DO NORMALIZATION CHOICE MATTER FOR CAUSAL LARGE TIME-SERIES MODELS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Large models for time-series forecasting have been emerged as a promising paradigm for training models on heterogeneous collections of signals. These models typically rely on causal autoregressive architectures, where each observation is sequentially predicted from past. In practice, real-world time-series exhibit non-stationarities, which significantly influence predictive performance. To mitigate this, normalization is commonly employed. However, in efficient causal settings it might induce information leakage from future observations during training. Recent alternatives, including causal normalization and statistics computed from initial observations, have been proposed to address this issue, but their practical implications remain insufficiently understood. In this work, we evaluate normalization strategies for transformer-based large time-series models trained with patching and efficient causal strategy. We showcase that normalization choice significantly influences both training convergence and forecasting performance.

Track: Research

1 INTRODUCTION

Motivated by the success of large language models (LLMs) (OpenAI et al., 2025; Grattafiori et al., 2024), large time-series models aim to learn transferable representations from large-scale and heterogeneous corpora of signals. Towards this end, modern architectures for forecasting (Cohen et al., 2025; Auer et al., 2025; Liu et al., 2025; Shi et al., 2025; Das et al., 2024; Graf et al., 2025; Ansari et al., 2025) typically adopt patch-based representations (Nie et al., 2023). *Efficient causal training* in this context refers to schemes where each patch in a sequence is predicted based on past ones, either sequentially or in parallel, while computational efficiency is maintained through reuse of intermediate representations.

However, the statistical heterogeneity of real-world time-series, manifested in varying means, variances, and distributional shifts, poses substantial challenges for performance. While normalization techniques such as RevIN (Kim et al., 2022), designed to remove and restore time-series statistics, are widely used in supervised forecasting, their integration into large models with efficient causal training is non-trivial. In particular, computing global statistics introduces look-ahead leakage that violates causal constraints during efficient training. Although recent approaches propose causal-based (Cohen et al., 2025; Graf et al., 2025) or prefix-based (Das et al., 2024; Liu et al., 2025) normalization schemes, a principled understanding of how normalization choices influence training dynamics and downstream performance remains limited.

In this work, we present a unified evaluation of normalization strategies for causal large time-series models. We isolate normalization as the varying factor within a fixed large-scale efficient causal training framework While we primarily consider univariate settings, our analysis can be extended to multivariate cases, and we focus on Transformer architecture (Vaswani et al., 2017).

The most closely related prior study, Ahmed et al. (2025), compares normalization strategies in early large time-series models (Ansari et al., 2024; Woo et al., 2024; Feng et al., 2024), concluding that mean–variance scaling leads to better performance. However, their analysis does not address the constraints imposed by efficient causal training. Consequently, strategies such as causal-based and prefix-based are not examined.

Our contributions are twofold: (i) we formalize a categorization of normalization strategies for large time-series models, dividing them into *vanilla, prefix*, and *causal variants*; (ii) we empirically demonstrate that normalization choice play a critical role in shaping both training convergence and forecasting accuracy.

2 A Categorization of Normalizations in Large Time-Series Models

Patching for time-series was introduced in Nie et al. (2023), where a patch-based Transformer architecture was proposed for forecasting. This paradigm has since become a standard component of large time-series models. Patches correspond to contiguous segments of a time-series and play a role analogous to tokens in language models. Formally, an input signal $\mathbf{x} \in \mathbb{R}^{\mathrm{w}}$ of length w is divided into consecutive patches of length L. Let $\mathrm{N} = \lfloor \frac{\mathrm{w}}{\mathrm{L}} \rfloor$ be the number of patches and $\mathbf{x}_{\mathrm{P}} = (\mathbf{x}_{\mathrm{P}_1}, \cdots, \mathbf{x}_{\mathrm{P}_\mathrm{N}})^\top \in \mathbb{R}^{\mathrm{N}\times\mathrm{L}}$ where the i-th patch is denoted by $\mathbf{x}_{\mathrm{P}_i}$. Models are trained to predict the next patch $\mathbf{x}_{\mathrm{P}_{i+1}}$ given the previous patches $\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_i}$ for $i = 1, \ldots, \mathrm{N}-1$. Thus, a single forecast estimates the L future observations.

However, unlike discrete tokens, time-series patches are continuous-valued, which necessitates normalization to ensure robustness across signals with varying scales. Given a patch $\mathbf{x}_{\mathrm{P}_i}$, normalization is defined as:

$$\tilde{\mathbf{x}}_{\mathrm{P}_i} = \frac{\mathbf{x}_{\mathrm{P}_i} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}, \tag{1}$$

where μ_i and σ_i^2 are the mean and variance used to normalize $\mathbf{x}_{\mathrm{P}_i}$ and computed by a given *normalization strategy*. Here, the parameter $\epsilon > 0$ ensures numerical stability.

It is standard to compute normalization statistics over the entire available window, yielding the most accurate estimate of the signal distribution. However, under an efficient causal setting, training a model using this strategy without relaxation is impossible. Specifically, the normalization statistics differ when forecasting $\mathbf{x}_{\mathrm{P}_i}$ and $\mathbf{x}_{\mathrm{P}_{i+1}}$, that prevent the reuse of computations between consecutive patches (See Appendix A.1). This motivates to modify normalization strategies.

2.1 Vanilla Reversible Instance Normalization

RevIN (Kim et al., 2022) is a normalization–denormalization framework that normalizes input sequences, feeds them into the model, and subsequently denormalizes the outputs to address distribution shifts between different signals. RevIN has been adopted in recent time-series models (Auer et al., 2025; Ansari et al., 2025). In a vanilla RevIN strategy, normalization statistics are computed as

$$\mu_i = \mathtt{Mean}\big(\text{concat}(\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_\mathrm{N}})\big), \quad \sigma_i^2 = \mathtt{Var}\big(\text{concat}(\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_\mathrm{N}})\big), \tag{2}$$

where each patch is normalized using the same global statistics. This allows for efficient training, but it violates causal constraints. However, during inference, this strategy provides the most accurate statistics for normalization.

2.2 Prefix@k Reversible Instance Normalization

To avoid statistics leakage during training, a prefix-based strategy computes normalization statistics using only the first k patches ($k = 8$ in our experiments), as proposed in Das et al. (2024); Liu et al. (2025). While Das et al. (2024) use only the first patch, Liu et al. (2025) generalize this approach to the first k patches, with $1 \leq k < \mathrm{N}$. To maintain training consistency, we do not forecast the first k patches; instead, we predict $\mathbf{x}_{\mathrm{P}_{k+1}}, \ldots, \mathbf{x}_{\mathrm{P}_\mathrm{N}}$, and the normalization statistics are computed as

$$\mu_i = \mathtt{Mean}\big(\text{concat}(\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_k})\big), \quad \sigma_i^2 = \mathtt{Var}\big(\text{concat}(\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_k})\big). \tag{3}$$

A principal limitation of this strategy is its sensitivity to non-stationarity that may lead to distribution shifts occurring after the prefix window, resulting in inaccurate normalization of subsequent patches. During inference, statistics can be computed either over the first k patches for consistency with training or over the full context as in vanilla RevIN. In our main experiments, we use the k-patch statistics, additional ablation results (see Appendix A.2) indicate that full-context statistics do not improve performance.

2.3 Causal Reversible Instance Normalization

Causal normalization computes statistics for patch $\mathbf{x}_{\mathrm{P}_i}$, using only preceding patches (Graf et al., 2025; Cohen et al., 2025), in the following way:

$$\mu_i = \texttt{Mean}\big(\text{concat}(\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_i})\big), \quad \sigma_i^2 = \texttt{Var}\big(\text{concat}(\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_i})\big). \tag{4}$$

This strategy preserves causal integrity and eliminates look-ahead bias. Moreover, it enables key-value caching (Gao et al., 2025) during inference, which avoids recomputation of key and value projections for previously processed patches. This mechanism significantly reduces inference cost with transformers (with an analogous reduction achieved through the reuse of hidden states in unidirectional models). Unlike other normalization approaches, this strategy uses patch-specific means and variances, adding only negligible computations (see Appendix A.3). However, the normalized representation may exhibit non-smooth transitions across patches.

2.4 Sinh and denormalization

The $\sinh^{-1}$ transformation is a continuous and monotonic function that behaves similarly to the logarithm for large values. Recent approaches (Ansari et al., 2025), building on earlier work (Uniejewski & Weron, 2018), apply this transformation after normalization to mitigate the influence of outliers in time-series. We evaluate this transformation under the three normalization strategies presented above. Thus, when incorporating the $\sinh^{-1}$ transformation, the final input becomes $\tilde{\mathbf{x}}_{\mathrm{P}_i} = \sinh^{-1}\left(\tilde{\mathbf{x}}_{\mathrm{P}_i}\right)$. During denormalization, the inverse transformation is applied:

$$\hat{\mathbf{x}}_{\mathrm{P}_{i+1}} = \left(\sqrt{\sigma_i^2 + \epsilon}\right) \cdot \mathcal{T}^{-1}\left(\widehat{\tilde{\mathbf{x}}}_{\mathrm{P}_{i+1}}\right) + \mu_i, \tag{5}$$

where $\widehat{\tilde{\mathbf{x}}}_{\mathrm{P}_{i+1}}$ denotes the model prediction in the normalized space before inverse transformation and $\mathcal{T}^{-1}$ represents the inverse of the transformation used during normalization (identity or $\sinh$), enabling recovery of the original scale.

3 Experiments

We conduct experiments to assess how these normalization strategies affect Transformer-based time-series forecasting models under efficient causal training. Although the models are trained with the pinball loss (Auer et al., 2025; Liu et al., 2025; Ansari et al., 2025) to estimate quantiles, we report results using only the 0.5 quantile (median) as a point forecast for clarity. Furthermore, we restrict our study to univariate time-series; however, the presented strategies can extend to multivariate settings by computing statistics independently for each channel. We use a 300M-parameter Transformer architecture with patch-based inputs, described in Appendix A.4. The models are trained on a large and diverse corpus of real-world time-series from the GIFT pretraining (Aksu et al., 2024) and UTSD (Liu et al., 2024) datasets, complemented with synthetic data, comprising approximately 715M patches, training and dataset details are provided in Appendix A.5. Evaluation is performed on test sets drawn either from the same distribution as the training data (using evaluation segments from UTSD and newly generated synthetic signals) or from distinct distributions (GIFT-Eval), across multiple context lengths $(128, 256, 512)$ and forecasting horizons $(32, 64, 96, \ldots, 512)$, yielding approximately $115{,}000$ distinct signals with our pipeline. During autoregressive inference, the normalization procedure is identical to that used during training (see Appendix A.6).

3.1 Training Convergence

Figure 1 presents the training loss trajectories. At initialization, Causal and Causal+$\sinh^{-1}$ achieve the lowest losses, followed by RevIN and RevIN+$\sinh^{-1}$, whereas Prefix@k and Prefix@k+$\sinh^{-1}$ exhibit substantially higher initial losses. Forecasts produced by causal-based strategies better match the local scale of each ground-truth patch, as they rely on patch-specific statistics, which likely accounts for their lower initial loss. In contrast, prefix-based strategies are more sensitive to non-stationarities in the training data: their forecasts are scaled according to the first k patches, which may differ from the scale of subsequent patches, leading to higher initial losses. Vanilla strategies occupy an intermediate position, with initial losses higher than those of causal-based strategies but lower than those of prefix-based strategies. As training progresses, the ranking changes:

RevIN+$\sinh^{-1}$ and RevIN reach the lowest minima as it can be seen in Figure 2. The Causal+$\sinh^{-1}$ and Prefix@k strategies converge to a comparable secondary plateau. Finally, Causal stabilizes at a higher loss, while Prefix@k+$\sinh^{-1}$ exhibits both the highest loss and the greatest variability (see discussion in Appendix A.7).

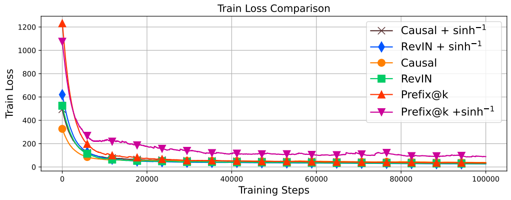


Figure 1: Training loss for first 100K steps.

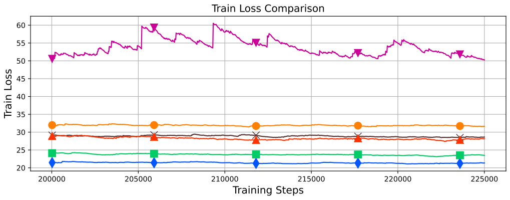


Figure 2: Training loss for 200K-225K steps.

3.2 Forecasting Performance

Across all test signals, context lengths, and forecasting horizons, two strategies seem to outperform the others: Causal+$\sinh^{-1}$ and Prefix@k, in terms of MAE, RMSE, and MASE (Mean Absolute Scaled Error; Eq. 24) as reported in Table 1. With respect to MAE and RMSE, Causal+$\sinh^{-1}$ achieves the strongest overall performance across all context lengths, with Prefix@k ranking second. In contrast, Prefix@k attains the best MASE results, followed closely by Causal+$\sinh^{-1}$ and Causal. The Prefix@k+$\sinh^{-1}$ strategy systematically underperforms across all metrics and context lengths. RevIN, RevIN+$\sinh^{-1}$, and Causal exhibit intermediate performance. Notably, the inclusion of the $\sinh^{-1}$ transformation improves the performance of Causal but degrades that of Prefix@k. Additional results such as pairwise skill scores are reported in Appendix A.8 and confirm our findings considering the leaders.

Table 1: Critical difference diagram (Demšar, 2006) results across context sizes w for each normalization strategy aggregated by datasets and forecasting horizons. The best overall mean rank for each metric and context size is shown in **bold**, and the second-best is underlined (lower ranks indicate better performance). Strategies connected within the same group (rank^{a}) are not statistically distinguishable according to the Nemenyi test with $\alpha = 0.05$. Best group is a, second-best group is b, and so on. Results are aggregated over all test signals and forecasting horizons.

	Prefix@k			Prefix@k+$\sinh^{-1}$			RevIN			RevIN+$\sinh^{-1}$			Causal			Causal+$\sinh^{-1}$		
w	128	256	512	128	256	512	128	256	512	128	256	512	128	256	512	128	256	512
MAE	2.7^{ab}	2.9^{b}	2.9^{b}	5.3^{c}	4.9^{d}	4.9^{c}	3.3^{ab}	3.5^{bc}	3.8^{bc}	3.4^{ab}	3.8^{bc}	4.0^{bc}	3.8^{b}	4.3^{cd}	3.8^{bc}	**2.5^{a}**	**1.5^{a}**	**1.7^{a}**
RMSE	**2.6^{a}**	2.9^{b}	2.8^{ab}	5.0^{c}	4.9^{d}	5.0^{c}	3.7^{ab}	3.7^{bc}	3.9^{bc}	3.4^{ab}	3.6^{bc}	3.9^{b}	3.8^{b}	4.4^{cd}	3.5^{b}	2.6^{a}	**1.5^{a}**	**1.8^{a}**
MASE	**2.2^{a}**	**1.8^{a}**	**2.2^{a}**	3.9^{b}	3.2^{b}	3.7^{bc}	4.4^{bc}	5.2^{c}	5.3^{d}	5.1^{c}	5.4^{c}	4.3^{cd}	2.6^{a}	3.1^{b}	2.7^{ab}	2.8^{a}	2.2^{ab}	2.9^{ab}

4 Conclusion

In this work, we propose a categorization of normalization strategies for large time-series models, highlighting the trade-off between causal integrity and forecasting accuracy. Our empirical analysis suggests two main insights for model design. First, we observe that although statistical leakage during training in RevIN and RevIN+$\sinh^{-1}$ can lead to lower training loss, this advantage does not systematically translate into improved downstream performance. This indicates that preserving causal integrity may contribute to better forecasting accuracy. Second, our results indicate that Causal+$\sinh^{-1}$ and Prefix@k are promising candidates for large-scale training. While they reach comparable asymptotic training losses, they exhibit complementary metric-dependent behavior: Causal+$\sinh^{-1}$ tends to perform better in terms of MAE and RMSE, whereas Prefix@k often achieves stronger results under MASE, with causal-based strategies yielding competitive performance. Together, these observations provide practical guidance for choosing normalization strategies that accommodate the diverse statistical characteristics of heterogeneous time-series data while preserving efficient causal training.

REFERENCES

Ihab Ahmed, Denis Krompaß, Cheng Feng, and Volker Tresp. A Comparative Study on How Data Normalization Affects Zero-Shot Generalization in Time Series Foundation Models, December 2025.

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, 2024. URL https://arxiv.org/abs/2410.10393.

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, 2024. URL https://arxiv.org/abs/2403.07815.

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, 2025. URL https://arxiv.org/abs/2510.15821.

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, 2025. URL https://arxiv.org/abs/2505.23719.

Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean Ogier 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, 2025. URL https://arxiv.org/abs/2505.14766.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting, 2024. URL https://arxiv.org/abs/2310.10688.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7(1):1–30, 2006.

Cheng Feng, Long Huang, and Denis Krompass. General time transformer: an encoder-only foundation model for zero-shot multivariate time series forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, pp. 3757–3761, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704369. doi: 10.1145/3627673.3679931. URL https://doi.org/10.1145/3627673.3679931.

Wei Gao, Xinyu Zhou, Peng Sun, Tianwei Zhang, and Yonggang Wen. Rethinking key-value cache compression techniques for large language model serving, 2025. URL https://arxiv.org/abs/2503.24000.

Lars Graf, Thomas Ortner, Stanisław Woźniak, and Angeliki Pantazi. Flowstate: Sampling rate invariant time series forecasting, 2025. URL https://arxiv.org/abs/2508.05287.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, and Abhinav Pandey et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

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, 2025. URL http://arxiv.org/abs/2511.11698.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models, 2024. URL `https://arxiv.org/abs/2402.02368`.

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. URL `https://openreview.net/forum?id=Jbdc0vTOcol`.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, and Sam Altman et al. gpt-oss-120b and gpt-oss-20b model card, 2025. URL `https://arxiv.org/abs/2508.10925`.

Noam Shazeer. Glu variants improve transformer, 2020. URL `https://arxiv.org/abs/2002.05202`.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting, 2026. URL `https://arxiv.org/abs/2509.26468`.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts, 2025. URL `https://arxiv.org/abs/2409.16040`.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL `https://arxiv.org/abs/2104.09864`.

Bartosz Uniejewski and Rafał Weron. Efficient forecasting of electricity spot prices with expert and lasso models. *Energies*, 11:2039, 08 2018. doi: 10.3390/en11082039.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers, 2024. URL `https://arxiv.org/abs/2402.02592`.

A Appendix

The code is available at Github Anonymized URL.

A.1 Formalization of the Problem Setting

We consider the task of univariate time-series forecasting using patch-based representations. Let a time-series be partitioned into a sequence of patches

$$\mathbf{x} = (\mathbf{x}_{\mathrm{P}_1}, \mathbf{x}_{\mathrm{P}_2}, \dots, \mathbf{x}_{\mathrm{P}_\mathrm{N}}), \tag{6}$$

where each patch $\mathbf{x}_{\mathrm{P}_\mathrm{i}} \in \mathbb{R}^{\mathrm{L}}$ corresponds to a contiguous segment of length L. Given the first n patches, the objective is to predict the next patch $\mathbf{x}_{\mathrm{P}_{n+1}}$.

We denote by f_θ a causal forecasting model parameterized by θ. The prediction of the next patch is defined as

$$\hat{\mathbf{x}}_{\mathrm{P}_{n+1}} = f_{\theta,n}(\mathbf{x}_{\mathrm{P}_1}, \dots, \mathbf{x}_{\mathrm{P}_n}), \tag{7}$$

where causality means that hidden representations of each patch depend only on past ones.

During training, the model is optimized to minimize a loss function $\mathcal{L}$ over all patches in the sequence:

$$\min_\theta \, \mathbb{E}_{\mathbf{x}=(\mathbf{x}_{\mathrm{P}_1},\dots,\mathbf{x}_{\mathrm{P}_\mathrm{N}})\sim\mathcal{D}} \left[\frac{1}{\mathrm{N}-1} \sum_{n=1}^{\mathrm{N}-1} \mathcal{L}\big(f_{\theta,n}(\mathbf{x}_{\mathrm{P}_1}, \dots, \mathbf{x}_{\mathrm{P}_n}), \mathbf{x}_{\mathrm{P}_{n+1}}\big) \right], \tag{8}$$

where $\mathcal{D}$ denotes the training distribution.

Causal Self-Attention Model. For clarity, we instantiate f_θ as a single-head self-attention mechanism with causal masking. A similar analysis can be done for LSTM-based models, where the hidden state at each step depends only on past inputs, and the same normalization issues arise. The prediction for the next patch $\hat{\mathbf{x}}_{\mathrm{P}_{n+1}}$ is computed as

$$f_{\theta,n}(\mathbf{x}_{\mathrm{P}_1},\dots,\mathbf{x}_{\mathrm{P}_n}) = \sum_{i=1}^{n} \alpha_{n,i}\, W_V \mathbf{x}_{\mathrm{P}_i}, \tag{9}$$

where the attention weights are defined as

$$\alpha_{n,i} = \frac{\exp\big((W_Q \mathbf{x}_{\mathrm{P}_n})^\top (W_K \mathbf{x}_{\mathrm{P}_i})\big)}{\sum_{j=1}^{n} \exp\big((W_Q \mathbf{x}_{\mathrm{P}_n})^\top (W_K \mathbf{x}_{\mathrm{P}_j})\big)}. \tag{10}$$

Here $W_Q, W_K, W_V \in \mathbb{R}^{\mathrm{L}\times\mathrm{L}}$ are the query, key, and value projection matrices, and $\theta = \{W_Q, W_K, W_V\}$.

Then, when forecasting the subsequent patch $\hat{\mathbf{x}}_{\mathrm{P}_{\mathrm{n}+2}}$, the model computes

$$f_{\theta,n+1}(\mathbf{x}_{\mathrm{P}_1},\dots,\mathbf{x}_{\mathrm{P}_{n+1}}) = \sum_{i=1}^{n+1} \alpha_{n+1,i}\, W_V \mathbf{x}_{\mathrm{P}_i}, \tag{11}$$

with updated attention weights

$$\alpha_{n+1,i} = \frac{\exp\big((W_Q \mathbf{x}_{\mathrm{P}_{n+1}})^\top (W_K \mathbf{x}_{\mathrm{P}_i})\big)}{\sum_{j=1}^{n+1} \exp\big((W_Q \mathbf{x}_{\mathrm{P}_{n+1}})^\top (W_K \mathbf{x}_{\mathrm{P}_j})\big)}. \tag{12}$$

Notably, the previously computed projections

$$(W_K \mathbf{x}_{\mathrm{P}_i},\, W_V \mathbf{x}_{\mathrm{P}_i}) \quad \text{for } i = 1,\dots,n \tag{13}$$

can be reused, enabling efficient incremental computation.

Furthermore, transformers architectures can perform all predictions during efficient causal training in parallel using matrix operations and causal masking during training, making it very efficient during training.

$$\underbrace{\begin{pmatrix} \hat{\mathbf{x}}_2 \\ \hat{\mathbf{x}}_3 \\ \vdots \\ \hat{\mathbf{x}}_{n+1} \end{pmatrix}}_{\in\mathbb{R}^{n\times\mathrm{L}}} = \mathrm{RowSoftmax}\left(\underbrace{\begin{pmatrix} W_Q\mathbf{x}_1 \\ W_Q\mathbf{x}_2 \\ \vdots \\ W_Q\mathbf{x}_n \end{pmatrix}}_{\mathbb{R}^{n\times\mathrm{L}}} \underbrace{\begin{pmatrix} W_K\mathbf{x}_1 \\ W_K\mathbf{x}_2 \\ \vdots \\ W_K\mathbf{x}_n \end{pmatrix}^{\top}}_{\mathbb{R}^{\mathrm{L}\times\mathrm{n}}} \odot \left(-\infty \cdot \underbrace{\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}}_{\mathbb{R}^{n\times n}} \right)\right) \underbrace{\begin{pmatrix} W_V\mathbf{x}_1 \\ W_V\mathbf{x}_2 \\ \vdots \\ W_V\mathbf{x}_n \end{pmatrix}}_{\mathbb{R}^{n\times\mathrm{L}}}$$

RowSoftmax applies the softmax operation independently to each row. The strictly lower-triangular mask enforces causality by preventing each query from attending to future patches.

This formulation naturally extends to multi-head attention and deeper Transformer architectures.

Normalization in the Patch Space. In practice, forecasting is commonly performed in the normalized space. Given the first n patches, we define the empirical mean and variance as

$$\mu_n = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{\mathrm{L}}\sum_{j=1}^{\mathrm{L}} \mathbf{x}_{\mathrm{P}_{i,j}}\right), \quad \sigma_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{\mathrm{L}}\sum_{j=1}^{\mathrm{L}} (\mathbf{x}_{\mathrm{P}_{i,j}} - \mu_n)^2\right). \tag{14}$$

Each patch is then normalized as

$$\tilde{\mathbf{x}}_{\mathrm{P}_i}^{(n)} = \frac{\mathbf{x}_{\mathrm{P}_i} - \mu_n}{\sqrt{\sigma_n^2 + \epsilon}}, \tag{15}$$

where $\epsilon > 0$ ensures numerical stability.

The model predicts the next normalized patch:

$$\hat{\tilde{\mathbf{x}}}_{\mathrm{P}_{n+1}} = f_{\theta,n}(\tilde{\mathbf{x}}_{\mathrm{P}_1}^{(n)},\dots,\tilde{\mathbf{x}}_{\mathrm{P}_n}^{(n)}), \tag{16}$$

and the prediction is mapped back to the original space via

$$\hat{\mathbf{x}}_{\mathrm{P}_{n+1}} = \hat{\tilde{\mathbf{x}}}_{\mathrm{P}_{n+1}} \cdot \sqrt{\sigma_n^2 + \epsilon} + \mu_n. \tag{17}$$

Impact of Normalization on Causal Computation. When using normalized inputs, the attention computation becomes

$$f_{\theta,n}(\tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_1}, \dots, \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_n}) = \sum_{i=1}^{n} \tilde{\alpha}_{n,i} \, W_V \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_i}, \tag{18}$$

with

$$\tilde{\alpha}_{n,i} = \frac{\exp\left((W_Q \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_n})^\top (W_K \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_i})\right)}{\sum_{j=1}^{n} \exp\left((W_Q \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_n})^\top (W_K \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_j})\right)}. \tag{19}$$

Crucially, when a new patch $\mathbf{x}_{\mathrm{P}_{n+1}}$ becomes available, the normalization statistics change from (μ_n, σ_n^2) to $(\mu_{n+1}, \sigma_{n+1}^2)$. As a consequence, all previously normalized patches must be recomputed:

$$\tilde{\mathbf{x}}^{(n+1)}_{\mathrm{P}_i} \neq \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_i}, \quad \forall i \leq n. \tag{20}$$

Therefore, previously computed key and value projections

$$(W_K \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_i}, \, W_V \tilde{\mathbf{x}}^{(n)}_{\mathrm{P}_i}) \quad \text{for } i = 1, \dots, n \tag{21}$$

cannot be reused.

This dependency breaks the efficient incremental computation property during training with Transformers. Hence, standard normalization schemes are fundamentally incompatible with causal pretraining unless one accepts either statistical leakage or modified normalization strategies.

This observation motivates the normalization strategies studied in the main paper.

A.2 Prefix Normalization Discussion

For Prefix normalization (Prefix@k and Prefix@k+sinh^{-1}), we evaluate a single prefix length of $k = 8$ patches. This value is chosen as a practical compromise between obtaining sufficiently stable normalization statistics and limiting the prefix window to reduce the risk of distribution shifts within the context. Nevertheless, the choice of k remains an important hyperparameter, and a systematic investigation of its impact on convergence and forecasting performance is left for future work.

During inference, normalization statistics can be computed either using the first k patches, in order to remain consistent with the training procedure, or using the full context (vanilla strategy). In the main experiments, we adopt the k-patch statistics (allowing the use of key–value caching) and additionally evaluate the use of full-context statistics at inference time. The results indicate that this alternative does not lead to performance improvements for either Prefix@k or Prefix@k+sinh^{-1}. Specifically, Prefix@k achieves the best overall performance, followed by its full-context variant Prefix@k V2, then Prefix@k+sinh^{-1} and its corresponding full-context variant Prefix@k V2+sinh^{-1}, as reported in Table 2.

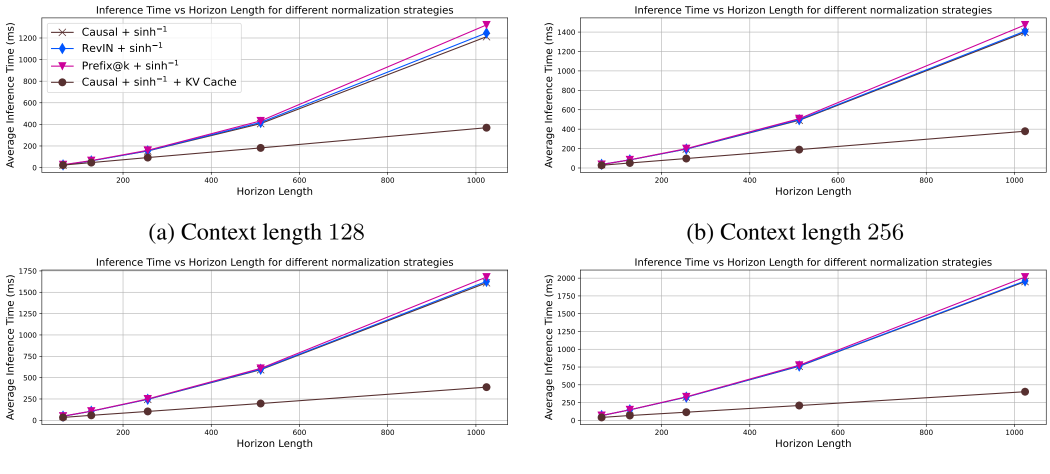
Table 2: Critical difference diagram (Demšar, 2006) results across context sizes w for Prefix normalization strategies aggregated by datasets and forecasting horizons. The best overall mean rank for each metric and context size is shown in **bold**, and the second-best is $\underline{\text{underlined}}$ (lower ranks indicate better performance). Strategies connected within the same group (rank^{a}) are not statistically distinguishable according to the Nemenyi test with $\alpha = 0.05$. Best group is a, second-best group is b, and so on. Results are aggregated over all test signals and forecasting horizons.

	Prefix@k				Prefix@k V2				Prefix@k+sinh^{-1}				Prefix@k V2+sinh^{-1}			
w	128	256	512	1024	128	256	512	1024	128	256	512	1024	128	256	512	1024
MAE	$\underline{1.7}^{\mathrm{a}}$	**1.3**$^{\mathrm{a}}$	**1.1**$^{\mathrm{a}}$	**1.3**$^{\mathrm{a}}$	**1.3**$^{\mathrm{a}}$	$\underline{1.7}^{\mathrm{a}}$	$\underline{2.1}^{\mathrm{b}}$	$\underline{1.9}^{\mathrm{a}}$	3.7^{b}	3.4^{b}	3.1^{c}	3.1^{b}	3.3^{b}	3.6^{b}	3.7^{c}	3.7^{b}
RMSE	$\underline{1.6}^{\mathrm{a}}$	**1.3**$^{\mathrm{a}}$	**1.1**$^{\mathrm{a}}$	**1.2**$^{\mathrm{a}}$	**1.4**$^{\mathrm{a}}$	$\underline{1.7}^{\mathrm{a}}$	$\underline{2.1}^{\mathrm{b}}$	$\underline{2.1}^{\mathrm{b}}$	3.6^{b}	3.4^{b}	3.1^{c}	3.0^{c}	3.4^{b}	3.6^{b}	3.7^{c}	3.8^{d}
MASE	**1.4**$^{\mathrm{a}}$	**1.2**$^{\mathrm{a}}$	**1.7**$^{\mathrm{a}}$	**1.3**$^{\mathrm{a}}$	$\underline{1.6}^{\mathrm{a}}$	$\underline{2.2}^{\mathrm{b}}$	$\underline{2.2}^{\mathrm{ab}}$	$\underline{2.5}^{\mathrm{b}}$	3.6^{b}	2.7^{b}	2.8^{bc}	2.9^{bc}	3.4^{b}	4.0^{c}	3.3^{c}	3.4^{c}

A.3 Computational Overhead

The computational cost introduced by the different normalization strategies remains negligible relative to the overall model complexity. Given N patches in the context, Causal and Causal+sinh^{-1}

require the computation of N means and variances, whereas Prefix@k, Prefix@k+$\sinh^{-1}$, RevIN, and RevIN+$\sinh^{-1}$ rely on a single mean and variance estimated over the entire context. Importantly, Causal and Causal+$\sinh^{-1}$ enable the use of the key–value caching mechanism at inference time without additional assumptions, thereby avoiding the recomputation of key and value projections for past patches and significantly reducing inference latency, as illustrated in Figure 3. Vanilla-based strategies, for each new patch appended to the context, require recomputing the normalization statistics and re-normalizing all past patches, which prevents the use of key–value caching. However, it is possible to consider some relaxations for using key–value caching with these strategies, such as editing the normalization statistics not at every step but only after a certain number of new patches, or using the statistics from the last steps for a certain number of new patches, which would introduce some approximation but could still yield significant computational savings.



(a) Context length 128

(b) Context length 256

(c) Context length 512

(d) Context length 1024

Figure 3: Inference time versus horizon length h, comparing normalization strategies and the use of key–value caching in Causal+$\sinh^{-1}$ across different context lengths. Similar behavior is observed without the $\sinh^{-1}$ transformation.

A.4 Model Architecture

We employ a Transformer architecture optimized for patch-based time-series forecasting. The model is trained to generate probabilistic forecasts, outputting the median alongside a set of predictive quantiles to characterize uncertainty.

The core architecture consists of stacked layers of multi-head self-attention and feed-forward networks. We use SwiGLU activation functions within the FFN blocks (Shazeer, 2020) and incorporate Rotary Positional Embeddings (Su et al., 2023) to inject relative temporal information into the attention mechanism. Standard residual connections are used throughout the stack, and temporal causality is strictly enforced via a triangular attention mask.

Table 3: Model configuration

Component	**Value**
Patch size L	32 observations
Max context N	32 patch (1024 observations)
Forecast horizon	1 patch (32 observations)
Quantiles $\mathcal{Q}$	$\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$
Model dimension	2048
Number of layers	6
Attention heads per layer	64
Head dimension	32
Parameters	$\approx$ 300M

A.5 Training

A.5.1 Training Data

Training such a model requires a large and heterogeneous corpus of time-series, enabling it to learn representations that generalize across signals rather than overfitting to individual instances. We therefore combine synthetic datasets, spanning diverse scales, dynamics, sampling rates, seasonalities, noise levels, and distribution shifts, with real-world data including industrial sensors, electricity load, and weather measurements.

Artificial Dataset. We synthesize approximately 1.4M time-series from basic families such as sinusoidal, linear, polynomial, and logarithmic functions, as well as their mixtures, using TSMixup (Ansari et al., 2024) to increase pattern diversity. In addition, we adopt the KernelSynth strategy from Ansari et al. (2024) to sample Gaussian-process signals with mixtures of kernels (e.g., RBF, periodic, linear), sweeping hyperparameters (amplitudes, frequencies/periods, length scales, noise) to cover a broad range of dynamics.

Real-World Dataset. For real-world signals, we leverage two datasets. The first is the Unified time-series Dataset (UTSD) (Liu et al., 2024), spanning seven domains: Energy, IoT, Nature, Web, Health, Transport, and Environment. UTSD is organized hierarchically to provide multiple benchmark subsets, where each smaller dataset is a strict subset of a larger one, enabling scalable training and evaluation. It aggregates publicly available time-series curated from diverse research groups and providers. For training, we use UTSD-12G, which yields approximately 19M signals under our preprocessing pipeline.

The second dataset is the GIFT-Eval pretraining corpus (Aksu et al., 2024), aligned with the GIFT-Eval benchmark but constructed to avoid data leakage with respect to the benchmark train/test split, enabling a fair evaluation of large models on GIFT-Eval. The dataset contains approximately 71 univariate and 17 multivariate time-series datasets from various domains and frequencies. After preprocessing, this yields approximately 1.2M univariate time-series.

A.5.2 Training Objective - Multi-Quantile (Pinball) Loss

The model is trained not only to forecast the next patch but also to estimate predictive quantiles (Auer et al., 2025; Liu et al., 2025; Ansari et al., 2025). We output quantiles $\mathcal{Q} = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, where the median ($q$=0.5) serves as the point forecast, and the remaining quantiles provide uncertainty bands.

During training, the model takes as input a sequence of N patches $\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_\mathrm{N}}$. For each position $i \in \{1, \ldots, \mathrm{N}\}$, it outputs quantile predictions for the next patch $\mathbf{x}_{\mathrm{P}_{i+1}}$, which is conditioned on the past patches $\mathbf{x}_{\mathrm{P}_1}, \ldots, \mathbf{x}_{\mathrm{P}_i}$. We denote the prediction at quantile q by $\hat{\mathbf{x}}^{(q)}_{\mathrm{P}_{i+1}} \in \mathbb{R}^{\mathrm{L}}$.

Then, the training objective is the multi-quantile (pinball) loss. Aggregated over positions, patch elements, and quantiles, it is defined as:

$$\mathcal{L} = \frac{1}{\mathrm{N}\,\mathrm{L}\,|\mathcal{Q}|} \sum_{i=1}^{\mathrm{N}} \sum_{t=1}^{\mathrm{L}} \sum_{q \in \mathcal{Q}} \rho_q\Big(\mathbf{x}_{\mathrm{P}_{i+1},t} - \hat{\mathbf{x}}^{(q)}_{\mathrm{P}_{i+1},t}\Big) . \tag{22}$$

where the pinball loss ρ_q is defined as:

$$\rho_q(u) = \begin{cases} q\,u, & u \geq 0, \\ (q-1)\,u, & u < 0. \end{cases} \tag{23}$$

A.5.3 Setup

Training was performed on 4 NVIDIA V100 GPUs. The implementation uses PyTorch Lightning with Distributed Data Parallel. We have a global batch size of 1024 and train for 225k steps in total. We use the AdamW optimizer, a learning rate starting at 10^{-5} with a linear warm-up over 10k steps until 5.10^{-4}, followed by cosine decay to 10^{-5} until 150k steps, and then a constant learning rate until the 225k steps.

A.6 Inference Loop

During inference, forecasts are generated autoregressively, patch by patch. Given an initial context of N patches, the model predicts the next patch, which is appended to the context and used to generate subsequent predictions. At each step, we apply the same normalization strategy as in training to ensure methodological consistency.

For RevIN and RevIN+$\sinh^{-1}$, normalization statistics are computed over the entire current context. After appending a newly predicted patch, the context is updated and the statistics are recomputed before the next prediction.

For Prefix@k and Prefix@k+$\sinh^{-1}$, statistics are computed over the first k patches of the context. When new patches are appended, the statistics remain unchanged.

For Causal and Causal+$\sinh^{-1}$, statistics are computed in a causal manner for each patch. When a new patch is added to the context, its normalization statistics are computed using only preceding patches.

This procedure is repeated until the desired forecast horizon is reached.

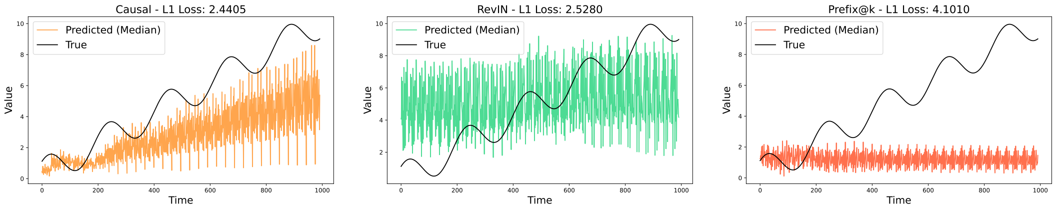
A.7 Extended Discussion on Training Dynamics

Causal normalization appears to exhibit faster initial convergence, as suggested by Figure 4(a). At initialization, the model is untrained and its parameters are randomly initialized, leading to essentially random patch predictions. Based on patch-specific statistics, the denormalization step in Causal and Causal+$\sinh^{-1}$ may help map these random patch predictions back to the original scale of the local data, which could explain the lower initial loss. As training progresses, however, the model must learn to cope with the non-smoothness introduced by patch-wise normalization, which may limit its asymptotic performance.

Vanilla normalization tends to begin with a higher loss, as this strategy is not specifically adapted to local statistics (Figure 4(b)). At initialization, the model's random patch predictions are denormalized using statistics computed over the entire context, which may not accurately reflect the local scale of each patch. Nevertheless, this strategy appears to achieve the lowest asymptotic loss in our experiments.

Prefix-based strategies generally start with the highest initial loss. At initialization, the model's random patch predictions are denormalized using statistics computed from a fixed prefix of the context. These statistics may not be representative of the local scale of subsequent patches, particularly when the signal exhibits non-stationarity or trends (Figure 4(c)). As a result, the denormalized predictions are expressed in the scale of the prefix, which can differ substantially from the scale of the local patch, potentially leading to a significant initial error.

Furthermore, prefix-based strategies appear to exhibit instability when combined with $\sinh^{-1}$. If the signal exhibits non-stationarity or a trend shift after the prefix window, later patches may become out-of-distribution relative to the prefix statistics. The $\sinh^{-1}$ transformation can compress these large values, resulting in a loss of feature resolution. Consequently, the model may struggle to map these compressed representations back to the original scale during denormalization, which can lead to increased forecasting error.



(a) Causal (b) Vanilla (c) Prefix@k

Figure 4: Forecast for each normalization strategy on a synthetic non-stationary sinusoidal signal at initialization.

A.8 More results

In this section, we provide additional experimental results. But first, we clarify the computation of the MASE metric in our experimental setting.

Given a context signal $\mathbf{x} \in \mathbb{R}^{\mathrm{w}}$, we define the future horizon of interest as the next h values, which we denote as the ground-truth observations as the target sequence $\mathbf{x}_{\text{tar}} \in \mathbb{R}^{h}$. The model's corresponding estimate for this horizon is denoted as the predicted sequence $\hat{\mathbf{x}}_{\text{tar}} \in \mathbb{R}^{h}$. The Mean Absolute Scaled Error (MASE) is then computed as follows:

$$\text{MASE} = \frac{\frac{1}{h}\sum_{i=1}^{h} |\mathbf{x}_{\text{tar},i} - \hat{\mathbf{x}}_{\text{tar},i}|}{\frac{1}{\mathrm{w}-1}\sum_{j=2}^{\mathrm{w}} |\mathbf{x}_{j} - \mathbf{x}_{j-1}|}. \tag{24}$$

Since the scaling factor in the denominator is based on one-step differences within the context, whereas the numerator aggregates errors over a potentially long forecast horizon, the MASE naturally tends to exceed one in most cases, even when the model exhibits strong predictive performance.

We complement the critical difference diagrams presented in the main paper with pairwise skill scores (Shchur et al., 2026). The skill score of a given strategy s with respect to a reference strategy r is defined as

$$\text{SkillScore}(s, r) = 1 - \sqrt[D]{\prod_{d=1}^{D} \text{clip}\left(\frac{\text{Err}_{s,d}}{\text{Err}_{r,d}},\, l,\, u\right)}, \tag{25}$$

where $\text{Err}_{s,d}$ denotes the error of strategy s on dataset d, D is the total number of datasets, and $\text{clip}(\cdot, l, u)$ constrains the ratio to the interval $[l, u]$ in order to prevent extreme values from disproportionately influencing the aggregated score. Following Shchur et al. (2026), we set $l = 0.01$ and $u = 100$.

The skill score takes values in $(-\infty, 1]$: a score of 1 corresponds to perfect performance (i.e., zero error), a score of 0 indicates performance equivalent to the reference strategy, and negative values indicate worse performance than the reference. Thus, $\text{SkillScore}(s, r)$ quantifies the average relative error reduction achieved by strategy s compared to the reference strategy r across datasets.

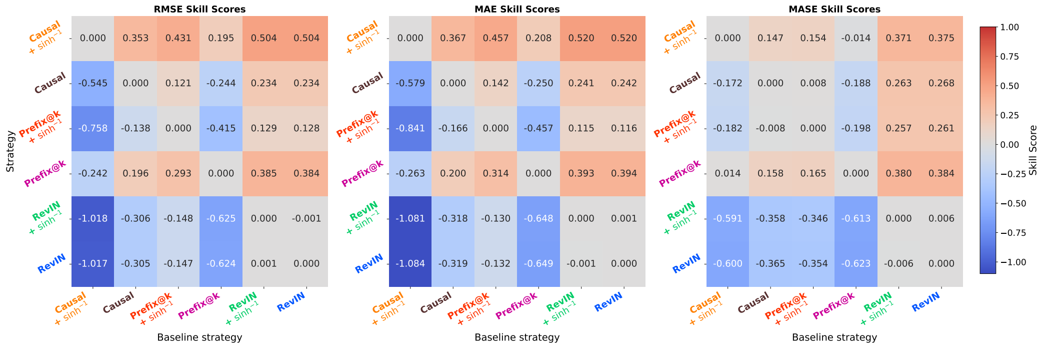
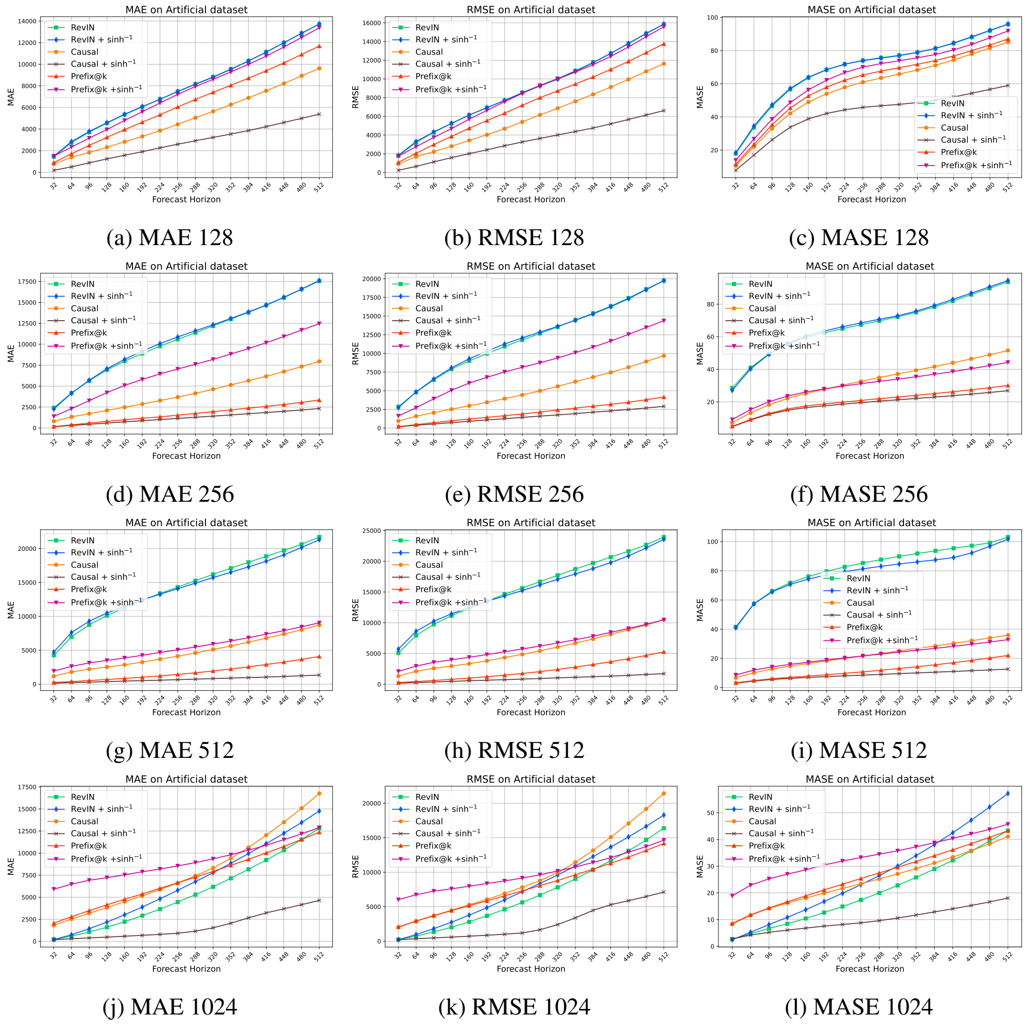


Figure 5: Pairwise skill scores for each normalization strategy, aggregated over context lengths, datasets, and forecasting horizons.

As shown in Figure 5 and consistent with the conclusions drawn from Table 1, Causal+$\sinh^{-1}$ achieves the best performance in terms of MAE and RMSE. However, when considering MASE, Prefix@k remains the leading approach, with only marginal improvements with respect to Causal+$\sinh^{-1}$. While RevIN and RevIN+$\sinh^{-1}$ previously exhibited intermediate performance, they now rank among the lowest-performing strategies under this aggregated skill score analysis.

Secondly, we report complementary comparisons by plotting forecasting performance across multiple prediction horizons, context lengths, and evaluation metrics.

During training, the maximum context length is set to 32 patches (=1024 observations). At evaluation time, when using a context length of 1024 observations and forecasting beyond the next 32 values, we do not truncate the signal when appending predictions to the existing context. Consequently, the effective context length increases with the prediction horizon and may exceed 1024. This experimental setting enables us to analyze model behavior when the effective context length surpasses the maximum sequence length encountered during training.



(a) MAE 128

(b) RMSE 128

(c) MASE 128

(d) MAE 256

(e) RMSE 256

(f) MASE 256

(g) MAE 512

(h) RMSE 512

(i) MASE 512

(j) MAE 1024

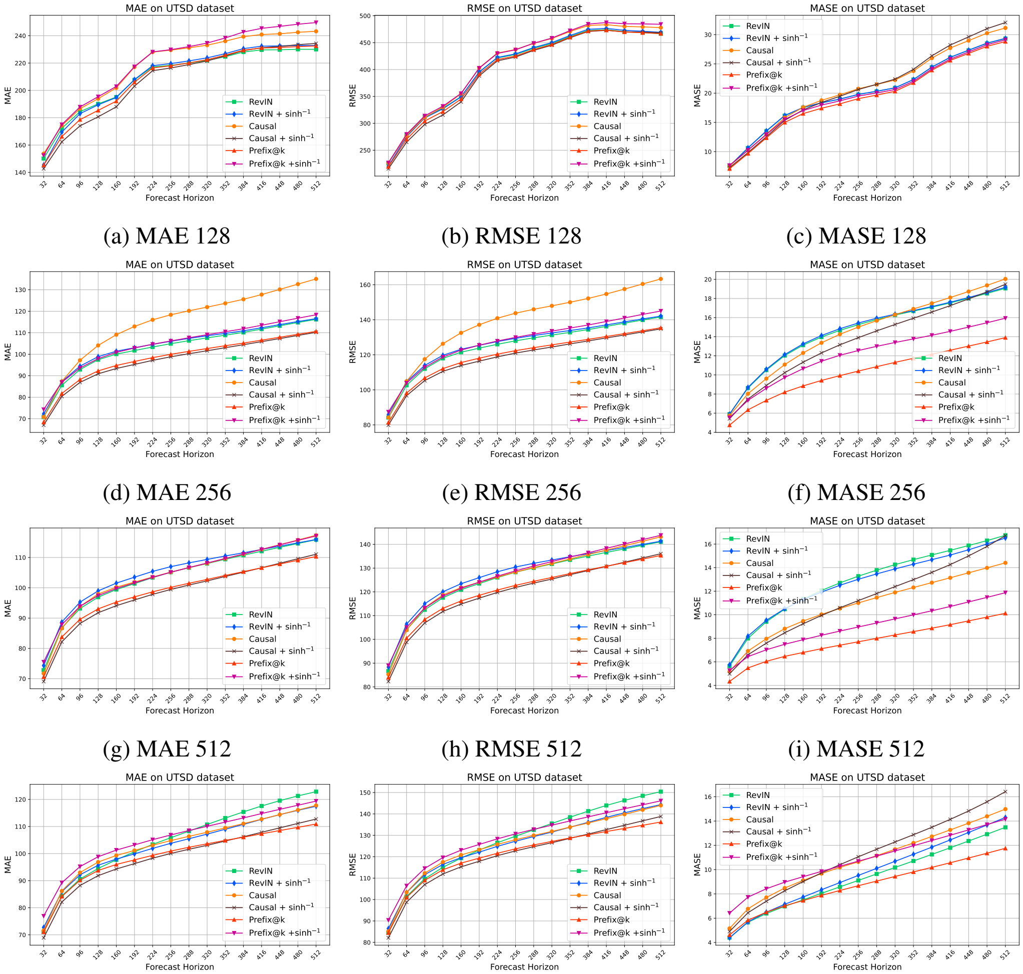
(k) RMSE 1024

(l) MASE 1024

Figure 6: Forecasting performance across prediction horizons on the Synthetic dataset for four context lengths (128, 256, 512, 1024).

On the synthetic benchmark (Figure 6), Causal+sinh^{-1} achieves the strongest overall performance across all context lengths and evaluation metrics. Causal exhibits stable intermediate performance, with a slight degradation at a context length of 1024. For context lengths of 256 and 512, Prefix@k performs competitively with Causal+sinh^{-1}. For context lengths below 1024, RevIN and RevIN+sinh^{-1} yield the weakest performance across all metrics. However, when the effective context length exceeds 1024 during inference, RevIN and RevIN+sinh^{-1} achieve performance comparable to most other methods, except for Causal+sinh^{-1}, which remains superior. Finally, Prefix@k+sinh^{-1} generally exhibits poor performance across all settings.

Results on the UTSD-12G benchmark (Figure 7) reveal a more nuanced performance profile. In terms of MAE and RMSE, Causal+sinh^{-1} and Prefix@k consistently rank among the strongest methods across all context lengths. However, under the MASE metric, Causal+sinh^{-1} exhibits intermediate performance, whereas Prefix@k remains the top-performing approach.



(a) MAE 128 (b) RMSE 128 (c) MASE 128

(d) MAE 256 (e) RMSE 256 (f) MASE 256

(g) MAE 512 (h) RMSE 512 (i) MASE 512

(j) MAE 1024 (k) RMSE 1024 (l) MASE 1024

Figure 7: Forecasting performance across prediction horizons on the UTSD-12G dataset for four context lengths (128, 256, 512, 1024).

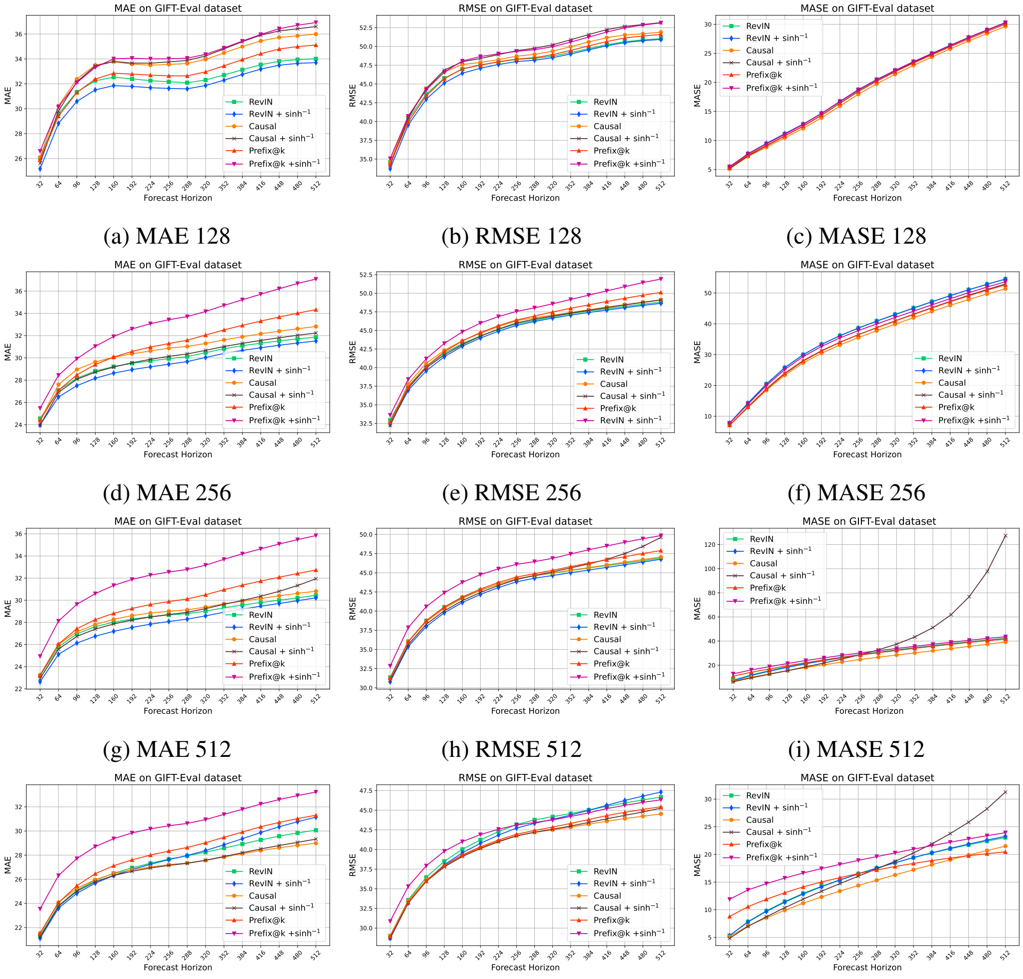
In contrast to the previous benchmarks, the GIFT-Eval benchmark (Figure 8) exhibits a shift in the optimal normalization strategy. For MAE and RMSE, the non-causal RevIN+$\sinh^{-1}$ and RevIN configurations achieve the strongest performance up to a context length of 1024, after which their performance slightly degrades.

However, under the MASE metric, Causal emerges as the best-performing strategy for the first time, while RevIN+$\sinh^{-1}$ and RevIN fall to intermediate ranks.

Notably, Causal+$\sinh^{-1}$ exhibits a pronounced scaling failure at context lengths of 512 and 1024: its performance deteriorates more rapidly than that of other strategies for horizons beyond 288, despite remaining competitive in terms of MAE and RMSE.

A.9 Example of normalized signal and error analysis

Figure 9 illustrates the behavior of the evaluated normalization strategies on a synthetic non-stationary sinusoidal signal. The RevIN strategy, leveraging global context statistics, produces the most stable representation with consistent zero-mean centering and unit-variance scaling across the entire sequence. However, as noted in Section 2, this result is achieved through non-causal information leakage during training. In contrast, the Prefix@k strategy achieves a perfect representation only for the patch k; once the distribution shifts, the normalization fails to track the evolving mean,



(a) MAE 128

(b) RMSE 128

(c) MASE 128

(d) MAE 256

(e) RMSE 256

(f) MASE 256

(g) MAE 512

(h) RMSE 512

(i) MASE 512

(j) MAE 1024

(k) RMSE 1024

(l) MASE 1024

Figure 8: Forecasting performance across prediction horizons on the GIFT-Eval dataset for four context lengths (128, 256, 512, 1024).

leading to significant bias in later segments. The Causal strategy preserves a well-scaled representation but introduces visible non-smoothness. Because statistics are updated at each patch, the resulting normalized signal appears disjointed. While this approach handles the distribution shift more robustly than the Prefix@k method, staying closer to a zero-mean representation, the lack of inter-patch continuity forces the model to manage these abrupt scaling transitions.

Figure 10 quantifies the normalization discrepancy between each training strategy and the global inference-time optimum. The RevIN approach exhibits a high initial error that monotonically diminishes to zero. This trajectory reflects the ”look-ahead” nature of the strategy: early patches are normalized using global future information during training that is not supposed to be available during inference, leading to a maximal mismatch that vanishes only at the final sequence element. Conversely, the Causal strategy begins with zero error—as the first patch is perfectly aligned with its only available context—but shows a divergent error trend. As the sequence progresses, the causal prefix statistics increasingly deviate from the more accurate statistics, reaching maximum discrepancy at the last patch. Finally, the Prefix@k strategy shows a sharp error divergence immediately following the initial prefix, highlighting its failure to track non-stationary signal shifts beyond the fixed prefix window.

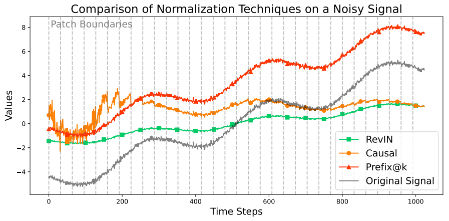


Figure 9: Example of normalization strategies on a simple increasing sinusoidal.

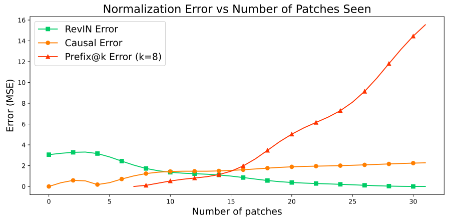


Figure 10: Normalization error against more accurate inference strategy.

A.10 Limitations of this Work

Our study focuses on univariate time-series, leaving the systematic extension to multivariate settings for future work. While the presented normalization srategies can extend to multivariate signals by computing statistics per channel, interactions across variables and cross-channel normalization effects may introduce additional dynamics that we do not explore here. Furthermore, our analysis is restricted to Transformer-based architectures. We do not study normalization strategies for state-space models (Graf et al., 2025) or xLSTM-based approaches (Auer et al., 2025) trained under comparable conditions. Our analysis is restricted to mean—variance-based normalization strategies, which remain the dominant choice in current large-scale time-series models. Alternative normalization families are therefore not considered in this work.