without any further tuning.

3.1 Read: Translating Patches with 128 Words

Following previous works (Nie et al., 2023; Ansari et al., 2025), we split the normalized input sequence X into non-overlapping patches of a fixed length P, denoted as $X^p \in \mathbb{R}^{L_p \times P}$, where $L_p = \lceil L/P \rceil$ is the number of patches. When L is not divisible by P, X^p is left-padded with zeros. Missing values are also filled with zeros. Therefore, to distinguish them from real observations, we provide a binary mask $M^p \in \{0,1\}^{L_p \times P}$ alongside the input, where 0 marks a padded or missing value. Each patch is then concatenated with its mask and encoded:

$$H = f^{\text{enc}}(X^p \,\|\, M^p) \in \mathbb{R}^{L_p \times d}, \tag{2}$$

where $\|$ denotes concatenation and d is the hidden dimension. The encoding function f^{enc} is implemented as a multi-layer perceptron (MLP) with a residual connection. We use $H_i \in \mathbb{R}^d$ to denote the encoding of the i-th patch.

As shown in Figure 2(a), rather than feeding H_i into the backbone, Hora translates it by *reading* a ***temporal vocabulary*** $V = [w_1, w_2, \ldots, w_N] \in \mathbb{R}^{N \times d}$, which consists of N learnable words $w_n \in \mathbb{R}^d$, with N set to 128 by default. The words are randomly initialized and learned jointly with the model during pretraining. *Values are specific to individual series, but temporal patterns*

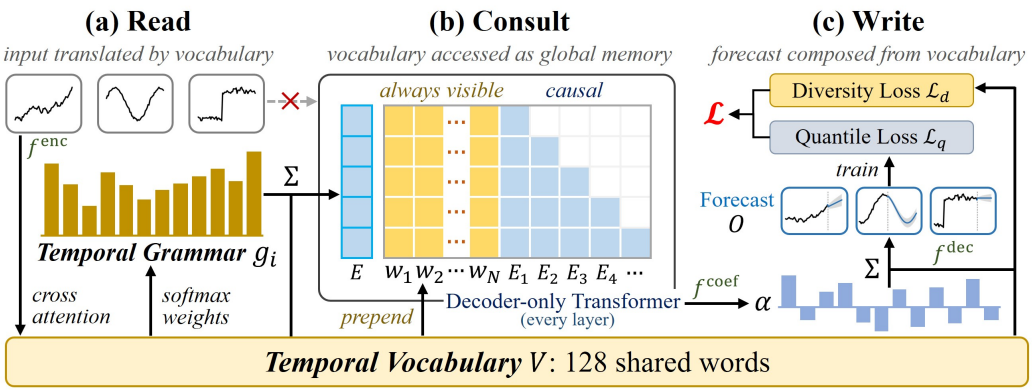


Figure 2: Overall illustration of Hora. The temporal vocabulary is used throughout forecasting.

can recur across domains. The patch encoding H_i is derived from the values of a single patch. It therefore characterizes that patch alone, and it remains implicit whether similar patterns also occur in other domains. In contrast, the vocabulary is independent of any individual input. It is learned from the entire pretraining corpus and shared across domains, sequences, and positions, which makes it **universal**. The vocabulary thus provides an explicit and common set of components for representing temporal structures, where the recurring patterns can be expressed consistently across domains.

Reading is formulated as a cross-attention between the encoded patches and the words:

$$g_i = \text{softmax}\left(\frac{H_i W_Q \left(V W_K\right)^\top}{\sqrt{d}}\right), \quad E_i = g_i \left(V W_V\right), \tag{3}$$

where $W_{\{Q,K,V\}} \in \mathbb{R}^{d\times d}$ are projection matrices, $g_i \in \mathbb{R}^N$ contains the attention weights of the i-th patch over the N words, and $E_i \in \mathbb{R}^d$ is the output vocabulary embedding. Unlike vector quantization (Wen et al., 2024) or sparse coding (Aharon et al., 2006), Hora imposes no hard assignment or sparsity constraint on g_i. How many words a patch uses is thus decided by pretraining rather than by design. As a result, a patch is translated into a mixture of the vocabulary, with g_i specifying each word's contribution. We then define g_i as the ***temporal grammar*** of the i-th patch. Since g_i is non-negative and sums to 1, it forms a distribution over the vocabulary that describes how the patch is expressed by the words. Patches that share similar patterns can therefore have similar grammars, even when they come from different domains. For a whole time series, we define its temporal grammar $g \in \mathbb{R}^N$ as the average over all patches, which summarizes its usage of the vocabulary:

$$g = \frac{1}{L_p}\sum_{i=1}^{L_p} g_i. \tag{4}$$

Reading the vocabulary forms a strict bottleneck in the model. The patch encoding H is not propagated. It is consumed by Equation (3) to produce the temporal grammar of each patch, and the backbone receives the vocabulary embeddings $E \in \mathbb{R}^{L_p\times d}$ instead. Since g_i is a distribution over the vocabulary, each E_i is a convex combination of the same N projected words, where the weights are given by the grammar. Therefore, once the vocabulary is learned, the grammar alone decides how a patch is represented, regardless of its domain or position. Patches with different grammars are represented differently. Thus, the grammar describes exactly what the backbone reads. This makes it an **interpretable** view of how the learned words are used.

3.2 Consult: Accessing the Vocabulary as Global Memory

Hora processes the vocabulary embeddings E with a decoder-only Transformer. In standard attention, each position can only gather information from the current sequence, while the temporal structures learned across sequences remain implicit in the model weights. However, in Hora, these structures are expressed through the shared temporal vocabulary, which makes them explicitly accessible as a **global memory**. The vocabulary embeddings E already carry this information, but

each embedding is a mixture from which individual words cannot be recovered. We therefore allow every Transformer layer to *consult* the vocabulary in addition to the current context, so that the words remain directly accessible throughout the backbone.

In detail, the Transformer backbone of Hora follows the standard decoder-only architecture (Das et al., 2024) with causal attention, RMSNorm (Zhang & Sennrich, 2019), rotary position embedding (Su et al., 2024), query-key normalization (Henry et al., 2020), and SwiGLU feed-forward network (Shazeer, 2020). We further adopt gated attention (Qiu et al., 2025). Let $Z^{(\ell)} \in \mathbb{R}^{L_p \times d}$ denote the hidden states at the ℓ-th layer, with $Z^{(0)} = E$. As illustrated in Figure 2(b), to consult the vocabulary, we prepend the words to the keys and values of the causal self-attention:

$$\mathrm{CausalAttention}\Big(Z^{(\ell)},\ \big[V; Z^{(\ell)}\big],\ \big[V; Z^{(\ell)}\big]\Big), \tag{5}$$

where $\mathrm{CausalAttention}(\cdot,\cdot,\cdot)$ denotes the causal self-attention operation over the queries, keys, and values, and $[V; Z^{(\ell)}]$ stacks the vocabulary words before the hidden states. Since the vocabulary appears before the entire sequence, every position can attend to all N words under the causal mask. Moreover, we do not apply positional embeddings to the vocabulary words since they are not part of the sequence. In the attention computation, the words are projected by the same matrices as the hidden states, so consulting the vocabulary introduces no additional parameters.

As a result, consulting provides the Transformer layers with two complementary sources of information: the hidden states $Z^{(\ell)}$, which are input-dependent and encode the context of the current sequence, and the temporal vocabulary V, which is input-invariant and encodes cross-domain temporal knowledge learned from the entire pretraining corpus. Since the words are position-free and precede all sequence positions, they are uniformly accessible at every depth and every position, including early patches whose causal context is still short. Therefore, the vocabulary not only represents the input, but also stays available to every layer as a global memory during reasoning. Unlike persistent memory (Sukhbaatar et al., 2019), which learns an independent set of key-value vectors in each layer, Hora consults a single vocabulary shared by all layers, and the same words are also used to read the input and to write the forecast.

3.3 Write: Composing Forecasts from the Vocabulary

As a decoder-only Transformer, Hora predicts the next patch at each position. Let $Z \in \mathbb{R}^{L_p \times d}$ denote the hidden states from the final Transformer layer, where Z_i is used to predict the $(i+1)$-th patch. A conventional output head projects Z_i directly into the output space (Nie et al., 2023). Such a projection is unconstrained by the vocabulary, so the words would only serve to represent the input while the forecast is generated in a separate space. Instead, Hora reuses the same vocabulary to *write* forecasts, as shown in Figure 2(c). The hidden state determines how the words should be composed, and each word is decoded into a patch-level output:

$$\alpha_i = f^{\mathrm{coef}}(Z_i), \quad O_i = \sum_{n=1}^{N} \alpha_{i,n}\, f^{\mathrm{dec}}(w_n), \tag{6}$$

where $f^{\mathrm{coef}} : \mathbb{R}^d \to \mathbb{R}^N$ is an MLP that produces the composition coefficients α_i, $f^{\mathrm{dec}} : \mathbb{R}^d \to \mathbb{R}^{P \times Q}$ is a decoding MLP shared across all words, and $O_i \in \mathbb{R}^{P \times Q}$ is the prediction of the next patch over Q quantile levels. Collecting the predictions across all positions gives $O \in \mathbb{R}^{L_p \times P \times Q}$.

Writing constrains the output in the same way that reading constrains the input, where every forecast patch is a linear combination of the N patch-level outputs decoded from the words. Regardless of the hidden state, the predicted values always lie in the subspace spanned by these outputs. Therefore, reading and writing are symmetric, as the vocabulary is used both for representing the input and generating the forecast. Unlike the temporal grammar g_i, the coefficients α_i are not normalized by softmax and can be negative. The grammar describes an observed patch, while α_i constructs a new one. This construction allows the words to be subtracted, so the forecast is not restricted to their weighted averages. Composing forecasts from a fixed set of components is also explored by Graf et al. (2026), whose decoder predicts coefficients over predefined basis functions. Our design differs in that the words are learned rather than predefined, and the same words are used throughout the forecasting process rather than introduced only at the output. In this way, reading, consulting, and writing operate on the same vocabulary V, and each accesses it through its own projection in

Equations (3), (5), and (6). Since the words are shared, all three stages propagate gradients to the same w_n, so that each word is shaped by the input representation, the backbone computation, and the forecast composition together, rather than being specialized to a single stage.

Masked decoding. A forecasting horizon L_f may contain multiple patches. Autoregressive rollout would require multiple forward passes, discard distributional information predicted at earlier steps, and accumulate errors as predictions are fed back as inputs. To generate the entire horizon in a single forward pass, we adopt the contiguous patch masking strategy (Auer et al., 2025). During training, we randomly mask contiguous segments of patches, and the model must predict the masked patches from the visible context rather than from their own values. At inference, we append $\lceil L_f/P \rceil$ masked patches after the context, and take the predictions at these positions as the forecast $\widehat{X}_{L+1:L+L_f}$.

Training objective. Let $Y \in \mathbb{R}^{L_p \times P}$ denote the labels, where Y_i is the patch following the i-th input patch. Hora predicts Q quantile levels $\tau \in \mathcal{T}$ and is trained with the pinball loss:

$$\mathcal{L}_q = \frac{1}{L_p P Q} \sum_{i=1}^{L_p} \sum_{j=1}^{P} \sum_{\tau \in \mathcal{T}} \rho_\tau\big(Y_{i,j} - O_{i,j,\tau}\big), \quad \rho_\tau(u) = \max\big(\tau u, (\tau - 1)u\big), \tag{7}$$

where $O_{i,j,\tau}$ is the predicted τ-quantile of $Y_{i,j}$ and $Q = |\mathcal{T}|$. However, $\mathcal{L}_q$ alone does not force the words to differ, which creates a risk of redundancy among them (Zbontar et al., 2021). If they converge to similar directions, different grammars would produce nearly identical vocabulary embeddings, which reduces the effective capacity of the vocabulary. We therefore regularize the vocabulary by a diversity loss $\mathcal{L}_d$:

$$\mathcal{L}_d = \frac{1}{N^2} \left\| \tilde{V}\tilde{V}^\top - I \right\|_F^2, \quad \tilde{V} = \mathrm{normalize}(VW_V), \tag{8}$$

where $\mathrm{normalize}(\cdot)$ scales each row to unit length and I is the identity matrix. We regularize the projected words VW_V rather than V itself, since they are the vectors from which the vocabulary embeddings are composed in Equation (3). Minimizing $\mathcal{L}_d$ encourages the projected words to be mutually orthogonal, reducing redundancy and preserving the capacity of the vocabulary to represent diverse temporal structures. Finally, the overall training objective combines the two losses:

$$\mathcal{L} = \mathcal{L}_q + \lambda_d \mathcal{L}_d, \tag{9}$$

where λ_d is a hyperparameter that controls the contribution of the diversity loss.

4 Experiments

4.1 Experimental Setup

Training settings. Our pretraining corpus consists of 173B time points across 7 domains, of which 77B are synthetic data generated from 11 families. Low-frequency data is supplemented by mean-pooling higher-frequency series. Details of datasets are described in Appendix B. We train two sizes of our model: the base model **Hora** with 59M parameters and the large model **Hora$_\mathbf{L}$** with 366M parameters. Both have a vocabulary of N=128 words, a patch length of P=16, and a context length of 4096. More training details are provided in Appendix C. Experiments are conducted on 8 NVIDIA RTX PRO 6000 Blackwell GPUs using BF16 precision.

Benchmarks. Hora is evaluated on GIFT-Eval (Aksu et al., 2024), which has 97 tasks across short, medium, and long forecasting horizons. We report the Continuous Ranked Probability Score (CRPS) for probabilistic forecasts and the Mean Absolute Scaled Error (MASE) for point forecasts. Both metrics are normalized by Seasonal Naive and aggregated by the geometric mean. Additionally, the average per-task rank is reported. According to the official criteria, Hora is a *zero-shot* model because our pretraining corpus contains no GIFT-Eval dataset.

Baselines. We compare our model against a broad set of baselines following the official categories of GIFT-Eval. Zero-shot TSFMs include VisionTS (Chen et al., 2025), Moirai (Woo et al., 2024), Sundial (Liu et al., 2025b), PatchFM, Kairos (Feng et al., 2025), YingLong (Wang et al., 2025a), TinyCast (Steinhauser, 2026), TabPFN-TS (Hoo et al., 2025), CleanTS, TempoPFN (Moroshan et al., 2025), Toto (Cohen et al., 2025), FlowState (Graf et al., 2026), and TimesFM-2.5. Pretrained TSFMs include TimesFM (Das et al., 2024), Chronos (Ansari et al., 2024), Chronos-Bolt,

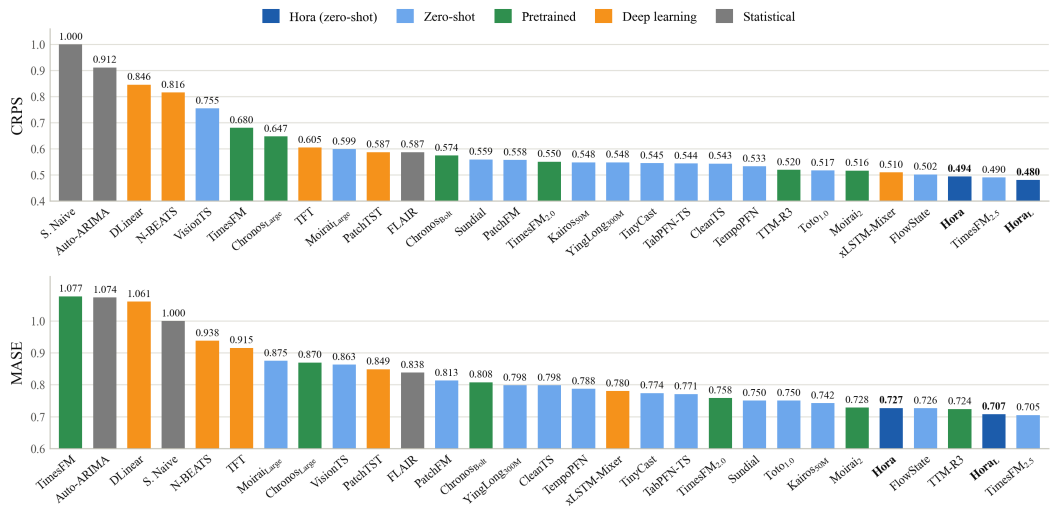


Figure 3: Performance on the GIFT-Eval benchmark. Lower CRPS/MASE is better.

TimesFM-2.0, TTM-R3 (Ekambaram et al., 2024), and Moirai2 (Liu et al., 2025a). Deep learning (full-shot) models include DLinear (Zeng et al., 2023), N-BEATS (Oreshkin et al., 2020), TFT (Lim et al., 2021), PatchTST (Nie et al., 2023), and xLSTM-Mixer (Kraus et al., 2025). Statistical models include Seasonal Naive (S. Naive), Auto-ARIMA, and FLAIR (Honda, 2026).

4.2 Zero-shot Forecasting

Figure 3 shows the performance comparison on GIFT-Eval, and Figure 7 in the appendix reports the ranks. We obtain the following observations. (1) Our model shows superior zero-shot forecasting performance. The large Hora_{L} achieves the best CRPS of 0.480 among the compared models, ahead of the strongest zero-shot baseline TimesFM-2.5 at 0.490. Its MASE of 0.707 is on par with state-of-the-art models. Moreover, Hora_{L} outperforms all baselines on both CRPS Rank and MASE Rank, indicating its robust and consistent performance across tasks. (2) The base Hora also achieves a competitive CRPS of 0.494 with only 59M parameters. It outperforms all full-shot deep learning models and even many large pretrained TSFMs that may have seen the GIFT-Eval training splits. (3) Hora_{L} improves over Hora on both metrics and their ranks, showing that the vocabulary-based design remains effective at a larger model size. Hereafter, we use the base Hora for all analyses.

4.3 Ablation Study

In this section, we conduct an ablation study with the following variants. (1) w/o vocabulary: removes the temporal vocabulary, reducing the model to a Transformer backbone. (2) w/o consult: removes vocabulary consulting from the attention layers. (3) w/o write: replaces the vocabulary-based output with a direct MLP projection. (4) w/o diversity loss: removes the diversity loss. (5) w/o consult at infer: disables consulting only at inference. (6) w/ residual read: adds a skip connection that lets the input patches bypass the vocabulary. (7) w/ hard assignment: reads each patch as its most similar word instead of a mixture. As shown in Table 1, every variant degrades the overall performance. First, disabling consulting at inference increases CRPS to 0.907 (1.84×), which is even worse than retraining the model without vocabulary. Therefore, the trained backbone actively accesses the vocabulary at inference rather than treating it as an optional addition. Second, the residual read variant breaks the strict bottleneck but brings no improvement, suggesting that the temporal grammar already carries what the backbone needs from the input. Third, the hard assignment variant increases CRPS to 0.619 (1.25×), which is the worst among all retrained variants. Patches assigned to the same word become indistinguishable, so the model loses the differences between similar inputs. This suggests that a mixture of the words is essential for keeping patches distinguished. Overall, each component contributes to the zero-shot forecasting performance of Hora.

Table 1: Ablation study on GIFT-Eval. Normalized CRPS and MASE are reported across short, medium, and long forecasting horizons, along with the overall results.

	Short		Medium		Long		Overall	
Variant	CRPS	MASE	CRPS	MASE	CRPS	MASE	CRPS	MASE
w/o vocabulary	0.517	0.702	0.470	0.754	0.473	0.779	0.497	0.729
w/o consult	0.516	0.703	0.474	0.750	0.475	0.780	0.498	0.729
w/o write	0.517	0.704	0.477	0.757	0.481	0.791	0.500	0.734
w/o diversity loss	0.520	0.704	0.479	0.764	0.478	0.789	0.502	0.734
w/o consult at infer	0.797	1.044	1.088	1.489	1.059	1.524	0.907	1.224
w/ residual read	0.521	0.705	0.478	0.756	0.475	0.783	0.501	0.732
w/ hard assignment	0.680	0.937	0.552	0.898	0.544	0.902	0.619	0.921
Hora	**0.515**	**0.702**	**0.465**	**0.748**	**0.470**	**0.773**	**0.494**	**0.727**

4.4 Hyperparameter Study

The number of words N controls the size of the temporal vocabulary. We study its effect by varying N from 32 to 512 while keeping all other settings fixed. Figure 4 reports the overall CRPS and MASE of Hora on GIFT-Eval. Both metrics reach their lowest values at N=128 and increase on both sides. A smaller vocabulary lacks the capacity to represent diverse time series patterns. A larger vocabulary also performs worse, possibly because the patches are spread over more words, so each word is shared less across domains. Moreover, it increases the inference cost, since the words are consulted in every attention layer. We therefore use N=128 for Hora, and adopt it for Hora$_{\text{L}}$ without further tuning.

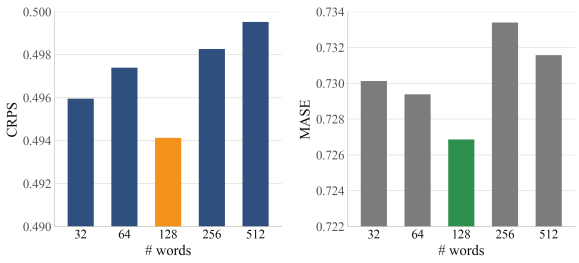


Figure 4: Overall CRPS and MASE of Hora on GIFT-Eval w.r.t. different numbers of words N.

4.5 Case Study

Since the backbone reads each patch only through its temporal grammar, visualizing the grammar shows how Hora represents a time series. For visualization clarity, we show each grammar relative to the average grammar of a pool of simple synthetic series, so that the words that distinguish a series can stand out. We first analyze the grammars of basic patterns, and then mixed patterns.

Grammar of basic patterns. We cluster a variety of public time series datasets and collect 15 basic patterns commonly seen in real-world time series, each with three example sequences from different datasets. Their temporal grammars are visualized in Figure 5. The grammar of a pattern is the average of the series grammar g in Equation (4) over its sequences, and the top 5 words consistently weighted high in its sequences are marked as its *signature*. We obtain the following observations. (1) Different patterns have different grammars, while sequences of the same pattern share similar grammars even when they come from different domains. For example, the three sequences in the *peak* panel are from nature, energy, and web. This suggests that the temporal vocabulary is **universal** across domains. (2) Patterns that look different can still share words. For example, the signatures of *trend*, *saturation*, *step*, and *random walk* share w_{76}, w_{77}, and w_{124}, and all of these sequences exhibit relatively stable dynamics without strong oscillations or bursts. Similarly, the signature of *spike* shares w_{41} and w_{127} with *peak* and w_2 and w_{88} with *intermittent*, suggesting a family of event-like patterns. These examples suggest that the grammar reflects how a series moves, such as smoothly or in bursts, rather than its shape alone. In summary, without any pattern labels, the grammars organize real-world patterns into a taxonomy.

Grammar of mixed patterns. Beyond single patterns, we further examine a mixture of patterns in Figure 6. The left sequence is a sinusoidal wave, and the right sequence adds noise to the same wave. The two sequences have similar shapes, since the noise only roughens the wave

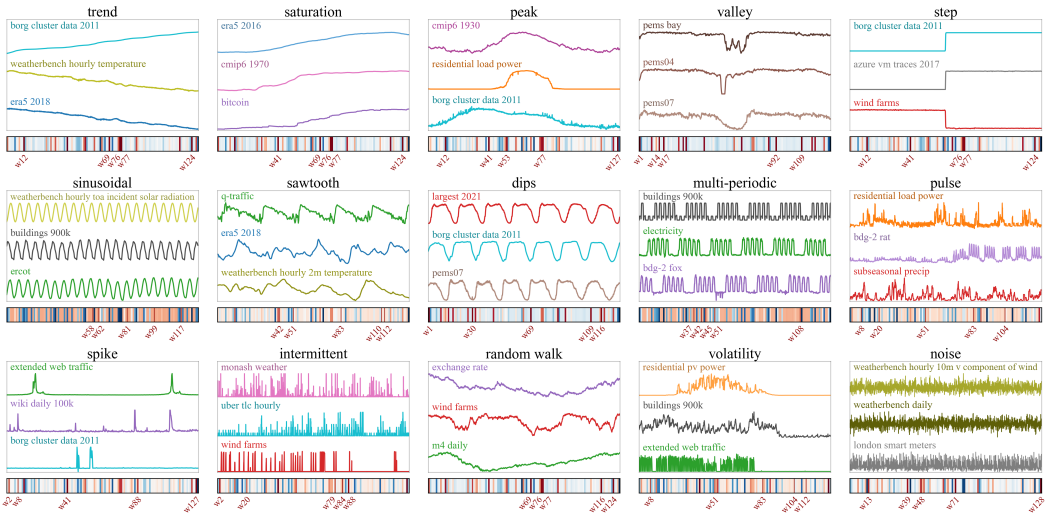


Figure 5: Temporal grammars of 15 basic time series patterns. Each panel shows three sequences of one pattern from different datasets. The strip below shows the temporal grammar of the pattern, with the top 5 words marked as its signature. Blue: relatively low weight; red: relatively high weight.

without changing its period. However, their grammars are different. The top 5 words of the clean wave are exactly the signature of the *sinusoidal* pattern in Figure 5. The noisy wave retains w_{62} and w_{117} from this signature, and it also places high weights on new words such as w_{18}, w_{49}, and w_{105}. Notably, these new words are not in the signature of the *noise* pattern. A mixture of patterns is therefore not simply a union of their signatures. Moreover, the two waves have close values but different grammars, while sequences of the same pattern share similar grammars across domains in Figure 5. Therefore, the vocabulary encodes patterns rather than values. Apart from input normalization, since the backbone reads the input only through the grammar, the difference between the two forecasts can only come from the difference between their grammars. Hora accurately continues the clean wave with a narrow prediction range, while for the noisy wave it recovers the underlying wave in the median and widens the range to cover the noise. These visualizations show that the semantics of a time series is carried by its grammar rather than by individual words, and suggest that the grammar can capture the underlying dynamics of time series. A further case study on the frequency domain is provided in Appendix D.2.

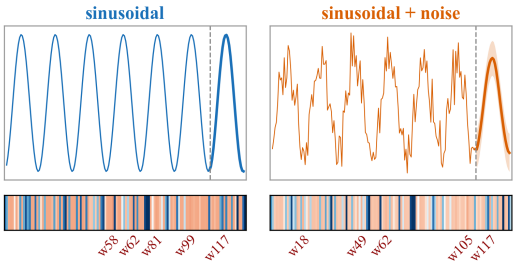


Figure 6: Temporal grammars of a sinusoidal wave and the same wave with added noise. Each panel shows the input sequence and the quantile forecast of Hora (median and range) after the dashed line. Below is the temporal grammar of each wave with its top 5 words marked. Blue: relatively low weight; red: relatively high weight.

5 Conclusion

In this paper, we study what the vocabulary of time series should be and propose Hora, a TSFM built on a temporal vocabulary of 128 words learned from billions of cross-domain time points. The same words are used to read the input, consulted by every layer, and written into the forecast. Hora achieves state-of-the-art zero-shot forecasting performance on GIFT-Eval. Ablations show that the backbone relies on the vocabulary, and that reading patches as mixtures of words is essential. Visualizations further show that the temporal grammar reflects patterns shared across domains. These results suggest that an explicit temporal vocabulary is an effective foundation for TSFMs. In future work, we will extend the temporal vocabulary to multivariate forecasting and other time series tasks.

AI USE STATEMENT

In this work, we used generative AI tools to help implement the generators of the synthetic data in our pretraining corpus. We have not used generative AI tools to develop the research idea, design the methodology, or design the experiments, and the formulation or proof of mathematical claims is not applicable to this work. Additionally, we used generative AI tools to polish the writing of the entire paper and to draft parts of the appendix, including the notation list and the summary and descriptions of the pretraining datasets together with their statistics. We have reviewed all AI-assisted work. We verified the correctness of the synthetic data generators, and checked the AI-drafted appendix content against the code and data sources. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Hora's pretraining corpus consists only of numerical time series. Our private dataset of road traffic flow contains no information about individual vehicles or persons, and it is used under an agreement with its provider. The other real-world series are drawn from public collections, which are anonymized by their providers and widely used in other works. The synthetic series are either sampled from mathematical processes or recombined from the above real-world series, so they introduce no additional data sources. Therefore, the corpus contains no personally identifiable information.

REPRODUCIBILITY STATEMENT

We have made the following efforts to ensure the reproducibility of our work. The architecture of Hora is described in Section 3, and the model and training hyperparameters of both model sizes are summarized in Appendix C. The composition of the pretraining corpus, including the real-world sources, the frequency-aggregated data, and the 11 families of synthetic data, is described in Appendix B. All real-world data are drawn from public collections except for a private dataset of road traffic flow, and every GIFT-Eval dataset is excluded. The source code of the model and the evaluation on GIFT-Eval are available at `https://anonymous.4open.science/r/Hora`.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11): 4311–4322, 2006.

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. *arXiv preprint arXiv:2410.10393*, 2024.

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. Gluonts: Probabilistic and neural time series modeling in python. *Journal of Machine Learning Research*, 21(116):1–6, 2020.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C Maddix, Hao Wang, 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.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

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. *Advances in Neural Information Processing Systems*, 38:57529–57580, 2025.

Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, François-Xavier Aubet, Laurent Callot, and Tim Januschowski. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6), 2022.

Christoph Bergmeir, Quang Bui, Frits de Nijs, and Peter Stuckey. Residential power and battery data. Zenodo, `https://doi.org/10.5281/zenodo.8219786`, 2023.

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 *International Conference on Machine Learning*, pp. 8979–9007. PMLR, 2025.

Song Chen. Beijing multi-site air quality. UCI Machine Learning Repository, `https://doi.org/10.24432/C5RK5G`, 2017.

Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean du Terrail, Anna-Monica Toon, Kan Wang, Stephan Xie, Zongzhe Xu, Viktoriya Zhukova, David Asker, Ameet Talwalkar, and Othmane Abou-Amal. This time is different: An observability perspective on time series foundation models. *Advances in Neural Information Processing Systems*, 38:50907–50951, 2025.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *International Conference on Machine Learning*, pp. 10148–10167. PMLR, 2024.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V Naidu, and Colin White. Forecastpfn: Synthetically-trained zero-shot forecasting. *Advances in Neural Information Processing Systems*, 36:2403–2426, 2023.

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. *Advances in Neural Information Processing Systems*, 37:74147–74181, 2024.

Shereen Elsayed, Daniela Thyssens, Ahmed Rashed, Hadi Samer Jomaa, and Lars Schmidt-Thieme. Do we really need deep learning models for time series forecasting? *arXiv preprint arXiv:2101.02118*, 2021.

Patrick Emami, Abhijeet Sahu, and Peter Graf. Buildingsbench: A large-scale dataset of 900k buildings and benchmark for short-term load forecasting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 19823–19857, 2023.

Kun Feng, Shaocheng Lan, Yuchen Fang, Wenchao He, Sihan Lu, Shuqi Gu, Lintao Ma, Xingyu Lu, and Kan Ren. Kairos: Toward adaptive and parameter-efficient time series foundation models. *arXiv preprint arXiv:2509.25826*, 2025.

Rakshitha W Godahewa, Christoph Bergmeir, Geoffrey Webb, Rob Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

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*, pp. 16115–16152. PMLR, 2024.

Lars Graf, Thomas Ortner, Stanisław Woźniak, and Angeliki Pantazi. Flowstate: Sampling-rate-equivariant time-series forecasting. In *International Conference on Machine Learning*, 2026.

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

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4246–4253, 2020.

Takato Honda. Don't learn the shape: Forecasting periodic time series by rank-1 decomposition. *arXiv preprint arXiv:2605.07222*, 2026.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending tabpfn-v2 to time series forecasting. *arXiv preprint arXiv:2501.02945*, 2025.

Rob J Hyndman and Yeasmin Khandakar. Automatic time series forecasting: the forecast package for r. *Journal of Statistical Software*, 27(3):1–22, 2008.

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

Ming Jin, Yaxuan Kong, Yuxuan Liang, Chaoli Zhang, Siqiao Xue, Xue Wang, James Zhang, Yi Wang, Haifeng Chen, Xiao-Li Li, et al. Large models for time series and spatio-temporal data: A survey and outlook. *ACM Computing Surveys*, 58(14):1–38, 2026.

Maurice Kraus, Felix Divo, Devendra Singh Dhami, and Kristian Kersting. xlstm-mixer: Multivariate time series forecasting by mixing via scalar memories. *Advances in Neural Information Processing Systems*, 38:6547–6581, 2025.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 95–104, 2018.

Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 6555–6565, 2024.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

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.

Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144, 2007.

Chenghao Liu, Taha Aksu, Juncheng Liu, Xu Liu, Hanshu Yan, Quang Pham, Silvio Savarese, Doyen Sahoo, Caiming Xiong, and Junnan Li. Moirai 2.0: When less is more for time series forecasting. *arXiv preprint arXiv:2511.11698*, 2025a.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022.

Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. Largest: A benchmark dataset for large-scale traffic forecasting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 75354–75371, 2023.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024a.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *International Conference on Machine Learning*, pp. 32369–32399. PMLR, 2024b.

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. In *International Conference on Machine Learning*, pp. 39295–39317. PMLR, 2025b.

Vladyslav Moroshan, Julien Siems, Arber Zela, Timur Carstensen, and Frank Hutter. Tempopfn: Synthetic pre-training of linear rnns for zero-shot time series forecasting. *arXiv preprint arXiv:2510.25502*, 2025.

Soukayna Mouatadid, Paulo Orenstein, Genevieve Flaspohler, Miruna Oprescu, Judah Cohen, Franklyn Wang, Sean Knight, Maria Geogdzhayeva, Sam Levang, Ernest Fraenkel, and Lester Mackey. Subseasonalclimateusa: A dataset for subseasonal forecasting and benchmarking. In *Advances in Neural Information Processing Systems*, volume 36, pp. 7960–7992, 2023.

K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *International Conference on Artificial Neural Networks*, pp. 999–1004, 1997.

Tung Nguyen, Jason Jewik, Hritik Bansal, Prakhar Sharma, and Aditya Grover. Climatelearn: Benchmarking machine learning for weather and climate modeling. In *Advances in Neural Information Processing Systems*, volume 36, pp. 75009–75025, 2023.

Zelin Ni, Hang Yu, Shizhan Liu, Jianguo Li, and Weiyao Lin. Basisformer: Attention-based time series forecasting with learnable and interpretable basis. *Advances in Neural Information Processing Systems*, 36:71222–71241, 2023.

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

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.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *Advances in Neural Information Processing Systems*, 38:100092–100118, 2025.

Stephan Rasp, Peter D Dueben, Sebastian Scher, Jonathan A Weyn, Soukayna Mouatadid, and Nils Thuerey. Weatherbench: A benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, 2020.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

Patrick Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

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. In *International Conference on Learning Representations*, pp. 34635–34667, 2025.

Armin Steinhauser. Tinycast: Probabilistic zero-shot forecasting with computed periodicity. *arXiv preprint arXiv:2608.15767*, 2026.

James H Stock and Mark W Watson. Vector autoregressions. *Journal of Economic Perspectives*, 15 (4):101–115, 2001.

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

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Hervé Jégou, and Armand Joulin. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*, 2019.

Sabera J Talukder, Yisong Yue, and Georgia Gkioxari. Totem: Tokenized time series embeddings for general time series analysis. *Transactions on Machine Learning Research*, 2024.

Xiaoyu Tao, Shilong Zhang, Mingyue Cheng, Daoyu Wang, Tingyue Pan, Bokai Pan, Changqing Zhang, and Shijin Wang. From values to tokens: An llm-driven framework for context-aware time series forecasting via symbolic discretization. *arXiv preprint arXiv:2508.09191*, 2025.

Willem G van Panhuis, Anne Cross, and Donald S Burke. Project tycho 2.0: A repository to improve the integration and reuse of data for global population health. *Journal of the American Medical Informatics Association*, 25(12):1608–1617, 2018.

Jingyuan Wang, Jiawei Jiang, Wenjun Jiang, Chao Li, and Wayne Xin Zhao. Libcity: An open library for traffic prediction. In *Proceedings of the 29th International Conference on Advances in Geographic Information Systems*, pp. 145–148, 2021.

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*, 2025a.

Zhixian Wang, Qingsong Wen, Chaoli Zhang, Liang Sun, Leandro Von Krannichfeldt, and Yi Wang. Benchmarks and custom package for electrical load forecasting. *arXiv preprint arXiv:2307.07191v1*, 2023.

Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025b.

Yunshi Wen, Tengfei Ma, Tsui-Wei Weng, Lam M Nguyen, and Anak Agung Julius. Abstracted shapes as tokens – a generalizable and interpretable model for time-series classification. *Advances in Neural Information Processing Systems*, 37, 2024.

Gerald Woo, Chenghao Liu, Akshat Kumar, and Doyen Sahoo. Pushing the limits of pre-training for time series forecasting in the cloudops domain. *arXiv preprint arXiv:2310.05063*, 2023.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *International Conference on Machine Learning*, pp. 53140–53164. PMLR, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023.

Shifeng Xie, Vasilii Feofanov, Jianfeng Zhang, Themis Palpanas, and Ievgen Redko. Cauker: Classification time series foundation models can be pretrained on synthetic data. In *International Conference on Learning Representations*, pp. 68606–68634, 2026.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pp. 12310–12320. PMLR, 2021.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11121–11128, 2023.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2267–2276, 2015.

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, pp. 11106–11115, 2021.

Jingbo Zhou, Xinjiang Lu, Yixiong Xiao, Jian Tang, Jiantao Su, Yu Li, Ji Liu, Junfu Lyu, Yanjun Ma, and Dejing Dou. Sdwpf: A dataset for spatial dynamic wind power forecasting over a large turbine array. *Scientific Data*, 11(1):649, 2024.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. *Advances in Neural Information Processing Systems*, 36:43322–43355, 2023.

A NOTATION LIST

For reference, Table 2 summarizes the key notations and their descriptions in this paper.

Table 2: Notations in the paper.

Symbol	Description
L	Length of the input time series
L_f	Forecasting horizon
P	Patch length
L_p	Number of patches
N	Number of words
d	Hidden dimension
d_{ff}	FFN hidden dimension
Q	Number of quantile levels
$X \in \mathbb{R}^L$	Input time series
$X^p \in \mathbb{R}^{L_p \times P}$	Patched input time series
$M^p \in \{0,1\}^{L_p \times P}$	Binary mask of the patched input
$H \in \mathbb{R}^{L_p \times d}$	Encoded patches
$V \in \mathbb{R}^{N \times d}$	Temporal vocabulary
$w_n \in \mathbb{R}^d$	The n-th word, i.e., the n-th row of V
$g_i \in \mathbb{R}^N$	Temporal grammar of the i-th patch
$g \in \mathbb{R}^N$	Temporal grammar of the input time series
$E \in \mathbb{R}^{L_p \times d}$	Vocabulary embeddings of the input
$Z^{(\ell)} \in \mathbb{R}^{L_p \times d}$	Hidden states of the ℓ-th Transformer layer
$Z \in \mathbb{R}^{L_p \times d}$	Output of the last Transformer layer
W_Q, W_K, W_V	Projection matrices in reading
$f^{\text{enc}}, f^{\text{dec}}$	Encoding and decoding functions
f^{coef}	Coefficient projection function
$\alpha_i \in \mathbb{R}^N$	Composition coefficients of the i-th predicted patch
$O \in \mathbb{R}^{L_p \times P \times Q}$	Next-patch prediction of the model
$Y \in \mathbb{R}^{L_p \times P}$	Next-patch labels for training
$\tau \in \mathcal{T}$	Quantile level and the set of quantile levels
$\mathcal{L}_q$	Quantile loss
$\mathcal{L}_d$	Diversity loss
λ_d	Weight of the diversity loss
$\mathcal{L}$	Total loss
r	Pearson correlation

B PRETRAINING DATASETS

We pretrain on a corpus of 62M time series and 173B time points assembled for this work (Table 3). It spans 7 domains (energy, finance, healthcare, nature, sales, transport, web) and sampling intervals from 4 seconds to 1 year, and consists of three parts: real-world series at their native frequency, frequency-aggregated views of those series, and synthetic series.

Real-world data. The real-world part (92B points) is curated from two large open collections of time series data, Time-300B (Shi et al., 2025) and LOTSA (Woo et al., 2024), as well as a private dataset of road traffic flow. Every GIFT-Eval dataset is excluded, so our evaluation is zero-shot.

Frequency-aggregated data. Real-world series are abundant at hourly and finer rates but scarce at daily, weekly, and monthly rates, whereas many forecasting tasks are sampled at these coarser rates. We therefore complement the native-frequency data with coarser views of the real datasets: series from selected datasets are mean-pooled onto minutely, hourly, daily, weekly, and monthly grids. Unlike synthetic data, these views are genuine observations of real processes and inherit the

calendar structure of their source. They contribute only 3.8B points but supply most of the weekly and monthly series in the corpus. Table 3 reports them as one row per target frequency.

Synthetic data. Synthetic series make up 77B points in 11 families. Three follow published generators, and the remaining eight are our own, each built around a behavior that real-world data under-represent.

- **KernelSynth** (Ansari et al., 2024) draws from Gaussian-process priors with randomly composed kernels, which we augment with explicit harmonic components and polynomial trends.
- **TSMixup** (Ansari et al., 2024) forms convex combinations of a few normalized crops of real series from the corpus, over-sampling the scarce domains when choosing the constituents; it recombines real data rather than sampling a prior.
- **CauKer** (Xie et al., 2026) samples structural causal models whose root nodes are Gaussian-process draws and whose descendants are random nonlinear transforms of their parents, producing saturated, folded, and gated shapes that smooth priors cannot.
- **Trend–seasonal–irregular** series follow the classical additive decomposition: a trend of random type (linear, piecewise-linear, logistic, or exponential), a few seasonal components at calendar periods, and irregular noise with sparse spikes and level shifts.
- **Regime-switching OU** series follow an Ornstein–Uhlenbeck process whose mean level, mean-reversion speed, and volatility change from regime to regime, either abruptly or by ramping; inspired by Moroshan et al. (2025), this family mimics non-stationary operational and financial signals.
- **Level shifts** are piecewise-constant series with hard or gradual transitions between levels and segment-wise slopes and noise regimes, targeting change-point behavior.
- **Calendar seasonality** combines nested calendar periods (e.g., daily and weekly cycles at hourly rate, weekly and yearly cycles at daily rate), rendered as repeating per-slot templates rather than sinusoids so that weekday–weekend contrasts and other non-sinusoidal profiles appear, together with slow amplitude modulation and trend.
- **Trend and growth curves** are trend-dominated series of exponential, logistic, Gompertz, epidemic-wave, power-law, or piecewise-linear shape, with weak seasonality and level-proportional noise.
- **Cumulative growth** stacks saturating growth waves and observes them through a random window, so that onset or saturation may lie outside the series; a fraction of the series are integer counters with flat plateaus and leading zeros, mimicking cumulative case counts whose context shows only the rising segment.
- **Intermittent counts** are non-negative integer series, either intermittent demand with sporadic arrivals or zero-inflated Poisson and negative-binomial counts driven by trend and calendar periods, mimicking sparse retail and hospital data.
- **Bursts** place sporadic bursts, shaped as plateaus or exponential decays with aftershocks, on a quiet autocorrelated background, mimicking operations telemetry at second-level resolution.

Table 3: Summary of the pretraining corpus. *Aggregated* rows are mean-pooled copies of the real datasets above on coarser grids, one row per target frequency. "Ours" marks data generated by us (KernelSynth and TSMixup follow Ansari et al. (2024), CauKer follows Xie et al. (2026)).

Dataset	Domain	Frequency	# Time Series	# Time Points	Source
Australian Electricity	Energy	30T	5	1,153,584	Godahewa et al. (2021)
BDG-2 Bear	Energy	H	215	1,422,320	Emami et al. (2023)
BDG-2 Bull	Energy	H	464	501,832	Wang et al. (2023)
BDG-2 Cockatoo	Energy	H	4	17,032	Wang et al. (2023)
BDG-2 Fox	Energy	H	179	2,285,288	Emami et al. (2023)
BDG-2 Hog	Energy	H	152	365,304	Wang et al. (2023)
BDG-2 Panther	Energy	H	136	893,840	Emami et al. (2023)
BDG-2 Rat	Energy	H	455	4,596,080	Emami et al. (2023)
Borealis	Energy	H	17	82,757	Emami et al. (2023)
Buildings900K	Energy	H	2,466,924	15,199,263,843	Emami et al. (2023)
Covid19 Energy	Energy	H	1	31,912	Wang et al. (2023)

Continued on next page

Table 3 continued from previous page

Dataset	Domain	Frequency	# Time Series	# Time Points	Source
ELF	Energy	H	1	21,792	Wang et al. (2023)
ERCOT Load	Energy	H	152	1,238,832	ERCOT
Elecdemand	Energy	30T	1	17,520	Godahewa et al. (2021)
GEF12	Energy	H	20	788,280	Wang et al. (2023)
GEF14	Energy	H	1	17,520	Wang et al. (2023)
GEF17	Energy	H	8	140,352	Wang et al. (2023)
IDEAL	Energy	H	225	1,253,088	Emami et al. (2023)
KDD Cup 2022	Energy	10T	405	4,687,594	Zhou et al. (2024)
London Smart Meters	Energy	30T	24,132	160,041,727	Godahewa et al. (2021)
Low Carbon London	Energy	H	1,032	9,354,695	Emami et al. (2023)
PDB	Energy	H	1	17,520	Wang et al. (2023)
Residential Load Power	Energy	T	79,508	404,832,695	Bergmeir et al. (2023)
Residential PV Power	Energy	T	9,999	364,764,078	Bergmeir et al. (2023)
SMART	Energy	H	5	95,709	Emami et al. (2023)
Sceaux	Energy	H	1	34,223	Emami et al. (2023)
Solar Power	Energy	4S	1	7,397,222	Godahewa et al. (2021)
Spanish	Energy	H	1	35,064	Wang et al. (2023)
Wind Farms	Energy	T	1,298	142,825,880	Godahewa et al. (2021)
Wind Power	Energy	4S	1	7,397,147	Godahewa et al. (2021)
Bitcoin	Finance	D	29	68,927	Godahewa et al. (2021)
CIF 2016	Finance	M	144	13,442	Godahewa et al. (2021)
Exchange Rate	Finance	B	13	56,096	Ansari et al. (2024)
FRED MD	Finance	M	104	71,624	Godahewa et al. (2021)
GoDaddy	Finance	M	6,270	257,070	Kaggle
M1 Monthly	Finance	M	590	44,080	Godahewa et al. (2021)
M1 Quarterly	Finance	Q	195	9,628	Godahewa et al. (2021)
M1 Yearly	Finance	Y	106	3,136	Godahewa et al. (2021)
M3 Monthly	Finance	M	1,428	141,858	Godahewa et al. (2021)
M3 Other	Finance	Q	174	11,933	Godahewa et al. (2021)
M3 Quarterly	Finance	Q	755	36,960	Godahewa et al. (2021)
M3 Yearly	Finance	Y	645	18,319	Godahewa et al. (2021)
NN5 Daily	Finance	D	111	81,585	Godahewa et al. (2021)
NN5 Weekly	Finance	W	111	11,655	Godahewa et al. (2021)
Tourism Monthly	Finance	M	359	98,867	Godahewa et al. (2021)
Tourism Quarterly	Finance	Q	427	39,128	Godahewa et al. (2021)
Tourism Yearly	Finance	Y	419	11,198	Godahewa et al. (2021)
CDC Fluview ILINet	Healthcare	W	316	279,652	CDC
CDC Fluview WHO NREVSS	Healthcare	W	296	167,040	CDC
Project Tycho	Healthcare	W	2,735	1,248,145	van Panhuis et al. (2018)
Beijing Air Quality	Nature	H	4,262	2,932,657	Chen (2017)
China Air Quality	Nature	H	17,686	4,217,605	Zheng et al. (2015)
ERA5 (2015–2018)	Nature	H	1,587,763	12,505,756,032	Nguyen et al. (2023)
Oikolab Weather	Nature	H	309	615,574	Godahewa et al. (2021)
Subseasonal	Nature	D	17,604	51,968,498	Mouatadid et al. (2023)
Subseasonal Precipitation	Nature	D	862	9,760,426	Mouatadid et al. (2023)
Sunspot	Nature	D	20	73,545	Godahewa et al. (2021)
Weather	Nature	D	3,010	42,941,700	Godahewa et al. (2021)
Weatherbench (Daily)	Nature	D	301,229	3,223,513,345	Rasp et al. (2020)
Weatherbench (Hourly)	Nature	H	1,357,738	21,773,804,182	Rasp et al. (2020)
Weatherbench (Weekly)	Nature	W	226,533	462,956,049	Rasp et al. (2020)
Dominick	Sales	D	3,712	759,817	Godahewa et al. (2021)
Favorita Sales	Sales	D	279,153	127,281,306	Kaggle
Favorita Transactions	Sales	D	258	81,196	Kaggle
M5	Sales	D	30,490	58,327,370	Alexandrov et al. (2020)
Beijing Subway	Transport	30T	2,760	629,280	Wang et al. (2021)
Covid Mobility	Transport	D	426	120,950	Godahewa et al. (2021)
HZMetro	Transport	15T	4,000	292,000	Wang et al. (2021)
LargeST (2017–2021)	Transport	5T	1,208,997	4,175,062,621	Liu et al. (2023)
Los-Loop	Transport	5T	3,381	6,231,168	Wang et al. (2021)
Mexico City Bikes	Transport	H	556	78,848	Ansari et al. (2024)
PEMS Bay	Transport	5T	3,980	15,975,920	Wang et al. (2021)
PEMS03	Transport	5T	1,651	9,210,432	Wang et al. (2021)
PEMS04	Transport	5T	6,634	14,638,784	Wang et al. (2021)
PEMS07	Transport	5T	3,828	23,789,760	Wang et al. (2021)
PEMS08	Transport	5T	2,612	8,684,480	Wang et al. (2021)
Pedestrian Counts	Transport	H	80	3,125,914	Godahewa et al. (2021)
Q-Traffic	Transport	15T	46,990	257,200,384	Wang et al. (2021)
Rideshare	Transport	H	2,304	443,892	Godahewa et al. (2021)
Road Traffic Flow	Transport	5T	1,427,921	30,045,950,052	Private
SHMetro	Transport	15T	52,992	3,868,416	Wang et al. (2021)

Continued on next page

Table 3 continued from previous page

Dataset	Domain	Frequency	# Time Series	# Time Points	Source
Taxi	Transport	30T	96,758	40,584,636	Alexandrov et al. (2020)
Taxi (Hourly)	Transport	H	2,433	1,762,024	Ansari et al. (2024)
Traffic	Transport	H	1,371	14,993,544	Godahewa et al. (2021)
Traffic Weekly	Transport	W	862	82,752	Godahewa et al. (2021)
Uber TLC Daily	Transport	D	235	42,533	Alexandrov et al. (2020)
Uber TLC Hourly	Transport	H	262	1,129,444	Alexandrov et al. (2020)
Vehicle Trips	Transport	D	309	25,497	Godahewa et al. (2021)
Alibaba Cluster Trace 2018	Web	5T	111,047	180,526,003	Woo et al. (2023)
Azure VM Traces 2017	Web	5T	263,928	880,648,165	Woo et al. (2023)
Borg Cluster Data 2011	Web	5T	275,528	1,029,374,675	Woo et al. (2023)
Extended Web Traffic	Web	D	161,890	332,586,145	Godahewa et al. (2021)
Kaggle Web Traffic Weekly	Web	W	133,388	15,206,232	Godahewa et al. (2021)
Wiki Daily (100k)	Web	D	100,001	274,099,872	Ansari et al. (2024)
Wiki-Rolling	Web	D	47,675	40,619,100	Alexandrov et al. (2020)
Aggregated real data	Mixed	T	2	986,291	Ours
Aggregated real data	Mixed	H	1,337,139	697,510,495	Ours
Aggregated real data	Mixed	D	5,132,434	2,048,124,676	Ours
Aggregated real data	Mixed	W	4,882,377	871,191,162	Ours
Aggregated real data	Mixed	M	566,529	149,896,356	Ours
KernelSynth	Synthetic	–	2,700,000	8,058,931,466	Ours
Trend–Seasonal–Irregular	Synthetic	–	2,700,000	8,065,683,833	Ours
TSMixup	Synthetic	–	13,500,000	15,944,714,625	Ours
CauKer	Synthetic	–	3,460,000	6,712,518,129	Ours
Regime-Switching OU	Synthetic	–	3,965,000	10,033,200,919	Ours
Calendar Seasonality	Synthetic	–	4,565,000	10,739,498,549	Ours
Trend & Growth Curves	Synthetic	–	2,420,000	4,692,539,055	Ours
Cumulative Growth	Synthetic	–	2,380,000	4,619,121,393	Ours
Level Shifts	Synthetic	–	1,380,000	2,672,579,550	Ours
Intermittent Counts	Synthetic	–	1,730,000	3,355,027,440	Ours
Bursts	Synthetic	–	1,040,000	2,013,833,474	Ours
Real data (native frequency)			10,395,999	91,960,217,898	
Aggregated real data			11,918,481	3,767,708,980	
Synthetic data			39,840,000	76,907,648,433	
Total			**62,154,480**	**172,635,575,311**	

C Training Details

Table 4 summarizes the architecture and training hyperparameters of Hora and Hora$_{\text{L}}$. The two models share the same architecture except for the hidden dimension, and are trained with the same schedule on the same corpus. Both predict 21 quantile levels $\{0.01, 0.05, 0.1, \ldots, 0.9, 0.95, 0.99\}$ with the pinball loss. During training, we apply contiguous patch masking (CPM), where contiguous segments of context patches are masked at a ratio up to 0.25. Hora$_{\text{L}}$ additionally masks the tail of the context with probability 0.5. Following Auer et al. (2025), we augment our data during training to further increase its diversity. Training takes 7 hours for Hora and 23 hours for Hora$_{\text{L}}$ on 8 GPUs.

D Additional Experiments

D.1 GIFT-Eval Ranks

Figure 7 reports the average per-task ranks that complement the aggregated metrics in Figure 3. For each of the 97 tasks, all models are ranked by CRPS or MASE, and the rank of a model is averaged over the tasks, so a lower value is better. The ranking pool consists of all 130 models on the public GIFT-Eval leaderboard as of September 2026 together with Hora and Hora$_{\text{L}}$, and the ranks of the baselines are recomputed within this pool. Since the leaderboard keeps growing, the absolute values depend on the pool and should be read relative to the baselines shown. Hora$_{\text{L}}$ obtains the best rank on both metrics (45.5 for CRPS and 51.5 for MASE). Hora ranks right after TimesFM-2.5 among the compared baselines. These ranks further confirm that the advantage of our model is consistent across tasks rather than driven by a few of them.

Table 4: Architecture and training hyperparameters of Hora and Hora_L.

	Hora	**Hora_L**
Layers	16	16
Attention heads	16	16
Hidden dimension d	512	1280
FFN dimension d_{ff}	2048	5120
# words N	128	128
Word dimension (=d)	512	1280
Patch length P	16	16
Context length	4096	4096
Quantile levels	21	21
# Parameters	59.3M	365.6M
Iterations	100k	100k
Batch size	2048	2048
Optimizer	AdamW	AdamW
Learning rate	10^{-3}	10^{-3}
LR schedule	cosine, 1k warm-up	cosine, 1k warm-up
Weight decay	0.1	0.1
(β_1, β_2)	(0.9, 0.95)	(0.9, 0.95)
Diversity loss weight λ_d	10^{-4}	10^{-4}
CPM ratio	[0, 0.25]	[0, 0.25]
CPM on tail	0	0.5
Precision	BF16	BF16

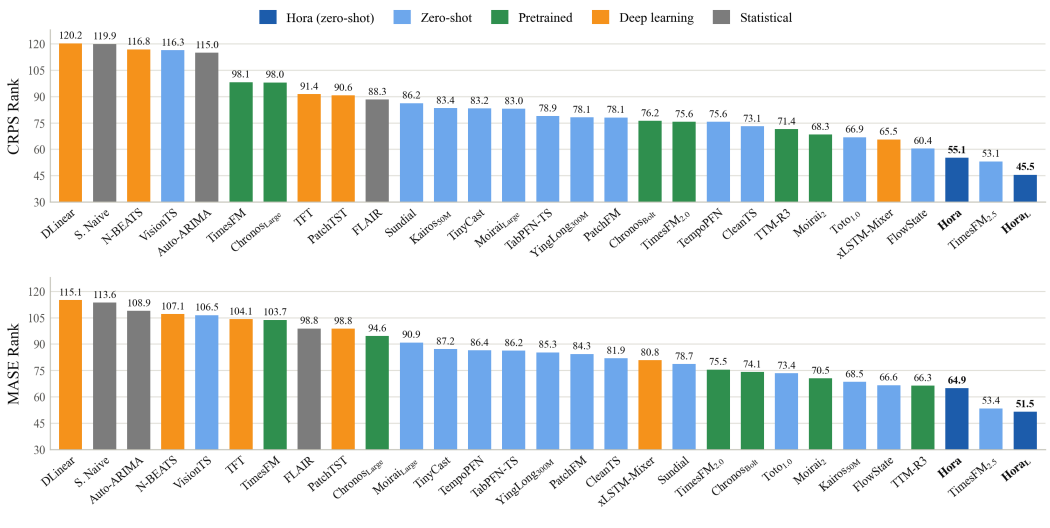


Figure 7: CRPS and MASE ranks on GIFT-Eval benchmark. Baselines are selected from zero-shot TSFMs, pretrained TSFMs, deep-learning (full-shot) models, and statistical models according to the official categories of GIFT-Eval.

D.2 Additional Case Study

We provide one more case study to show how the temporal grammar of a time series relates to its frequency-domain characteristics.

Grammar and frequency. Figure 8 compares three synthetic sequences in the time domain, temporal grammar, and frequency domain. A is a random walk, B is a slow triangular wave, and C is B with strong added noise, much stronger than the noise in Figure 6. In the time domain, A and B look very different, with a correlation of $r(A, B) = -0.05$, whereas B and C share the same underlying wave and remain similar with $r(B, C) = 0.78$. However, their temporal grammars tell a different story. A and B have similar grammars with $r(g_A, g_B) = 0.88$, while the grammars of B and C are

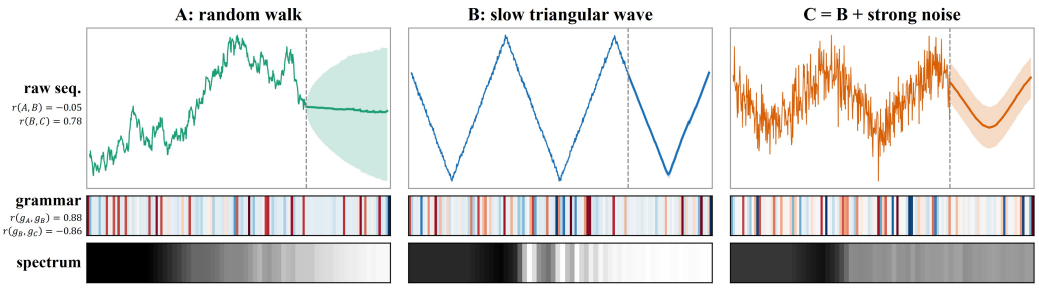


Figure 8: Temporal grammars and power spectra of three synthetic sequences. A is a random walk, B is a slow triangular wave, and C is B with added noise. The first row shows each sequence and Hora's quantile forecast after the dashed line, including the median and prediction range. The second row shows the temporal grammar of each sequence. Blue: relatively low weight; red: relatively high weight. The third row shows the power spectrum over 64 frequency bands ordered from slow (left) to fast (right), with darker bands indicating higher energy. The values on the left are the Pearson correlations between the raw sequences and between their temporal grammars.

strongly negatively correlated with $r(g_B, g_C) = -0.86$. The power spectra are consistent with this correlation reversal. A and B both concentrate their energy in the slow-frequency bands, and the vocabulary represents them with similar temporal grammars despite their different waveforms. Although A and B are different patterns, B is slow enough that each patch sees only a linear segment, so at the patch scale it behaves like a local trend. This is consistent with Figure 5, where random walk shares its signature words with trend. In contrast, the strong noise in C introduces substantial energy into the fast-frequency bands, which completely changes the grammar even though the underlying wave remains visible in the time domain. Since the grammar correlations do not follow the value correlations, the grammar reflects the patterns of a series rather than its values. It can also capture frequency-domain characteristics, not only waveform shape.