Toto: An Open Time Series Foundation Model Optimized for Observability

Abstract

We introduce Toto, a time series forecasting foundation model with 151 million parameters. Toto uses a modern decoder-only architecture coupled with architectural innovations designed to account for specific challenges found in multivariate observability time series data. Toto's pre-training corpus is a mixture of observability data, open datasets, and synthetic data, and is 4-10× larger than those of leading time series foundation models. We source observability data exclusively from our own telemetry and internal observability metrics. Extensive evaluations demonstrate that Toto achieves state-of-the-art performance on on established general purpose time series forecasting benchmarks. Toto's model weights, inference code, and evaluation scripts are available as open source under the Apache 2.0 License.

1. Introduction

Observability is the practice of collecting and analyzing data generated by distributed computer systems to detect, diagnose, and swiftly resolve performance and reliability issues (Majors et al., 2022). A major component of observability is monitoring time series metrics; observability tools generate massive and diverse sets of metrics that reflect a system's operational health over time. These metrics encompass a wide variety of indicators—such as memory usage, CPU load, disk I/O, network throughput, hit counts, error rates, and latency—that each exhibit distinct behavioral patterns, and collectively represent an important but under-studied subset of general time series data.

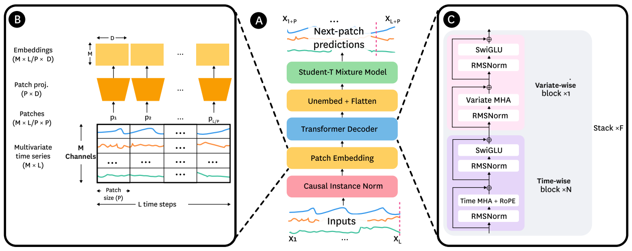


Figure 1: Overview of the Toto architecture. Ⓐ Multivariate input time series of L steps are scaled using **causal patch-based instance normalization**, transformed into patch embeddings, and passed through a decoder-only transformer stack. The transformed features are unembedded and passed through a **Student-T mixture model** (optimized via a **composite robust loss**) which generates probabilistic next-patch predictions. Ⓑ The patch embedding takes as input a time series of M variates by L time steps. It divides the time dimension into patches and projects these linearly into an embedding space . The resulting output is fed to the transformer decoder. Ⓒ The transformer stack features **proportional factorized attention**.

Accurately modeling observability metrics is essential for critical tasks like anomaly detection (Li et al., 2020) (e.g., identifying spikes in error rates) and predictive forecasting (Chang, 2017) (e.g., anticipating resource exhaustion or scaling needs). Observability data present challenges for traditional forecasting methods due to diversity, high-dimensionality, and complex distributional characteristics. Moreover, real-world observability systems routinely generate millions to billions of distinct time series (gra; clo, 2023; rec), rendering fine-tuning or supervised training of complex models per time series infeasible. These operational challenges suggest a compelling use case for zero-shot time series foundation models (FMs). However, we find that existing FMs (Liu et al., 2024a; Ansari et al., 2024; Das et al., 2024) trained for general-purpose forecasting struggle to generalize to observability data (see Section 4).

In this work, we focus on the unique challenges of modeling observability data, while accounting for the constraints of production settings. Our main contribution is **Toto** (**T**ime Series **O**ptimized **T**ransformer for **O**bservability) a novel open-weights time series forecasting foundation model, with a focus on zero-shot capabilities. **Toto uses a modern decoder-only architecture coupled with architectural innovations to account for the specific challenges found in observability time series data**: a novel per-variate patch-based causal scaling to address highly nonstationary sequences; proportional time-variate factorized attention to judiciously attend across a large number of covariates; and a Student-T mixture prediction head optimized via a robust composite loss to fit complex and highly skewed distributions. Toto's pretraining corpus contains 4-10× more unique data points than those of other time series FMs (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Shi et al., 2025), using a mix of domain-specific observability time series data, multi-domain public datasets, and synthetic data.

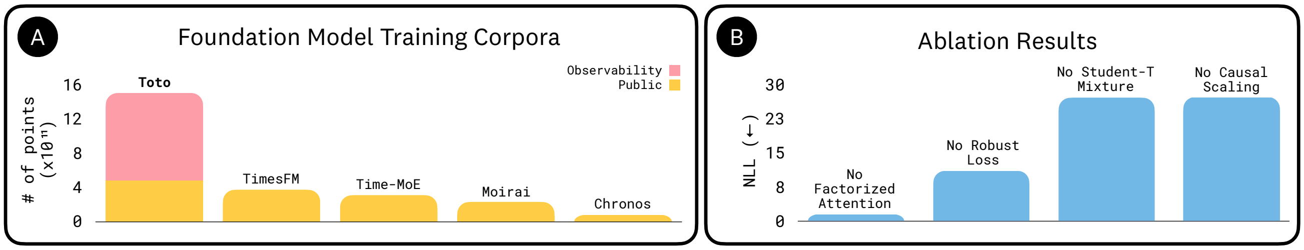


Figure 2: Ⓐ A comparison of the number unique time series points within the pretraining corpora of different time series foundation models. The scale of TOTO's training corpus is 4× that of TimesFM 1.0, 5× that of Time-MoE, 6.5× that of Moirai, and over 10× that of Chronos. Ⓑ Ablation results demonstrate the impact of four of TOTO's architectural components motivated by unique properties of observability time series data. Results report the change (relative to the full TOTO model) in negative log likelihood on held-out observability pretraining data when systematically disabling one component at a time. See Appendix D for details.

In our evaluations against leading foundation models and traditional time series forecasting baselines, **TOTO achieves state-of-the-art performance on both general-purpose and observability-oriented time series forecasting benchmarks.** On BOOM, a recent obesrvability-focused benchmark, TOTO achieves a 12% improvement in terms of CRPS compared to the next best method (see Section 4). TOTO also achieves the top position by a wide margin on two standard general-purpose time series benchmarks—GIFT-Eval and Long Sequence Forecasting (LSF)—implying our observability-focused design also pays dividends in other time series domains (Aksu et al., 2024; Zhou et al., 2020). We additionally perform ablations to motivate TOTO's architecture design (Fig. 2B). We provide TOTO's model weights, inference code, and evaluation scripts under a permissive (Apache 2.0) license available at (`https://github.com/XXXXXXX/toto`).

2. Related Work

By pre-training on large multi-domain datasets, several time series foundation models (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Shi et al., 2025; Garza and Mergenthaler-Canseco, 2023; Rasul et al., 2023; Gruver et al., 2023; Liu et al., 2024b; Chen et al., 2024; Goswami et al., 2024) have achieved impressive zero-shot prediction capabilities on general purpose time series benchmarks, eliminating the need for domain-specific training or fine-tuning. This approach is promising for observability workloads, as a single model can be deployed and horizontally scaled to provide low-latency and relatively low-cost zero-shot inference. Our evaluations indicate that TOTO outperforms existing time series foundation models by a wide margin on both public forecasting benchmarks (Section 4).

3. TOTO

3.1. Model architecture

Transformer models for time series forecasting have variously used encoder-decoder (Wu et al., 2021; Ansari et al., 2024; Zhou et al., 2020), encoder-only (Woo et al., 2024; Nie et al., 2023; Liu et al., 2024c), and decoder-only architectures (Das et al., 2024; Rasul et al., 2023). TOTO uses a decoder-only architecture (trained on a next-patch prediction task), as is has shown to scale well with respect to training efficiency when provided with sufficient data (Radford and Narasimhan, 2018; Radford et al., 2019). We use non-overlapping patch embeddings (Nie et al., 2023; Cordonnier et al., 2020; Dosovitskiy et al., 2021), with a patch of size $P = 64$, to project input time series of context length $L = 4096$ points to embeddings of size $64 \times D$ per variate, where $D = 768$ is the embedding dimension for our final model (Fig. 1B). We also utilize techniques demonstrated to yield performance and efficiency improvements in contemporary transformer literature, including pre-normalization (Xiong et al., 2020), RMSNorm (Zhang and Sennrich, 2019), SwiGLU feed-forward layers (Shazeer, 2020), and RoPE (Su et al., 2024) with XPOS (Sun et al., 2022) for improved extrapolation.

We further develop four specialized components purpose-built for handling multivariate observability time series data. Fig. 2B presents an ablation study that demonstrates the impact of these components.

Patch-based causal instance normalization to handle highly nonstationary data. To improve generalization across varying input scales, instance normalization is commonly applied prior to embedding time series data (for example, RevIN (Kim et al., 2022)). However, computing normalization statistics from the entire series would leak information from future time steps. This violates the causality of next patch prediction training and results in poor performance (see ablation in Appendix D). Das et al. (2024) normalize the entire series according to the statis-

		Zero Shot							*Baselines*		
Dataset	**Metric**	**Toto**	**Moirai$_{\text{Base}}$**	**TimesFM$_{\text{2.0}}$**	**Chronos$_{\text{Bolt-Base}}$**	**Timer**	**Time-MoE$_{\text{Base}}$**	**VisionTS**	**Auto-ARIMA**	**Auto-ETS**	**Auto-Theta**
Boom	**MASE ↓**	**0.617**	0.710	0.725	0.726	0.796	0.806	0.988	0.824	0.842	1.123
	CRPS ↓	**0.375**	0.428	0.447	0.451	0.639	0.649	0.673	0.736	1.975	1.018
	Rank ↓	**2.336**	4.253	5.155	5.447	9.370	9.381	10.317	9.16	10.956	11.712

Table 1: **Boom results.** Performance of Toto, other zero-shot models, and baselines. MASE and CRPS are normalized by the Seasonal Naive forecast and aggregated across tasks using shifted geometric mean. Rank is the mean rank across tasks with respect to CRPS. For model families with multiple sizes (Moirai, Chronos) we show the best-performing variant. Toto significantly outperforms other methods on all metrics. Additional results, including all model sizes evaluated as well as categorical breakdowns, are available in Appendix C.2. Key: **Best results**, Second-best results.

tics of the first patch. That approach preserves causality, but can be ineffective for highly nonstationary data with statistics that vary significantly over time, as is the case with observability data. To resolve these issues, we propose a novel per-patch normalization approach, where scaling factors for each patch are computed exclusively from the current patch and past data. Thus, our final approach predominantly preserves causality while substantially enhancing forecasting performance, particularly for highly nonstationary series. Additional technical and implementation details are provided in Appendix A.1.

Proportional factorized attention to judiciously capture variate interactions. We design Toto to natively handle multivariate forecasting by analyzing relationships in the time dimension ("time-wise" interactions) and the variate dimension ("variate-wise" interactions). While prior works that do not utilize variate-wise relationships (such as PatchTST (Nie et al., 2023) and TimesFM (Das et al., 2024)) can still achieve competitive performance on multivariate datasets, other studies (e.g. Woo et al. (2024)) have shown benefits from including variate-wise attention in ablations. However, observability metrics are often high-cardinality, multivariate time series, and a full attention schema simultaneously attending to both the time and variate dimensions can be computationally costly.

Drawing from our experience that time relationships are often more important than cross-variate relationships, we propose a relaxation of factorized attention. Factorized attention strictly alternates attention operations in the time and variate dimensions, allowing for time and variate mixing with lower algorithmic complexity (Zhang and Yan, 2023; Rao et al., 2021; Arnab et al., 2021). Our design provides more granular control over the relative proportion of time-wise and variate-wise interactions. Specifically, each transformer block has attention along only a single axis, and we can change the ratio of time-wise to variate-wise transformer blocks as a hyperparameter (as illustrated in Figure 1C). Toto uses an 11:1 ratio (11 time-wise transformer blocks followed by a single variate-wise transformer block), which we found via hyperparameter optimization (see Appendix A.6).

Student-T mixture model (SMM) head to model heavy-tailed observability time series. Producing probabilistic outputs is a critical feature of time series models in several domains, including observability (Zhu and Laptev, 2017; Lee et al., 2023; Hang et al., 2024). In order to produce probabilistic forecasts across the wide range of output distributions present in observability data, we employ a method based on Gaussian mixture models (GMMs), which can approximate any density function (Goodfellow et al., 2016). We found that fitting GMMs in the presence of the extreme outliers and high skew found in observability data leads to numerical instability in training, so we instead utilize a Student-T mixture model (SMM) of K distributions, which robustly generalizes GMMs (Peel and McLachlan, 2000) and has shown promise for modeling heavy-tailed financial time series (Meitz et al., 2018; WONG et al., 2009). In a contemporaneous work, Yao et al. (2025) also explored time series foundation models which model a Student-T mixture output. A mathematical formulation of of the mixture model, including equations and parameterizations, is provided in Appendix A.3.

Composite robust loss to stabilize training dynamics. Mixture models optimized via maximum likelihood are known to suffer from singularities (Bishop, 2006) and cluster collapse (Eisner). We use a composite loss formulation that we find, in practice, mitigates these effects. During training, we optimize a next-patch prediction task, where the model's objective is to predict the distribution of values in the next patch given all previous patches. Our training combines the standard negative log-likelihood loss, $\mathcal{L}_{\text{NLL}}$, and a general robust loss, $\mathcal{L}_{\text{Robust}(\alpha,\delta)}$ (Barron, 2017). The robust loss provides a unified framework that allows for smoothly interpolating between several common robust loss functions (Black and Anandan, 1996; Geman and Geman, 1986; Aubert et al., 1994; Sun et al., 2010; Jr. and and, 1978; Leclerc, 1989; Charbonnier et al., 1994; Huber, 1964; Zhang, 1997), using parameters $\alpha \in [-\infty, 2]$ and $\delta > 0$ (see Fig. 4). In our case, after hyperparameter optimization, we found the Cauchy loss ($\alpha = 0$) performed best in our setting. While the NLL loss utilizes the full probabilistic output of the model, the robust loss operates point-wise and measures the prediction er-

	Zero Shot					*Full Shot*				*Baselines*		
Metric	**Toto**	**Moirai$_{\text{Large}}$**	**TimesFM$_{2.0}$**	**Chronos$_{\text{Bolt-Base}}$**	**TabPFN-TS**	**TEMPO**	**TTM-R2**	**PatchTST**	**TFT**	**Auto-ARIMA**	**Auto-ETS**	**Auto-Theta**
MASE ↓	**0.673**	0.785	0.680	0.725	0.748	0.773	<u>0.679</u>	0.762	0.822	0.964	1.088	0.978
CRPS ↓	<u>0.437</u>	0.506	0.465	0.485	0.480	**0.434**	0.492	0.496	0.511	0.770	6.327	1.051
Rank ↓	**5.495**	10.330	8.412	<u>8.309</u>	8.402	8.897	10.103	10.268	11.629	21.608	25.134	24.134

Table 2: **GIFT-Eval results.** Toto's performance compared to other models reproduced from the GIFT-Eval leaderboard (gif).

ror between the predicted SMM mean and the ground truth data point. The final combined loss used for training Toto is: $\mathcal{L} = \lambda_{\text{NLL}} \cdot \mathcal{L}_{\text{NLL}} + (1 - \lambda_{\text{NLL}}) \cdot \mathcal{L}_{\text{Robust}(\alpha,\delta)}$, where $\lambda_{\text{NLL}} \in [0, 1]$ is a ratio tuned simultaneously with the robust loss hyperparameters, with optimal value $\lambda_{\text{NLL}} = 0.57$. Further details, including explicit definitions of each loss component, are provided in Appendix A.5.

3.2. Training data

We trained Toto with a dataset of approximately 2.36 trillion time series points, of which 1.59 trillion are non-repeated and non-synthetic. This is significantly larger than the pretraining corpora of existing time series foundation models (Fig. 2A). Critically, 43% of our training mixture contains anonymous observability metrics from a leading commercial observability platform. This data excludes any customer data and is sourced solely from the platform's own monitoring of internal systems. It consists only of numerical time series data. However, much of this data is sparse, noisy, or too granular or high in cardinality to be useful in its raw form. To curate a high-quality dataset, we sample queries based on quality and relevance signals from dashboards, monitor alerts, and notebooks built by domain experts using the platform.

Alongside the observability data, we include public time series datasets, in particular, the GIFT-Eval Pretrain (Aksu et al., 2024) and Chronos (Ansari et al., 2024) collections. Importantly, we remove the subset of the Chronos datasets that overlap with the GIFT-Eval benchmark in order to avoid any leakage from the test data. We also find that adding synthetic data improves model generalization and performance. For more details on the preparation of public, synthetic, and observability data, please see Appendix B.

4. Experiments

We evaluate Toto on three benchmarks: Boom, a recently released benchmark focused on observability (Datadog, 2025), GIFT-Eval, and LSF. We compare against a comprehensive set of methods, including zero-shot foundation models ('Zero Shot'), neural models trained on the target data ('Full Shot'), and classical supervised approaches ('Baselines'). Details of the inference settings and evaluation procedures for all models are described in Appendix C.

Boom. We evaluate Toto's zero-shot forecasting performance alongside other foundation models, (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Shi et al., 2025; Liu et al., 2024b; Chen et al., 2024), as well as full-shot statistical baselines. As shown in Table 1, Toto consistently outperforms other models, achieving 13.1% and 12.4% lower MASE and CRPS, respectively, than the next best (Moirai$_{\text{Base}}$), and a significantly lower rank (2.351 vs. 4.278).

GIFT-Eval. We evaluate Toto's zero-shot performance on general-purpose time series forecasting via the GIFT-Eval benchmark (Aksu et al., 2024). Toto achieves the top performance among all reported models, with an average ranking score of 5.495 as of May 2025. It achieves strong results both in point forecasting, with a MASE of 0.673, and probabilistic forecasting, with a CRPS of 0.437.. Notably, Toto is the top-performing method in spite of the fact that several competing models have known partial data leakage with the benchmark (Aksu et al. (2024)).

LSF. We evaluate Toto on the widely-used Long Sequence Forecasting (LSF) benchmark (Wu et al., 2021). Toto achieves state-of-the-art results in zero-shot evaluations, attaining the best performance on 8 out of 12 reported metrics when compared against other zero-shot methods, and the lowest average MAE and MSE, see Appendix C.3. We also explored the efficacy of fine-tuning Toto on the training splits of LSF and report the results in Table 10. We find that Toto achieves state-of-the-art results in full-shot evaluations, also attaining the best performance on 8 out of 12 reported metrics, and the lowest average MAE and MSE of all methods.

5. Conclusion

This work reframes time series forecasting through the lens of observability—a domain marked by scale, complexity, and real-world urgency. We presented Toto, a foundation model purpose-built to forecast multivariate observability metrics with zero-shot accuracy, which advances the frontier in zero-shot time series forecasting and sets new state-of-the-art results on Boom, GIFT-Eval, and LSF. By open-sourcing both model and benchmark, we hope to accelerate research to answer these and other open questions, contribute to the community, and to draw attention to an important real-world application.

References

C. Majors, L. Fong-Jones, and G. Miranda. *Observability Engineering*. O'Reilly Media, 2022. ISBN 9781492076414. URL https://books.google.com/books?id=KGZuEAAAQBAJ.

Ze Li, Qian Cheng, Ken Hsieh, Yingnong Dang, Peng Huang, Pankaj Singh, Xinsheng Yang, Qingwei Lin, Youjiang Wu, Sebastien Levy, and Murali Chintalapati. Gandalf: an intelligent, end-to-end analytics service for safe deployment in cloud-scale infrastructure. In *Proceedings of the 17th Usenix Conference on Networked Systems Design and Implementation*, NSDI'20, page 389–402, USA, 2020. USENIX Association. ISBN 9781939133137.

Emily Chang. Introducing metric forecasts for predictive monitoring in Datadog, December 2017. URL https://www.datadoghq.com/blog/forecasts-datadog/.

How we scaled our new Prometheus TSDB Grafana Mimir to 1 billion active series. URL https://grafana.com/blog/2022/04/08/how-we-scaled-our-new-prometheus-tsdb-grafana-mimir-to-1-billion-active-series/.

How Cloudflare runs Prometheus at scale, March 2023. URL https://blog.cloudflare.com/how-cloudflare-runs-prometheus-at-scale/.

Observability at Scale: How the Right Stack Can Help | Recurly. URL https://recurly.com/blog/observability-at-scale/.

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

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.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=jn2iTJas6h.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=Yd8eHMY1wz.

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 (ICLR)*, 2025. URL https://openreview.net/forum?id=e1wDDFmlVu. Spotlight Presentation.

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.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wan Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. 2020. URL https://api.semanticscholar.org/CorpusID:229156802.

Azul Garza and Max Mergenthaler-Canseco. Timegpt-1, 2023.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Hassen, Anderson Schneider, Sahil Garg, Alexandre Drouin, Nicolas Chapados, Yuriy Nevmyvaka, and Irina Rish. Lag-llama: Towards foundation models for time series forecasting. In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023. URL https://openreview.net/forum?id=jYluzCLFDM.

Nate Gruver, Marc Anton Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=md68e8iZK1.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: generative pre-trained transformers are large time series models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024b.

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. *arXiv preprint arXiv:2408.17253*, 2024. doi: 10.48550/arXiv.2408.17253.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: a family of open time-series foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. 2021. URL `https://openreview.net/forum?id=J4gRj6d5Qm`.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. 2023. URL `https://openreview.net/forum?id=Jbdc0vTOcol`.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. 2024c. URL `https://openreview.net/forum?id=JePfAI8fah`.

Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. URL `https://api.semanticscholar.org/CorpusID:49313245`.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL `https://api.semanticscholar.org/CorpusID:160025533`.

Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=HJlnC1rKPB`.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020. URL `https://openreview.net/forum?id=B1x8anVFPr`.

Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization. In *Advances in Neural Information Processing Systems 32*, Vancouver, Canada, 2019. URL `https://openreview.net/references/pdf?id=S1qBAf6rr`.

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

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.127063. URL `https://www.sciencedirect.com/science/article/pii/S0925231223011864`.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. A length-extrapolatable transformer. In *ACL 2023*, December 2022. URL `https://www.microsoft.com/en-us/research/publication/a-length-extrapolatable-transformer/`.

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`.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=vSVLM2j9eie`.

Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8844–8856. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/rao21a.html`.

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6816–6826, 2021. doi: 10.1109/ICCV48922.2021.00676.

Lingxue Zhu and Nikolay Laptev. Deep and confident prediction for time series at uber. In *2017 IEEE international conference on data mining workshops (ICDMW)*, pages 103–110. IEEE, 2017.

Cheryl Lee, Tianyi Yang, Zhuangbin Chen, Yuxin Su, and Michael R Lyu. Maat: Performance metric anomaly anticipation for cloud services with conditional diffusion. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 116–128. IEEE, 2023.

Haitian Hang, Xiu Tang, Jianling Sun, Lingfeng Bao, David Lo, and Haoye Wang. Robust auto-scaling with probabilistic workload forecasting for cloud databases. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 4016–4029. IEEE, 2024.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

D. Peel and G.J. McLachlan. Robust mixture modelling using the t distribution. *Statistics and Computing*, 10(4): 339–348, 2000.

Mika Meitz, Daniel P. A. Preve, and Pentti Saikkonen. A mixture autoregressive model based on student's t–distribution. *Communications in Statistics - Theory and Methods*, 52:499 – 515, 2018. URL https://api.semanticscholar.org/CorpusID:73615847.

C. S. WONG, W. S. CHAN, and P. L. KAM. A student t -mixture autoregressive model with applications to heavy-tailed financial data. *Biometrika*, 96(3):751–760, 2009. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/27798861.

Qingren Yao, Chao-Han Huck Yang, Renhe Jiang, Yuxuan Liang, Ming Jin, and Shirui Pan. Towards neural scaling laws for time series foundation models, 2025. URL https://arxiv.org/abs/2410.12360.

GIFT Eval - a Hugging Face Space by Salesforce. URL https://huggingface.co/spaces/Salesforce/GIFT-Eval.

Christopher M. Bishop. Mixture models and em. In *Pattern Recognition and Machine Learning*, pages 423–495. Springer, New York, 1 edition, 2006. ISBN 978-0387310732. URL https://link.springer.com/book/9780387310732.

Jason Eisner. Dynamics of optimizing gaussian mixture models. https://www.cs.jhu.edu/~jason/tutorials/GMM-optimization.html. Accessed: 2025-04-09.

Jonathan T. Barron. A more general robust loss function. *CoRR*, abs/1701.03077, 2017. URL http://arxiv.org/abs/1701.03077.

Michael J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996. ISSN 1077-3142. doi: https://doi.org/10.1006/cviu.1996.0006. URL https://www.sciencedirect.com/science/article/pii/S1077314296900065.

Donald Geman and Stuart Geman. Bayesian image analysis. In E. Bienenstock, F. Fogelman Soulié, and G. Weisbuch, editors, *Disordered Systems and Biological Organization*, pages 301–319, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg. ISBN 978-3-642-82657-3.

G. Aubert, M. Barlaud, L. Blanc-Feraud, and P. Charbonnier. A deterministic algorithm for edge-preserving computed imaging using Legendre transform . In *12th IAPR International Conference on Pattern Recognition, 1994*, volume 1, pages 188–191 vol.3, Los Alamitos, CA, USA, October 1994. IEEE Computer Society. doi: 10.1109/ICPR.1994.577154. URL https://doi.ieeecomputersociety.org/10.1109/ICPR.1994.577154.

Deqing Sun, Stefan Roth, and Michael J. Black. Secrets of optical flow estimation and their principles. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2432–2439, 2010. doi: 10.1109/CVPR.2010.5539939.

John E. Dennis Jr. and Roy E. Welsch and. Techniques for nonlinear least squares and robust regression. *Communications in Statistics - Simulation and Computation*, 7(4):345–359, 1978. doi: 10.1080/03610917808812083. URL https://doi.org/10.1080/03610917808812083.

Yvan G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3(1):73–102, 1989. ISSN 1573-1405. doi: 10.1007/BF00054839. URL https://doi.org/10.1007/BF00054839.

Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings 1994 International Conference on Image Processing, Austin, Texas, USA, November 13-16, 1994*, pages 168–172. IEEE Computer Society, 1994. doi: 10.1109/ICIP.1994.413553. URL https://doi.org/10.1109/ICIP.1994.413553.

Peter J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35:492–518, 1964. URL https://api.semanticscholar.org/CorpusID:121252793.

Zhengyou Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59–76, 1997. ISSN 0262-8856. doi: https://doi.org/10.1016/S0262-8856(96)01112-2. URL https://www.sciencedirect.com/science/article/pii/S0262885696011122.

Datadog. Boom: Benchmark of observability metrics. https://huggingface.co/datasets/Datadog/BOOM, 2025. Accessed: 2025-05-19.

B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

Bias correction in weighted variance. https://stats.stackexchange.com/a/61641, 2012. Accessed: 2025-04-09.

Romain Ilbert, Ambroise Odonnat, Vasilii Feofanov, Aladin Virmaux, Giuseppe Paolo, Themis Palpanas, and Ievgen Redko. SAMformer: Unlocking the potential of transformers in time series forecasting with sharpness-aware minimization and channel-wise attention. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=8kLzL5QBh2.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36:1181–1191, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.07.001. URL https://www.sciencedirect.com/science/article/pii/S0169207019301888.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan K Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tiDE: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=pCbC3aQB5W.

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. URL http://jmlr.org/papers/v21/19-820.html.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. *CoRR*, abs/1907.10902, 2019. URL http://arxiv.org/abs/1907.10902.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Minicpm: Unveiling the potential of small language models with scalable training strategies, 2024. URL https://arxiv.org/abs/2404.06395.

Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library. https://github.com/facebookresearch/xformers, 2022.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision, 2024. URL https://arxiv.org/abs/2407.08608.

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

Zongzhe Xu, Ritvik Gupta, Wenduo Cheng, Alexander Shen, Junhong Shen, Ameet Talwalkar, and Mikhail Khodak. Specialized foundation models struggle to beat supervised baselines. *arXiv preprint arXiv:2411.02796*, 2024.

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, 2024. URL https://arxiv.org/abs/2310.01728.

Tian Zhou, PeiSong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all:power general time series analysis by pretrained lm, 2023. URL `https://arxiv.org/abs/2302.11939`.

Musleh Alharthi and Ausif Mahmood. xlstmtime : Long-term time series forecasting with xlstm, 2024. URL `https://arxiv.org/abs/2407.10240`.

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

A. Model architecture details

A.1. Input/output scaling

For a timestep t, we define the causal mean $\hat{\mu_t}$ and causal variance $\hat{s_t}$ as:

$$\hat{\mu_t} = \frac{\sum_{i=1}^{t} w_i x_i}{\sum_{i=1}^{t} w_i}, \quad \hat{s_t} = \sqrt{\frac{\sum_{i=1}^{t} w_i (x_i - \hat{\mu_t})^2}{\sum_{i=1}^{t} w_i - 1} + 0.1}\,,$$

where x_i represents the input value and w_i the corresponding weight at timestep i. We set the weight to 0 for padding positions and 1 for all other positions. We add a minimum value of 0.1 to the causal standard deviation, in order to limit the amount of scaling applied to any particular value and avoid numerical overflow. Timesteps within each patch share the normalization values determined by the final timestep of that patch. As computing causal statistics for every subsequence would have suboptimal $O(n^2)$ complexity in the sequence dimension, we instead use Welford's online algorithm (Welford, 1962), a method that provides numerically stable variance calculations in $O(n)$ time. We gain further efficiency with a vectorized adaptation of the algorithm, allowing for GPU parallelism.

This normalization approach preserves causality and is more adaptive than a fixed per-variate scaling factor. However, in practice, we still find training instability in the presence of extreme nonstationarity. To address this, we relax our requirement of strict causality and introduce a simple clipping mechanism using variate-level statistics. We constrain $\hat{s_t}$ within a range defined by a minimum value, constant exponent κ, and the full-variate variance s: $\max(0.1, s \times 10^{-\kappa}) \leq \hat{s_t} \leq s \times 10^{\kappa}$, (we set $\kappa = 10$ in practice). At inference we compute statistics based solely on the historical context.

To ensure numerical stability, we compute the $\hat{\mu_t}$ and $\hat{s_t}$ using an efficient vectorized implementation (Listing 1) of Welford's online algorithm (Welford, 1962), incorporating Bessel's correction to provide an unbiased estimator of variance, as described in Option A of (sta, 2012). We stabilize training against extreme outliers by incorporating weak information from the global statistics.

```
def compute_causal_statistics(
    data: torch.Tensor,
    weights: torch.Tensor,
    minimum_scale: float,
) -> Tuple[torch.Tensor, torch.Tensor]:
    # Compute causal means at each time
    step
    weighted_data = weights * data
    cum_weights = torch.cumsum(weights, dim
    =-1)
    cum_values = torch.cumsum(weighted_data
    , dim=-1)
    denominator = cum_weights.clamp_min
    (1.0)
    causal_means = cum_values / denominator

    # For Welford's algorithm, we need to compute the correction term
    # delta using the difference between the current value and the
    # previous running mean.
    shifted_means = torch.zeros_like(causal_means)
    shifted_means[..., 1:] = causal_means[..., :-1]
    delta = data - shifted_means

    # Compute m_2, the second moment accumulator for Welford's
    # algorithm.
    increment = delta * (data - causal_means) * weights
    m_2 = torch.cumsum(increment, dim=-1)

    # Compute the variance using Bessel's correction.
    causal_variance = m_2 / torch.clamp(denominator - 1.0, min=1.0)
    causal_scale = torch.sqrt(causal_variance + minimum_scale)

    return causal_means, causal_scale
```

Listing 1: Vectorized PyTorch implementation of Welford's algorithm for computing causal statistics

In our ablation study (Section D), we find that causal scaling leads to dramatic performance improvements over naive global scaling.

A.2. Attention mechanism

To address the unique challenges of time series data, and particularly to adapt transformer architectures for multivariate time-series forecasting, several works have implemented modifications to the attention mechanism. These strategies have included:

- Concatenating variates along the time dimension and computing full self-attention between every variate/time location, as in the "any-variate attention" used by Woo et al. (2024). This can capture every possible variate and time interaction, but it is costly in terms of computation and memory usage.
- Assuming variate independence, and computing attention only in the time dimension as in Nie et al. (2023); Shi et al. (2025). This is efficient, but throws away all information about variate-wise interactions.
- Computing attention only in the variate dimension, and using a feed-forward network in the time dimension (Ilbert et al., 2024; Liu et al., 2024c).
- Computing "factorized attention," where each transformer block contains a separate variate and time attention computation (Zhang and Yan, 2023; Rao et al., 2021; Arnab et al., 2021). This allows both variate and time mixing, and is more efficient than full cross-attention. However, it doubles the effective depth of the network.

In Section 3.1, we propose a novel approach that allows for both variate and time interactions, while reducing the computational cost and improving overall scalability.

A.2.1. Complexity analysis

After the patchwise embedding layer, we have inputs of shape $\mathbf{X} \in \mathbb{R}^{B \times M \times \frac{L}{P} \times D}$, where B is the batch dimension, M is the number of variates per batch item, $\frac{L}{P}$ is time steps divided by patch width, and D is the model embedding dimension.

Time-wise attention. We parallelize along the time dimension by reshaping the input tensor from 4 dimensions to 3:

$$\mathbf{X} \in \mathbb{R}^{B \times M \times \frac{L}{P} \times D} \rightarrow \mathbf{X}_{\text{time}} \in \mathbb{R}^{(B \times M) \times \frac{L}{P} \times D}$$

This allows for attention to be calculated independently in parallel per variate, giving a complexity of:

$$\mathcal{O}(M \times (\frac{L}{P})^2 \times D)$$

In the time-wise attention blocks, we use causal masking and rotary positional embeddings (Su et al., 2024) with XPOS (Sun et al., 2022) in order to autoregressively model time-dependent features.

Variate-wise attention. We similarly parallelize along the variate dimension by reshaping the input tensor:

$$\mathbf{X} \in \mathbb{R}^{B \times M \times \frac{L}{P} \times D} \rightarrow \mathbf{X}_{\text{variate}} \in \mathbb{R}^{(B \times \frac{L}{P}) \times M \times D}$$

We calculate attention in parallel for each time step, with complexity:

$$\mathcal{O}(\frac{L}{P} \times M^2 \times D)$$

In the variate-wise blocks, we use full bidirectional attention (without causal masking) in order to preserve permutation invariance of the covariates, with a block-diagonal ID mask to ensure that only related variates attend to each other. This masking allows us to pack multiple independent multivariate time series into the same batch, in order to improve training efficiency and reduce the amount of padding.

Computational complexity. Each transformer block in our model contains N time-wise attention layers and 1 variate-wise layer. The complexity for full self-attention over $N + 1$ layers, where interactions can occur across all variates and sequence positions, would be of complexity:

$$\mathcal{O}\left((N+1) \times M^2 \times \left(\frac{L}{P}\right)^2 \times D\right) \tag{1}$$

This reflects the quadratic dependence on both the sequence length $\frac{L}{P}$ and the variate dimension M, with linear dependence on the embedding dimension D. However, by utilizing factorized attention, we can reduce the computational complexity of the attention calculation to:

$$\begin{gathered}\mathcal{O}\left(N \times M \times \left(\frac{L}{P}\right)^2 \times D + \frac{L}{P} \times M^2 \times D\right) = \\ \mathcal{O}\left(D \times \frac{L}{P} \times M \times \left(N \times \frac{L}{P} + M\right)\right)\end{gathered} \tag{2}$$

We demonstrate that factorized variate-wise attention is asymptotically smaller in computational complexity than full self-attention (see Equation 1 and Equation 2). When comparing a model with full self-attention, we can assume N and D are fixed. Therefore:

$$\mathcal{O}\left(M \times \left(\frac{L}{P}\right)^2 + \frac{L}{P} \times M^2\right) < \mathcal{O}\left(M^2 \times \left(\frac{L}{P}\right)^2\right)$$

which reduces to:

$$\mathcal{O}\left(M + \frac{L}{P}\right) < \mathcal{O}\left(M \times \frac{L}{P}\right) .$$

Thus, by factorizing attention into time-wise and variate-wise components, the computational complexity is reduced, especially for large numbers of variates M or long sequences $\frac{L}{P}$, making it more scalable than full self-attention.

A.3. Probabilistic prediction

Practitioners who rely on time series forecasting typically prefer probabilistic predictions. A common practice in neural time series models is to use an output layer where the model regresses the parameters of a probability distribution. This allows for prediction intervals to be computed using Monte Carlo sampling (see Appendix A.4) (Salinas et al., 2020).

Common choices for an output layer are Normal (Salinas et al., 2020) and Student-T (Das et al., 2023; Rasul et al., 2023), which can improve robustness to outliers. Moirai (Woo et al., 2024) allows for more flexible residual distributions by proposing a novel mixture model incorporating a weighted combination of Gaussian, Student-T, Log-Normal, and Negative-Binomial outputs.

However, real-world time series can often have complex distributions that are challenging to fit, with outliers, heavy tails, extreme skew, and multimodality. In order to accommodate these scenarios, we introduce an even more flexible output likelihood in Section 3.1 based on a Student-T mixture model (Peel and McLachlan, 2000).

Toto makes predictions using a mixture of K Student-T distributions (where K is a hyperparameter) for each time step, as well as a learned weighting. Formally, the SMM is defined by:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{T}(x \mid \mu_k, \tau_k, \nu_k) \tag{3}$$

where $\pi_{k \in K}$ are nonnegative mixing coefficients which sum to 1 for the kth Student's t-distribution $\mathcal{T}_k$ with ν_k degrees of freedom, mean μ_k, and scale τ_k. $\mathcal{T}(x \mid \mu, \sigma, \nu)$ is defined as:

$$\begin{gathered}\mathcal{T}(x \mid \mu, \tau, \nu) = \\ \frac{\Gamma\left(\frac{\nu+d}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)(\nu\pi)^{d/2}|\tau|^{1/2}} \left(1 + \frac{1}{\nu}(x-\mu)^\top \tau^{-1}(x-\mu)\right)^{-\frac{\nu+d}{2}}\end{gathered}$$

where $\Gamma(\cdot)$ is the gamma function.

In our ablation study (Appendix D), we find that the SMM improves both point prediction and probabilistic forecasting accuracy when compared with a single Student-T distribution as used in TiDE (Das et al., 2023), Lag-Llama (Rasul et al., 2023), and implementations of DeepAR (Salinas et al., 2020), PatchTST (Nie et al., 2023), iTransformer (Shazeer, 2020), and others in the popular open-source GluonTS library (Alexandrov et al., 2020).

The parameters of this mixture model are computed from the flattened features $h_t \in \mathbb{R}^D$ produced by the transformer backbone for each time step t, where D is the model's embedding dimension. Using a set of linear projections with weight matrices $W \in \mathbb{R}^{K \times D}$ and bias vectors $b \in \mathbb{R}^K$, we derive all K mixture components simultaneously. For each

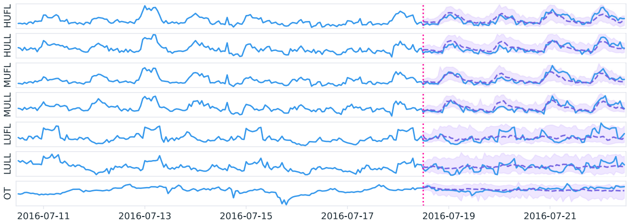


Figure 3: Example of TOTO 's 96-step zero-shot forecasts on the ETTh1 dataset, showing multivariate probabilistic predictions. Solid lines represent ground truth, dashed lines represent median point forecasts, and shaded regions represent 95% prediction intervals.

time step t, the parameters are computed as:

$$\nu_t = 2 + \max(\text{softplus}(W_\nu h_t + b_\nu), \epsilon) \tag{4}$$

$$\mu_t = W_\mu h_t + b_\mu \tag{5}$$

$$\tau_t = \max(\text{softplus}(W_\tau h_t + b_\tau), \epsilon) \tag{6}$$

$$\tilde{\pi}_t = W_\pi h_t + b_\pi \tag{7}$$

where each equation produces a vector in $\mathbb{R}^K$ containing the parameters for all mixture components at time t. The individual component parameters $\nu_{t,k}$, $\mu_{t,k}$, $\tau_{t,k}$, and $\tilde{\pi}_{t,k}$ (the mixture logits) are the kth elements of these vectors. The parameter ϵ is machine epsilon (the smallest positive floating-point number), and softplus(x) = $\log(1 + e^x)$. The use of softplus and ϵ ensure that the scale τ remains positive. Similarly, we add the constraint $\nu > 2$ to ensure that each component of our mixture has well-defined first and second moments (mean and variance).

The mixture weights π are computed using by applying softmax to the logits:

$$\begin{aligned}\pi_{t,k} &= \text{softmax}(\tilde{\pi}_t, k) \\ &= \frac{e^{\tilde{\pi}_{t,k}}}{\sum_{j=1}^{K} e^{\tilde{\pi}_{t,j}}}\end{aligned}$$

An example distribution median and 95th percentile is illustrated in Fig. 3.

A.4. Forecasting

When performing inference, we draw u (for some user specified integer $u > 0$) samples from the mixture distribution at each timestamp, then feed each sample back into the decoder for the next prediction, resulting in n identically and independently sampled time-series. This allows us to produce prediction intervals at any quantile, limited only by the number of samples. Our exact sampling procedure for several tasks is detailed in Section 4.

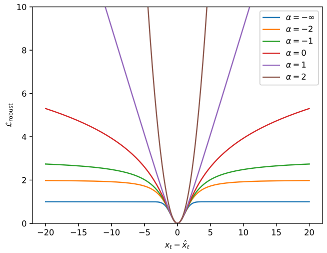


Figure 4: Visualization of generalized robust loss for different values of α, with δ fixed at 1. Changing δ scales the horizontal axis.

A.5. Loss function

TOTO learns the conditional distribution $p(X_{i+1}|X_{1:i})$, where X_i represents the i-th patch containing multiple time steps.

The $\mathcal{L}_{\text{NLL}}$ optimizes probabilistic predictions and is defined as:

$$\begin{aligned}&\mathcal{L}_{\text{NLL}}(x, \mu, \tau, \nu) = -\log\left(p(x_t|X_{1:i})\right) = \\ &-\log\left(\sum_{k=1}^{K} \pi_{t,k} \mathcal{T}(x_t \mid \mu_{t,k}, \tau_{t,k}, \nu_{t,k})\right)\end{aligned} \tag{8}$$

where $p(x_t|X_{1:i})$ is the probability density of the ground truth x_t under the model's predicted mixture distribution conditioned on all previous patches. The parameters $\pi_{t,k}$, $\mu_{t,k}$, $\tau_{t,k}$, and $\nu_{t,k}$ are the mixture weights and Student-T parameters computed by the model for time step t.

For a ground truth value x_t in patch $i + 1$ and the mean prediction $\hat{x}_t = \mathbb{E}[p(x_t|X_{1:i})]$, the robust loss is defined in Barron (2017).

Here, $\mathcal{L}_{\text{Robust}(\alpha,\delta)}$ serves as a point prediction error measure, where $\alpha \leq 2$ is a shape parameter that controls the robustness to outlier observations (Fig. 4) and $\delta > 0$ is a scale parameter that determines the size of the parabolic portion of the loss curve. This loss component directly penalizes point prediction accuracy, and we conjecture this may help steer the mixture model away from degenerate solutions of the type described in (Eisner). In our ablation study, we find that adding the robust loss component significantly improves point forecasting accuracy without hurting probabilistic predictions (Section D).

$\mathcal{L}$ is applied to each timestep t in the target patch X_{i+1}, and the total loss is aggregated across all timesteps during training. By combining the probabilistic $\mathcal{L}_{NLL}$ loss with the robust point-prediction loss, we achieve both accurate distribution modeling and stable convergence, especially in domains with highly heterogeneous data characteristics. The hyperparameter λ_{NLL} controls the balance between these two loss components and is tuned empirically.

A.6. Hyperparameter Optimization

To determine the optimal architecture and training configuration for Toto, we conducted an extensive hyperparameter sweep using Optuna (Akiba et al., 2019), a Bayesian optimization framework. We employed the Tree-structured Parzen Estimator (TPE) algorithm with 65 trials to efficiently explore the high-dimensional search space.

Our optimization objective was to minimize the validation mean absolute error (MAE) on multi-step forecasting tasks on a random validation split of the observability portion of the pretraining data. We train the model using the AdamW optimizer (Loshchilov and Hutter, 2019) with a WSD learning rate scheduler (Hu et al., 2024). We performed this sweep over 50,000 steps over 133 iterations over ranges described in Table 3.

The resulting hyperparameter configuration described in Table 4 obtained the best multistep (average of 96 and 192) MAE on the observability validation set.

In Section D, we perform an ablation study on the impact of various model components. We optimize speed and memory usage by utilizing fused kernel implementations and memory efficient attention operations via xformers (Lefaudeux et al., 2022), (with the FlashAttention-3 kernel (Shah et al., 2024)).

We ran all experiments, including hyperparameter tuning, final model training, and benchmark evaluation on a GPU cluster consisting of A100s and H100s.

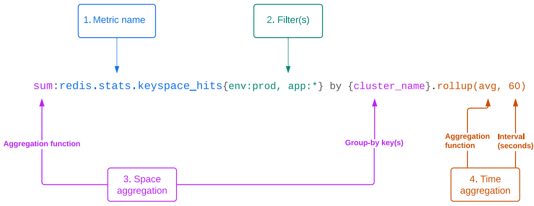
B. Training data preprocessing

B.1. Observability Dataset

Observability metrics are retrieved from a large-scale time series database using a specialized query language supporting filters, group-bys, time aggregation, and various transformations and postprocessing functions (Fig. 5). We consider groups returned from the same query to be related variates in a multivariate time series. After we retrieve the query results, we discard the query strings and group identifiers, keeping only the raw numeric data. As described in Section 3.2, we source metrics defined by user-generated queries. This excludes any customer data and is sourced solely from the internal users and telemetry.

B.2. Public Datasets

We train on a public dataset corpus, which exposes the model to diverse time series behaviors across different domains and sampling frequencies, contributing approximately 250 billion time series points to our training data.



```
sum:redis.stats.keyspace_hits{env:prod, app:*} by {cluster_name}.rollup(avg, 60)
```

Figure 5: Example metric query in the observability platform. The metric name (1) determines which metric is being queried. The filter clause (2) limits which contexts are queried, in this case restricting the query to apps in the prod environment. The space aggregation (3) indicates that the sum of the metric value should be returned for each unique value of the group-by key(s), aggregated across all other keys. The time aggregation (4) indicates that metric values should be aggregated to the average for each 60-second interval. The query results will be a multivariate time series with 1-minute time steps, and with separate individual variates for each unique value of `cluster_name`.

Our pre-training dataset incorporates a diverse collection of time series from the GIFT-Eval Pretrain collection (Godahewa et al., 2021) and non-overlapping Chronos datasets (Ansari et al., 2024). These datasets include `ercot`, `exchange_rate`, `weatherbench_daily`, `weatherbench_hourly`, `weatherbench_monthly`, `dominick`, `mexico_city_bikes`, `ushcn_daily`, and `wiki_daily_100k`.

Handling this vast amount of data requires several preprocessing steps to ensure consistency and quality. We describe the details of preprocessing and data augmentation in Appendix B.4.

B.3. Synthetic Data

We supplement our training with synthetic data to further improve model performance. Our synthetic dataset consists of procedurally generated time series using an approach similar to TimesFM (Das et al., 2024), as well as `kernel_synth_1m` from the Chronos dataset (Ansari et al., 2024). Synthetic data constitutes approximately 33% of our training dataset.

We generate synthetic time series through the composition of components such as piecewise linear trends, ARMA processes, sinusoidal seasonal patterns, and various residual distributions. Our procedural generation randomly combines multiple processes per variate to introduce diverse patterns. The generation includes creating base series with transformations, clipping extreme values, and rescaling to specified ranges.

These synthetic datasets help the model learn robust representations by providing examples with specific characteristics that might be underrepresented in real-world data.

Category	Values / Ranges
Patch Size	$\{16, 32, 64\}$
Variate-wise Attention Frequency	Every $\{3, 4, 6, 12\}$ layers
Variate-wise Layer First	[True, False]
$\mathcal{T}$ Components	[8, 16, 24, 32]
Loss Function	$\lambda_{\text{NLL}} \in [0.05, 1.0]$
Robust Loss Params	$\alpha \in \{-\infty, -2, 0, 0.5, 1.0\}$, $\delta \in [0.1, 3.0]$
Warmup Steps	[0, 10,000]
Stable Ratio*	[.1, .9]
Learning Rate	$[10^{-5}, 5 \times 10^{-3}]$
Weight Decay	$[10^{-3}, 10^{-1}]$
Synthetic Data Proportion	$[0.0, 0.75]$
Shuffling Type	[Normally Distributed, Adjacent, Random, None]
Normally Distributed Shuffling Standard Deviation	[.15, 5000]
Shuffling Frequency	$[0.0, 0.3]$

Table 3: Summary of hyperparameter search space. *Stable Ratio defines the proportion of steps that are stable after the warmup phase of the WSD learning rate schedule.

Hyperparameter	Value
Embedding Dimension	768
MLP Dimension	3072
# Layers	12
# Heads	12
# Variates	32
Spacewise Layer Cadence	12
Patch Size	64
# $\mathcal{T}$ Mixture Model Components	24
Annealing Schedule	WSD
Optimizer	AdamW
$(\beta_1, \ \beta_2)$	(0.9579, 0.9581)
Weight Decay	0.0014
Initial Learning Rate	0.0005
Warmup Steps	6784
Stable Steps	112,255
Decay Steps	15,962
Batch Size	128
Total Train Steps	135,001
$\mathcal{L}_{\text{Robust}}$ α	0.0000
$\mathcal{L}_{\text{Robust}}$ δ	0.1010
λ_{NLL}	0.5755
κ	10

Table 4: Hyperparameters for Toto

B.4. Preprocessing

To prepare the raw time series for training, we apply padding and masking techniques to align the series lengths, making them divisible by the patch stride. This involves adding necessary left-padding to both the time series data and the ID mask, ensuring compatibility with the model's requirements.

Next, various data augmentations are employed to enhance the dataset's robustness. We introduce random time offsets to prevent memorization caused by having series always align the same way with the patch grid. After concatenating the observability and public datasets for training, we also implement a variate shuffling strategy to maintain diversity and representation. Specifically, we randomly combine variates from either observability, open source datasets (GIFT-Eval pretrain and Chronos datasets), and/or synthetic data with a probability of 14%, thus creating new, diverse combinations of data points. We shuffle series with adjacent indices (batched by 32 variates), favoring data points that were closer together in the original datasets. This approach improves the model's ability to generalize across different types of data effectively.

C. Results

C.1. Inference procedures

To evaluate the comparison models on BOOM, we closely follow the evaluation methodology used in the GIFT-Eval implementation. For models not included in GIFT-Eval, we rely on their official implementations and recommended evaluation procedures. All foundation models are evaluated using a unified context length of 2048. This choice is informed by preliminary experiments showing that a shorter context length (512) leads to a general degradation in performance across models. Therefore, we opt for a relatively large context window (2048) to preserve forecast quality, while ensuring feasibility on available hardware.

To evaluate the zero-shot performance of other foundation models on BOOM, we follow the sampling procedures outlined in their respective manuscripts. For Chronos, we generate 20 samples and use the median prediction as the point forecast. For Moirai, we generate 100 samples, again taking the median, and set the patch size to "auto". For TOTO we generate 256 samples and take the median as the point forecast. TimesFM produces only point forecasts of the mean, which we use directly. In all cases, we compute CRPS with respect to the probabilistic samples and MASE with respect to the point forecast. Since TimesFM and Chronos support only univariate forecasting, we evaluate each variate independently. In contrast, both Moirai and TOTO support joint prediction over groups of related variates.

For the three statistical baselines—AutoARIMA, AutoTheta, and AutoETS—we use the default hyperparameter settings from the statsforecast package, with one exception: for AutoARIMA, we reduce max_d and max_D from 2 to 1 due to frequent numerical instability when d = D = 2. Following GIFT-Eval, we set the maximum input length for all statistical models to 1000.

C.2. BOOM

In Fig. 6, we present qualitative comparisons across three representative forecasting scenarios to highlight the behavioral differences between TOTO, Chronos, and Moirai. In the first example (a), features a highly stochastic signal interwoven with complex seasonal components. While Moirai and Chronos models tend to overfit short-term fluctuations—resulting in jagged forecasts and unstable confidence intervals— TOTO effectively identifies and extrapolates the latent seasonal structure, yielding smoother, more coherent trajectories and uncertainty bands that reflect a deeper structural understanding of the series dynamics. Example (b) the target signal exhibits high dynamism with rapidly oscillating structure and sustained amplitude modulations—posing a challenge for long-range temporal modeling. While both Moirai and Chronos models progressively lose phase alignment and dampen their amplitude estimates, TOTO consistently maintains sharp, temporally aligned forecasts with well-calibrated uncertainty, accurately tracking the intricate periodic structure far into the forecast horizon. Finally, example (c), the target series is characterized by sparse, bursty impulses with high variance across events. Here, although TOTO 's mean prediction does not always precisely capture individual peaks, its predictive distribution faithfully mirrors the underlying spikiness of the series, in stark contrast to Chronos, which collapses to an overconfident flat trajectory.

Table 5 reports the results for all versions and sizes of the zero-shot models.

To better understand the capabilities and limitations of different forecasting models, we conduct a disaggregated evaluation across four major characteristics that describe time series in the BOOM dataset. This analysis enables us to probe how models respond to structural diversity in real-world time series data.

Across all three categorical axes, the TOTO consistently achieves the lowest CRPS, with strong margins over all baselines.

C.3. LSF

In addition to our primary evaluations, we also assess the model's performance on the Long Sequence Forecasting (LSF) benchmark datasets—ETTh1, ETTh2, ETTm1, ETTm2, Electricity, and Weather (Wu et al., 2021). As noted by Aksu et al. (2024), these datasets are limited in size and diversity, and recent findings (Xu et al., 2024) suggest that strong supervised baselines can already perform near the upper bound on such benchmarks. This may indicate a saturation point where further gains from foundation models are difficult to observe, rather than a fundamental limitation of the models themselves. Nevertheless, as it remains a widely used legacy benchmark in the literature, we report zero-shot results of TOTO on it to maintain consistency with established practices in the field.

Furthermore we leverage its small scale and constrained use-cases to examine TOTO's capacity to transfer to new datasets and specialized domains by conducting fine-tuning experiments on the training splits of its datasets.

Following standard practice for the LSF benchmark, we report normalized Mean Absolute Error (MAE) and Mean Squared Error (MSE), in order to be able to compare performance across different datasets. We evaluate using forecast lengths of 96, 192, 336, and 720 time steps. Predictions are generated using sliding windows with a stride of 1. For the Electricity dataset, however, we use a stride equal to the prediction length to reduce computational resource requirements. The results are then averaged. We compare TOTO's performance with results reported by recent state-of-the-art time series foundation models, including Moirai (Woo et al., 2024), VisionTS (Chen et al., 2024), TimesFM (Das et al., 2024), Time-MoE (Shi et al., 2025), TimeLLM (Jin et al., 2024), GPT4TS (Zhou et al., 2023), xLSTM-Time (Alharthi and Mahmood, 2024) and other models evaluated in Woo et al. (2024) and Das et al. (2024). We display zero-shot and full-shot TOTO results in Table 9 and Table 10 respectively. We also provide additional per prediction length results in Table 11 and Table 12.

Table 9 shows that TOTO consistently delivers the best overall performance across all datasets, achieving the lowest average MAE and MSE, and outperforming other zero-shot baselines on 8 out of 12 evaluation metrics. Its per-

Dataset	Metric	Toto	$\text{Moirai}_{\text{Small}}$	$\text{Moirai}_{\text{Base}}$	$\text{Moirai}_{\text{Large}}$	$\text{TimesFM}_{2.0}$	$\text{Chronos-bolt}_{\text{Small}}$	$\text{Chronos-bolt}_{\text{Base}}$	Timer	Time-MoE	VisionTS	Naive
BOOM	MASE ↓	**0.617**	0.738	0.710	0.720	0.725	0.733	0.726	0.796	0.806	0.991	1.000
	CRPS ↓	**0.375**	0.447	0.428	0.436	0.447	0.455	0.451	0.639	0.649	0.675	1.000

Table 5: **BOOM results.** Full results across all models evaluated from Table 1. Key: **Best results**, Second-best results.

Real Term	Metric	Toto	$\text{Moirai}_{\text{Small}}$	$\text{Moirai}_{\text{Base}}$	$\text{Moirai}_{\text{Large}}$	$\text{TimesFM}_{2.0}$	$\text{Chronos-bolt}_{\text{Small}}$	$\text{Chronos-bolt}_{\text{Base}}$	Timer	Time-MoE	VisionTS	Naive
Long	MASE ↓	**0.688**	0.795	0.780	0.799	0.817	0.813	0.798	0.809	0.886	1.026	1.000
	CRPS ↓	**0.424**	0.482	0.473	0.491	0.522	0.528	0.519	0.661	0.724	0.698	1.000
Medium	MASE ↓	**0.657**	0.771	0.753	0.770	0.780	0.782	0.782	0.804	0.866	1.011	1.000
	CRPS ↓	**0.406**	0.476	0.460	0.475	0.499	0.508	0.507	0.671	0.725	0.698	1.000
Short	MASE ↓	**0.535**	0.670	0.627	0.626	0.619	0.638	0.632	0.779	0.704	0.947	1.000
	CRPS ↓	**0.318**	0.399	0.370	0.369	0.359	0.368	0.365	0.597	0.541	0.640	1.000

Table 6: Performance comparison of TOTO and other zero-shot models across different **prediction terms**. MASE and CRPS are normalized by the Seasonal Naive forecast and aggregated across tasks using the shifted geometric mean. Key: **Best results**, Second-best results.

Type	Metric	Toto	$\text{Moirai}_{\text{Small}}$	$\text{Moirai}_{\text{Base}}$	$\text{Moirai}_{\text{Large}}$	$\text{TimesFM}_{2.0}$	$\text{Chronos-Bolt}_{\text{Small}}$	$\text{Chronos-Bolt}_{\text{Base}}$	Timer	Time-MoE	VisionTS	Naive
Count	MASE ↓	0.687	0.814	0.795	0.813	0.919	0.883	0.880	0.663	**0.652**	1.220	1.000
	CRPS ↓	**0.317**	0.370	0.353	0.372	0.403	0.403	0.402	0.662	0.651	0.603	1.000
Distribution	MASE ↓	**0.658**	0.741	0.724	0.729	0.745	0.759	0.753	0.890	0.878	1.034	1.000
	CRPS ↓	**0.382**	0.434	0.422	0.428	0.440	0.452	0.446	0.608	0.604	0.674	1.000
Gauge	MASE ↓	**0.583**	0.720	0.686	0.700	0.706	0.706	0.696	0.721	0.760	0.922	1.000
	CRPS ↓	**0.382**	0.471	0.444	0.456	0.466	0.469	0.463	0.658	0.694	0.672	1.000
Rate	MASE ↓	**0.634**	0.753	0.728	0.733	0.726	0.742	0.739	0.864	0.846	1.041	1.000
	CRPS ↓	**0.369**	0.433	0.418	0.422	0.431	0.445	0.443	0.630	0.619	0.687	1.000

Table 7: Performance comparison of TOTO and other zero-shot models across different **metric types**. MASE and CRPS are normalized by the Seasonal Naive forecast and aggregated across tasks using the shifted geometric mean. Key: **Best results**, Second-best results.

Domain	Metric	Toto	$\text{Moirai}_{\text{Small}}$	$\text{Moirai}_{\text{Base}}$	$\text{Moirai}_{\text{Large}}$	$\text{TimesFM}_{2.0}$	$\text{Chronos-Bolt}_{\text{Small}}$	$\text{Chronos-Bolt}_{\text{Base}}$	Timer	Time-MoE	VisionTS	Naive
Application usage	MASE ↓	**0.639**	0.747	0.721	0.730	0.736	0.748	0.748	0.871	0.863	1.042	1.000
	CRPS ↓	**0.378**	0.440	0.422	0.430	0.441	0.452	0.451	0.636	0.633	0.691	1.000
Database	MASE ↓	**0.635**	0.751	0.738	0.743	0.765	0.761	0.757	0.716	0.714	1.017	1.000
	CRPS ↓	**0.362**	0.429	0.414	0.418	0.440	0.444	0.441	0.619	0.618	0.647	1.000
Infrastructure	MASE ↓	**0.568**	0.692	0.650	0.670	0.679	0.678	0.663	0.728	0.791	0.863	1.000
	CRPS ↓	**0.391**	0.476	0.446	0.462	0.471	0.474	0.466	0.655	0.713	0.666	1.000
Networking	MASE ↓	**0.635**	0.795	0.786	0.773	0.765	0.779	0.757	0.871	0.856	1.035	1.000
	CRPS ↓	**0.400**	0.493	0.484	0.484	0.493	0.506	0.489	0.725	0.721	0.734	1.000
Security	MASE ↓	**0.682**	0.741	0.739	0.736	0.717	0.734	0.729	0.828	0.770	0.924	1.000
	CRPS ↓	**0.476**	0.505	0.504	0.504	0.525	0.539	0.535	0.664	0.625	0.735	1.000

Table 8: Performance comparison of TOTO and other zero-shot models across different **metric domains**. MASE and CRPS are normalized by the Seasonal Naive forecast and aggregated across tasks using the shifted geometric mean. Key: **Best results**, Second-best results.

formance is especially strong on ETTm2, Electricity, and Weather, where it continues to excel even in zero-shot scenarios.

Furthermore, Table 10 shows that even when starting from a strong SOTA baseline, TOTO's performance improves with fine-tuning, showing it can achieve full-shot SOTA results and adapt to new domains with limited data. This highlights TOTO's robustness and versatility as a foundation model for a wide range of time-series forecasting tasks.

Full-shot results on LSF benchmarks We conduct fine-tuning experiments on Toto following similar procedure delineated by (Wu et al., 2023) and (Woo et al., 2024). The full-shot results for each dataset, comparing fine-tuned and zero-shot performance, are reported in Table 10.

Results Our experimental results demonstrate that when finetuned, denoted as TOTO_{FT}), achieves state-of-the-art performance on 3 out of 6 datasets in the LSF benchmark—specifically, ETTm2, Electricity, and Weather—where it outperforms all other models on both MAE and MSE metrics. Additionally, TOTO_{FT} achieves the best MAE score on ETTm1 and ETTh2, although it does not lead on MSE for those datasets. Compared to its zero-shot counterpart, TOTO_{FT} consistently improves both MAE and MSE metrics across most datasets, with particularly notable gains in ETTm1 (MAE: 0.378 → 0.357, MSE: 0.396 → 0.349) and ETTm2 (MAE: 0.303 → 0.291,

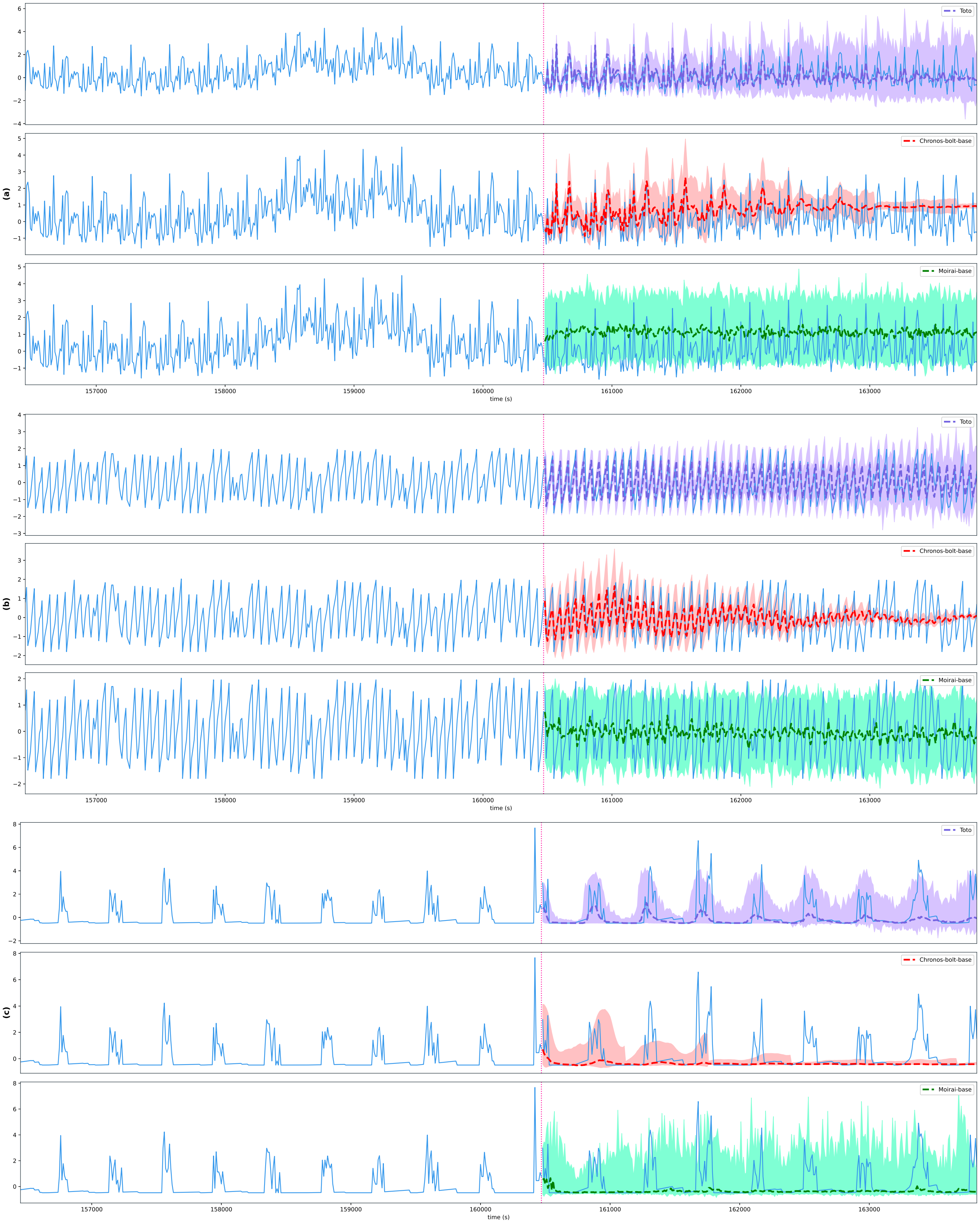


Figure 6: Example of 336-step zero-shot comparative forecasts on the Boom, showing multivariate probabilistic predictions. Solid lines represent ground truth, dashed lines represent median point forecasts, and shaded regions represent 95% prediction intervals.

Dataset	Metric	Toto	Moirai$_{\text{Small}}$	Moirai$_{\text{Base}}$	Moirai$_{\text{Large}}$	Time-MoE$_{\text{Base}}$	Time-MoE$_{\text{Large}}$	Time-MoE$_{\text{Ultra}}$	VisionTS
ETTh1	**MAE ↓**	**0.413**	0.424	0.438	0.469	0.424	0.419	0.426	0.414
	MSE ↓	0.435	0.400	0.434	0.510	0.400	0.394	0.412	**0.390**
ETTh2	**MAE ↓**	**0.363**	0.379	0.382	0.376	0.404	0.415	0.399	0.375
	MSE ↓	0.340	0.341	0.345	0.354	0.366	0.405	0.371	**0.333**
ETTm1	**MAE ↓**	0.378	0.409	0.388	0.389	0.415	0.405	0.391	**0.372**
	MSE ↓	0.396	0.448	0.381	0.390	0.394	0.376	**0.356**	0.374
ETTm2	**MAE ↓**	**0.303**	0.341	0.321	0.320	0.365	0.361	0.344	0.321
	MSE ↓	**0.267**	0.300	0.272	0.276	0.317	0.316	0.288	0.282
Electricity	**MAE ↓**	**0.242**†	0.320	0.274	0.273	-	-	-	0.294
	MSE ↓	**0.158**†	0.233	0.188	0.188	-	-	-	0.207
Weather	**MAE ↓**	**0.245**	0.267	0.261	0.275	0.297	0.300	0.288	0.292
	MSE ↓	**0.224**	0.242	0.238	0.259	0.265	0.270	0.256	0.269
Mean	**MAE ↓**	**0.324**	0.357	0.344	0.350	-	-	-	0.345
	MSE ↓	**0.303**	0.327	0.310	0.330	-	-	-	0.309
Best Count		**8**	**0**	**0**	**0**	**0**	**0**	**1**	**3**

Table 9: **LSF results** Zero-Shot comparison of models on the LSF benchmark. Non-TOTO values are reproduced from published tables. Key: **Best results**, Second-best results. Values marked with † use a window stride equal to the prediction length on the Electricity dataset. "Best Count" row reports the number of times each model attains the best result for a given dataset-metric pair.

		Zero Shot	*Full Shot*																	
Dataset	**Metric**	**Toto**	**Toto$_{\text{FT}}$**	**TimeLLM**	**GPT4TS**	**VisionTS$_{\text{FT}}$**	**Time-MoE$_{\text{BaseFT}}$**	**Time-MoE$_{\text{LargeFT}}$**	**Time-MoE$_{\text{UltraFT}}$**	**TimesFM***	**xLSTMTime**	**iTransformer**	**TimesNet**	**PatchTST**	**Crossformer**	**TiDE**	**DLinear**	**SCINet**	**FEDformer**
ETTh1	**MAE ↓**	0.413	0.409	0.423	0.426	0.409	0.406	**0.404**	0.406	0.426	0.428	0.448	0.450	0.455	0.522	0.507	0.452	0.647	0.460
	MSE ↓	0.435	0.415	0.408	0.427	0.395	0.379	0.375	**0.373**	-	0.408	0.454	0.458	0.469	0.529	0.541	0.456	0.747	0.440
ETTh2	**MAE ↓**	0.363	**0.363**	0.383	0.394	0.382	0.386	0.386	0.380	0.410	0.386	0.407	0.497	0.407	0.684	0.550	0.515	0.723	0.449
	MSE ↓	0.340	0.339	**0.334**	0.354	0.336	0.346	0.361	**0.334**	-	0.346	0.383	0.414	0.387	0.942	0.611	0.559	0.954	0.437
ETTm1	**MAE ↓**	0.378	**0.357**	0.372	0.383	0.367	0.381	0.371	0.373	0.388	0.371	0.410	0.406	0.400	0.495	0.419	0.407	0.481	0.452
	MSE ↓	0.396	0.349	0.329	0.352	0.338	0.345	**0.322**	0.329	-	0.347	0.407	0.400	0.387	0.513	0.419	0.403	0.486	0.448
ETTm2	**MAE ↓**	0.303	**0.291**	0.313	0.326	0.319	0.335	0.332	0.334	0.334	0.310	0.332	0.333	0.326	0.611	0.404	0.401	0.537	0.349
	MSE ↓	0.267	**0.244**	0.251	0.266	0.261	0.271	0.284	0.277	-	0.254	0.288	0.291	0.281	0.757	0.358	0.350	0.571	0.305
Electricity	**MAE ↓**	0.242†	**0.233**†	0.252	0.263	0.249	-	-	-	-	0.250	0.270	0.295	0.304	0.334	0.344	0.300	0.365	0.327
	MSE ↓	0.158†	**0.150**†	0.158	0.167	0.156	-	-	-	-	0.157	0.178	0.193	0.216	0.244	0.252	0.212	0.268	0.214
Weather	**MAE ↓**	0.245	**0.233**	0.257	0.270	0.262	0.275	0.273	0.280	-	0.255	0.278	0.287	0.281	0.315	0.320	0.317	0.363	0.360
	MSE ↓	0.224	**0.206**	0.225	0.237	0.227	0.236	0.234	0.250	-	0.222	0.258	0.259	0.259	0.259	0.271	0.265	0.292	0.309
Mean	**MAE ↓**	0.324	**0.314**	0.333	0.344	0.331	-	-	-	-	0.333	0.358	0.378	0.362	0.494	0.424	0.399	0.519	0.400
	MSE ↓	0.303	**0.284**	**0.284**	0.300	0.286	-	-	-	-	0.289	0.328	0.336	0.333	0.541	0.409	0.374	0.553	0.359
Best Count			**8**	**1**	**0**	**0**	**0**	**2**	**2**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table 10: Full-Shot comparison of models on the LSF benchmark, with TOTO's Zero-Shot result in the first data column.*TimesFM only reports values for MAE on ETTh1, ETTh2, ETTm1, and ETTm2 after fine-tuning.
Key: **Best results**, Second-best results. Values marked with † use a window stride equal to the prediction length on the Electricity dataset. "Best Count" row reports the number of times each model attains the best result for a given dataset-metric pair.

MSE: 0.267 → 0.244). Overall, TOTO$_{\text{FT}}$ ranks first in 8 out of 12 metric-dataset pairs, outperforming all other models, including both zero-shot and full-shot baselines. Notably, it also delivers the best overall performance on the benchmark, achieving the lowest average MAE (0.314) and MSE (0.284). These results underscore the effectiveness of fine-tuning in enhancing Toto's predictive performance, establishing TOTO$_{\text{FT}}$ as the new SOTA model on the LSF benchmark. In addition, this demonstrates that Toto is a robust foundation model, adaptable to a wide range of downstream datasets, including those from entirely new domains, making it a versatile choice for time-series forecasting tasks.

A closer examination of the results reveals that while Toto$_{\text{FT}}$ achieves state-of-the-art performance on most datasets, the effectiveness of fine-tuning varies across them. Fine-tuning proves especially beneficial on ETTm1, ETTm2, and Weather, where it significantly enhances model predictions. In contrast, the improvements on ETTh1 are more modest, and for ETTh2, fine-tuning yields no notable gains—potentially due to the relatively small size of these datasets. Moreover, even though fine-tuning generally improves performance over the original TOTO model, TOTO$_{\text{FT}}$ does not outperform other full-shot models on ETTh1.

Additional details on zero-shot and full-shot results per prediction length are displayed in Table 12

D. Ablations

We evaluate the contribution of various architectural components of the TOTO model by systematically disabling one component at a time and measuring the relative performance degradation. The full Toto model serves as the

			Zero Shot							
Dataset	**Prediction Length**	**Metric**	**Toto**	**$\text{Moirai}_{\text{Small}}$**	**$\text{Moirai}_{\text{Base}}$**	**$\text{Moirai}_{\text{Large}}$**	**$\text{Time-MoE}_{\text{Base}}$**	**$\text{Time-MoE}_{\text{Large}}$**	**$\text{Time-MoE}_{\text{Ultra}}$**	**VisionTS**
ETTh1	**96**	**MAE ↓**	0.381	0.402	0.402	0.398	0.381	0.382	**0.379**	0.383
		MSE ↓	0.382	0.375	0.384	0.380	0.357	0.350	**0.349**	0.353
	192	**MAE ↓**	0.408	0.419	0.429	0.434	**0.404**	0.412	0.413	0.410
		MSE ↓	0.428	0.399	0.425	0.440	**0.384**	0.388	0.395	0.392
	336	**MAE ↓**	**0.422**	0.429	0.450	0.474	0.434	0.430	0.453	0.423
		MSE ↓	0.457	0.412	0.456	0.514	0.411	0.411	0.447	**0.407**
	720	**MAE ↓**	**0.440**	0.444	0.473	0.568	0.477	0.455	0.462	0.441
		MSE ↓	0.472	0.413	0.470	0.705	0.449	0.427	0.457	**0.406**
ETTh2	**96**	**MAE ↓**	**0.310**	0.334	0.327	0.325	0.359	0.354	0.352	0.328
		MSE ↓	0.273	0.281	0.277	0.287	0.305	0.302	0.292	**0.271**
	192	**MAE ↓**	**0.356**	0.373	0.374	0.367	0.386	0.385	0.379	0.367
		MSE ↓	0.339	0.340	0.340	0.347	0.351	0.364	0.347	**0.328**
	336	**MAE ↓**	0.387	0.393	0.401	0.393	0.418	0.425	0.419	**0.381**
		MSE ↓	0.374	0.362	0.371	0.377	0.391	0.417	0.406	**0.345**
	720	**MAE ↓**	**0.400**	0.416	0.426	0.421	0.454	0.496	0.447	0.422
		MSE ↓	**0.375**	0.380	0.394	0.404	0.419	0.537	0.439	0.388
ETTm1	**96**	**MAE ↓**	**0.333**	0.383	0.360	0.363	0.368	0.357	0.341	0.347
		MSE ↓	0.320	0.404	0.335	0.353	0.338	0.309	**0.281**	0.341
	192	**MAE ↓**	0.364	0.402	0.379	0.380	0.388	0.381	**0.358**	0.360
		MSE ↓	0.371	0.435	0.366	0.376	0.353	0.346	**0.305**	0.360
	336	**MAE ↓**	0.388	0.416	0.394	0.395	0.413	0.408	0.395	**0.374**
		MSE ↓	0.408	0.462	0.391	0.399	0.381	0.373	**0.369**	0.377
	720	**MAE ↓**	0.426	0.437	0.419	0.417	0.493	0.477	0.472	**0.405**
		MSE ↓	0.485	0.490	0.434	0.432	0.504	0.475	0.469	**0.416**
ETTm2	**96**	**MAE ↓**	**0.237**	0.282	0.269	0.260	0.291	0.286	0.288	0.282
		MSE ↓	**0.172**	0.205	0.195	0.189	0.201	0.197	0.198	0.228
	192	**MAE ↓**	**0.280**	0.318	0.303	0.300	0.334	0.322	0.312	0.305
		MSE ↓	**0.232**	0.261	0.247	0.247	0.258	0.250	0.235	0.262
	336	**MAE ↓**	**0.320**	0.355	0.333	0.334	0.373	0.375	0.348	0.328
		MSE ↓	**0.290**	0.319	0.291	0.295	0.324	0.337	0.293	0.293
	720	**MAE ↓**	0.375	0.410	0.377	0.386	0.464	0.461	0.428	**0.370**
		MSE ↓	0.372	0.415	0.355	0.372	0.488	0.480	0.427	**0.343**
Electricity	**96**	**MAE ↓**	**0.211**†	0.299	0.248	0.242	-	-	-	0.266
		MSE ↓	**0.125**†	0.205	0.158	0.152	-	-	-	0.177
	192	**MAE ↓**	**0.228**†	0.310	0.263	0.259	-	-	-	0.277
		MSE ↓	**0.145**†	0.220	0.174	0.171	-	-	-	0.188
	336	**MAE ↓**	**0.244**†	0.323	0.278	0.278	-	-	-	0.296
		MSE ↓	**0.157**†	0.236	0.191	0.192	-	-	-	0.207
	720	**MAE ↓**	**0.284**†	0.347	0.307	0.313	-	-	-	0.337
		MSE ↓	**0.207**†	0.270	0.229	0.236	-	-	-	0.256
Weather	**96**	**MAE ↓**	**0.179**	0.212	0.203	0.208	0.214	0.213	0.211	0.257
		MSE ↓	**0.149**	0.173	0.167	0.177	0.160	0.159	0.157	0.220
	192	**MAE ↓**	**0.223**	0.250	0.241	0.249	0.260	0.266	0.256	0.275
		MSE ↓	**0.192**	0.216	0.209	0.219	0.210	0.215	0.208	0.244
	336	**MAE ↓**	**0.265**	0.282	0.276	0.292	0.309	0.322	0.290	0.299
		MSE ↓	**0.245**	0.260	0.256	0.277	0.274	0.291	0.255	0.280
	720	**MAE ↓**	**0.312**	0.322	0.323	0.350	0.405	0.400	0.397	0.337
		MSE ↓	**0.310**	0.320	0.321	0.365	0.418	0.415	0.405	0.330
Best Count			**29**	**0**	**0**	**0**	**2**	**0**	**6**	**11**

Table 11: Zero-Shot-Shot Comparison of different models with TOTO on the LSF benchmark datasets for each prediction length. Non-TOTO values are reproduced from published tables.
Key: **Best results**, Second-best results. Values marked with † use a window stride equal to the prediction length on the Electricity dataset. "Best Count" row reports the number of times each model attains the best result for a given metric.

control, and each variant's performance is presented relative to this baseline in Table 13. All models in the ablation study, including the control, were trained for 75,000 steps (a subset of the full-length training of the TOTO base model).

To compare performance between the different arms of the experiment, we look at NLL loss on a held-out validation split of the observability portion of the pretraining data. This summarizes the output distribution and gives us a single performance metric to compare both point forecasting and probabilistic forecasting. For each model, we pick the checkpoint with lowest NLL throughout the training run (evaluating on the validation set every 5,000 steps).

The results reveal that removing key modeling elements significantly impacts performance. Disabling Causal Scaling leads to the largest degradation, with an increase of

			Zero Shot	*Full Shot*																
Dataset	**Prediction Length**	**Metric**	**Toto**	**Toto$_{\text{FT}}$**	**TimeLLM**	**GPT4TS**	**VisionTS$_{\text{FT}}$**	**Time-MoE$_{\text{BaseFT}}$**	**Time-MoE$_{\text{LargeFT}}$**	**Time-MoE$_{\text{UltraFT}}$**	**TimesFM***	**xLSTMTime**	**iTransformer**	**TimesNet**	**PatchTST**	**Crossformer**	**TiDE**	**DLinear**	**SCINet**	**FEDformer**
ETTh1	**96**	**MAE ↓**	0.381	0.374	0.392	0.397	0.376	0.373	0.371	**0.365**	0.398	0.395	0.405	0.402	0.419	0.448	0.464	0.400	0.599	0.419
		MSE ↓	0.382	0.364	0.362	0.376	0.347	0.345	0.335	**0.323**	-	0.368	0.386	0.384	0.414	0.423	0.479	0.386	0.654	0.376
	192	**MAE ↓**	0.408	0.402	0.418	0.418	0.400	0.396	0.400	**0.391**	0.424	0.416	0.436	0.429	0.445	0.474	0.492	0.432	0.631	0.448
		MSE ↓	0.428	0.409	0.398	0.416	0.385	0.372	0.374	**0.359**	-	0.401	0.441	0.436	0.460	0.471	0.525	0.437	0.719	0.420
	336	**MAE ↓**	0.422	0.418	0.427	0.433	0.415	**0.412**	**0.412**	0.418	0.436	0.437	0.458	0.469	0.466	0.546	0.515	0.459	0.659	0.465
		MSE ↓	0.457	0.436	0.430	0.442	0.407	0.389	0.390	**0.388**	-	0.422	0.487	0.491	0.501	0.570	0.565	0.481	0.778	0.459
	720	**MAE ↓**	0.440	0.440	0.457	0.456	0.443	0.443	**0.433**	0.450	0.445	0.465	0.491	0.500	0.488	0.621	0.558	0.516	0.699	0.507
		MSE ↓	0.472	0.454	0.442	0.477	0.439	0.410	**0.402**	0.425	-	0.441	0.503	0.521	0.500	0.653	0.594	0.519	0.836	0.506
ETTh2	**96**	**MAE ↓**	0.310	**0.309**	0.328	0.342	0.328	0.340	0.335	0.338	0.356	0.333	0.349	0.374	0.348	0.584	0.440	0.387	0.621	0.397
		MSE ↓	0.273	0.272	**0.268**	0.285	0.269	0.276	0.278	0.274	-	0.273	0.297	0.340	0.302	0.745	0.400	0.333	0.707	0.358
	192	**MAE ↓**	0.356	**0.355**	0.375	0.389	0.374	0.371	0.373	0.370	0.400	0.378	0.400	0.414	0.400	0.656	0.509	0.476	0.689	0.439
		MSE ↓	0.339	0.338	**0.329**	0.354	0.332	0.331	0.345	0.330	-	0.340	0.380	0.402	0.388	0.877	0.528	0.477	0.860	0.429
	336	**MAE ↓**	0.387	**0.386**	0.409	0.407	0.395	0.402	0.402	0.396	0.428	0.403	0.432	0.541	0.433	0.731	0.571	0.541	0.744	0.487
		MSE ↓	0.374	0.372	0.368	0.373	**0.351**	0.373	0.384	0.362	-	0.373	0.428	0.452	0.426	1.043	0.643	0.594	1.000	0.496
	720	**MAE ↓**	0.400	**0.400**	0.420	0.441	0.430	0.431	0.437	0.417	0.457	0.430	0.445	0.657	0.446	0.763	0.679	0.657	0.838	0.474
		MSE ↓	0.375	0.374	0.372	0.406	0.390	0.404	0.437	**0.370**	-	0.398	0.427	0.462	0.431	1.104	0.874	0.831	1.249	0.463
ETTm1	**96**	**MAE ↓**	0.333	**0.313**	0.334	0.346	0.322	0.334	0.325	0.323	0.345	0.335	0.368	0.375	0.367	0.426	0.387	0.372	0.438	0.419
		MSE ↓	0.320	0.278	0.272	0.292	0.281	0.286	0.264	**0.256**	-	0.286	0.334	0.338	0.329	0.404	0.364	0.345	0.418	0.379
	192	**MAE ↓**	0.364	0.345	0.358	0.372	0.353	0.358	0.350	**0.343**	0.374	0.361	0.391	0.387	0.385	0.451	0.404	0.389	0.450	0.441
		MSE ↓	0.371	0.328	0.310	0.332	0.322	0.307	0.295	**0.281**	-	0.329	0.377	0.374	0.367	0.450	0.398	0.380	0.439	0.426
	336	**MAE ↓**	0.388	**0.368**	0.384	0.394	0.379	0.390	0.376	0.374	0.397	0.379	0.420	0.411	0.410	0.515	0.425	0.413	0.485	0.459
		MSE ↓	0.408	0.364	0.352	0.366	0.356	0.354	**0.323**	0.326	-	0.358	0.426	0.410	0.399	0.532	0.428	0.413	0.490	0.445
	720	**MAE ↓**	0.426	**0.403**	0.411	0.421	0.413	0.445	0.435	0.452	0.436	0.411	0.459	0.450	0.439	0.589	0.461	0.453	0.550	0.490
		MSE ↓	0.485	0.426	**0.383**	0.417	0.391	0.433	0.409	0.454	-	0.416	0.491	0.478	0.454	0.666	0.487	0.474	0.595	0.543
ETTm2	**96**	**MAE ↓**	0.237	**0.227**	0.253	0.262	0.256	0.265	0.259	0.273	0.263	0.250	0.264	0.267	0.259	0.366	0.305	0.292	0.377	0.287
		MSE ↓	0.172	**0.158**	0.161	0.173	0.169	0.172	0.169	0.183	-	0.164	0.180	0.187	0.175	0.287	0.207	0.193	0.286	0.203
	192	**MAE ↓**	0.280	**0.269**	0.293	0.301	0.294	0.306	0.295	0.301	0.309	0.288	0.309	0.309	0.302	0.492	0.364	0.362	0.445	0.328
		MSE ↓	0.232	**0.212**	0.219	0.229	0.225	0.228	0.223	0.223	-	0.218	0.250	0.249	0.241	0.414	0.290	0.284	0.399	0.269
	336	**MAE ↓**	0.320	**0.306**	0.329	0.341	0.334	0.345	0.341	0.339	0.349	0.322	0.348	0.351	0.343	0.542	0.422	0.427	0.591	0.366
		MSE ↓	0.290	**0.263**	0.271	0.286	0.278	0.281	0.293	0.278	-	0.271	0.311	0.321	0.305	0.597	0.377	0.369	0.637	0.325
	720	**MAE ↓**	0.375	**0.362**	0.379	0.401	0.392	0.424	0.433	0.424	0.415	0.380	0.407	0.403	0.400	1.042	0.524	0.522	0.735	0.415
		MSE ↓	0.372	**0.344**	0.352	0.378	0.372	0.403	0.451	0.425	-	0.361	0.412	0.408	0.402	1.730	0.558	0.554	0.960	0.421
Electricity	**96**	**MAE ↓**	0.211†	**0.205**†	0.224	0.238	0.218	-	-	-	-	0.221	0.240	0.272	0.285	0.314	0.329	0.282	0.345	0.308
		MSE ↓	0.125†	**0.121**†	0.131	0.139	0.126	-	-	-	-	0.128	0.148	0.168	0.195	0.219	0.237	0.197	0.247	0.193
	192	**MAE ↓**	0.228†	**0.223**†	0.241	0.251	0.237	-	-	-	-	0.243	0.253	0.289	0.289	0.322	0.330	0.285	0.355	0.315
		MSE ↓	0.145†	**0.142**†	0.152	0.153	0.144	-	-	-	-	0.150	0.162	0.184	0.199	0.231	0.236	0.196	0.257	0.201
	336	**MAE ↓**	0.244†	**0.238**†	0.248	0.266	0.256	-	-	-	-	0.259	0.269	0.300	0.305	0.337	0.344	0.301	0.369	0.329
		MSE ↓	0.157†	**0.153**†	0.160	0.169	0.162	-	-	-	-	0.166	0.178	0.198	0.215	0.246	0.249	0.209	0.269	0.214
	720	**MAE ↓**	0.284†	**0.264**†	0.298	0.297	0.286	-	-	-	-	0.276	0.317	0.320	0.337	0.363	0.373	0.333	0.390	0.355
		MSE ↓	0.207†	**0.185**†	0.192	0.206	0.192	-	-	-	-	**0.185**	0.225	0.220	0.256	0.280	0.284	0.245	0.299	0.246
Weather	**96**	**MAE ↓**	0.179	**0.165**	0.201	0.212	0.192	0.203	0.201	0.208	-	0.187	0.214	0.220	0.218	0.230	0.261	0.255	0.306	0.296
		MSE ↓	0.149	**0.134**	0.147	0.162	0.142	0.151	0.149	0.154	-	0.144	0.174	0.172	0.177	0.158	0.202	0.196	0.221	0.217
	192	**MAE ↓**	0.223	**0.211**	0.234	0.248	0.238	0.246	0.244	0.251	-	0.236	0.254	0.261	0.259	0.277	0.298	0.296	0.340	0.336
		MSE ↓	0.192	**0.177**	0.189	0.204	0.191	0.195	0.192	0.202	-	0.192	0.221	0.219	0.225	0.206	0.242	0.237	0.261	0.276
	336	**MAE ↓**	0.265	**0.253**	0.279	0.286	0.282	0.288	0.285	0.287	-	0.272	0.296	0.306	0.297	0.335	0.335	0.335	0.378	0.380
		MSE ↓	0.245	**0.225**	0.262	0.254	0.246	0.247	0.245	0.252	-	0.237	0.278	0.280	0.278	0.272	0.287	0.283	0.309	0.339
	720	**MAE ↓**	0.312	**0.302**	0.316	0.337	0.337	0.366	0.365	0.376	-	0.326	0.349	0.359	0.348	0.418	0.386	0.381	0.427	0.428
		MSE ↓	0.310	**0.288**	0.304	0.326	0.328	0.352	0.352	0.392	-	0.313	0.358	0.365	0.354	0.398	0.351	0.345	0.377	0.403
Best Count				**31**	**3**	**0**	**1**	**1**	**4**	**9**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table 12: Full-Shot Comparison of different models with TOTO on the LSF benchmark datasets for each prediction length, with TOTO's Zero-Shot result in the first data column.
Key: **Best results**, Second-best results. Values marked with † use a window stride equal to the prediction length on the Electricity dataset. "Best Count" row reports the number of times each model attains the best result for a given metric.

Model	Best NLL Loss (% increase) ↓
Control	**0.0%**
No Variate-wise Attention	1.6%
No Robust Loss	11.1%
No Student-T Mixture	27.2%
No Causal Scaling	27.3%

Table 13: Relative change in NLL on held-out observability pretraining data when removing key design features of the TOTO architecture.

27.3% in NLL when we replace the causal scaler with a naive global scaler. Replacing the Student-T mixture model with a single Student-T output causes a similar NLL increase of 27.2%. Interestingly, removing the robust loss component and optimizing NLL alone actually leads to a *worse* overall NLL, with an 11.1% increase; we speculate this is because the robust loss stabilizes the training, as discussed in Section 3.1. Finally, removing the variate-wise attention (i.e. making all the attention layers time-wise while holding the parameter count constant) leads to a more modest increase in NLL of 1.6%.