DELPHYNE: A Pre-Trained Model for General and Financial Time Series

Anonymous Author(s)
Affiliation
Address
email

Abstract

Time-series data is a vital modality within data science communities, particularly in financial applications, where it helps in detecting patterns, understanding market behavior, and making informed financial decisions based on historical data. Recent advances in language modeling have led to the rise of time-series pre-trained models that are trained on vast collections of datasets and applied to diverse tasks across financial domains. However, across financial applications, existing time-series pre-trained models have not shown promising performance boost over simple finance benchmarks in both zero-shot and fine-tuning settings. This phenomenon occurs because of a i) lack of financial data within the pre-training stage, and ii) the negative transfer effect due to inherently different time-series patterns across domains. Furthermore, time-series data is continuous, noisy, and can be collected at varying frequencies and different lags across variables, making this data more challenging to model than languages. To address the above problems, we introduce a Pre-trained Mo**DEL** for **FIN**ance Tim**E**-series (**Delphyne**). **Delphyne** achieves competitive performance to existing foundation and full-shot models with few fine-tuning steps on publicly available datasets, and also shows superior performances on various financial tasks.

1 Introduction

Time series is one of the most ubiquitous modalities in finance. Time-series analysis is critical to various tasks, such as asset pricing, volatility modeling, risk management, economic indicator analysis, etc. In recent years, deep learning-based methods are being applied to these financial tasks (e.g., [21, 3, 26, 16]). However, previous research [52, 24, 7, 19, 47, 8, 18] shows that directly prompting LLMs for financial tasks brings only modest benefits over traditional methods like Generalized Autoregressive Conditional Heteroskedasticity (GARCH) in financial tasks.

We believe that lack of financial data in existing public pre-training datasets is one of the key issues since they have very different distributional patterns. However joint training on substantially different domains can lead to *negative transfer*. Negative transfer has been extensively discussed within the literature, defined as a case when "transferring knowledge from the training data has an negative impact on the target tasks" [45]. This has been identified as a key obstacle towards pre-training graph foundation model [43, 28], yet is not explored within the context of time-series data. In Sec. J, we show that the negative transfer effect is a real challenge when pre-training time-series models. Cross-domain transfer learning is difficult, as time-series data tends to be noisier and more continuous compared to languages and images. To alleviate the negative transfer effect, we believe that fine-tuning is the only remedy. We contend that the strength of pre-trained time-series models lies in their capacity to rapidly "unlearn" the biases of the pre-training stage and "adapt" to the specific distribution of new tasks, given limited training data and time. Building on our analysis of negative transfer, we introduce our Pre-Trained Mo**DEL** for **FIN**ance Tim**E**-series (**Delphyne**), the first time-series model capable of both general and finance-specific tasks.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

2 Negative Transfer Effect

Negative transfer has been widely studied in foundation models across different modalities [43]. This problem is typically seen as the model's reduced performance on downstream tasks due to mismatches between the source training data and the target distribution [45]. However, this phenomenon has not been thoroughly explored in the context of time series. In pre-trained time-series models, negative transfer can occur when cross-domain data is added during pre-training. The appearances of data from too different distributions can lead to less effective zero-shot forecast results, even if we feed the downstream tasks within similar domains. We present few examples to highlight the presence of negative transfer, in particular the difficulty of cross-domain transfer from other areas to finance data.

Pre-training with GARCH and Wavelet Data To simulate datasets encountered in real-world scenarios, we generate two types of synthetic data: Wavelet functions and GARCH-style data. Wavelet functions are composed of a combination of sine and cosine waves, while in GARCH models the current time-steps are based on past squared residuals and past variances, frequently used to capture volatility clustering seen in financial time-series data. [5, 8, 33] We train models on GARCH data only, Wavelet data only, and then train on combination of both. Each model utilizes a standard autoregressive transformer decoder without any additional embeddings or patching. Further details can be found in the Appendix J.1. Table 1 presents the negative log-likelihood (NLL) for each model's zero-shot forecasts with context lengths of 128. As expected, the mixed-data models performs significantly worse in terms of zero-shot NLL compared to the models trained on a single data source, with performance getting worse with increasing mix ratio, Appendix J.3.

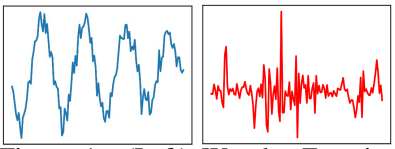

Figure 1: (Left) Wavelet Function. (Right) GARCH-style data.

Table 1: Zero-shot NLL(↓) for models trained on different data types.

	Model	Wavelet Pred.	GARCH Pred.
Context Len= 128	GARCH Only	–	**0.0882**
	25 % Wavelet	-0.0445	0.1636
	50 % Wavelet	-0.0732	0.1176
	75 % Wavelet	-0.0780	0.1742
	Wavelet Only	**-1.3300**	–

Transformer Training We observe similar effects while training our Delphyne model. During the pre-training phase, we evaluate different checkpoints to assess the zero-shot and fine-tuned forecast performance on the ETTh2 dataset for two versions of our model: Delphyne-A (trained with the LOTSA and financial data) and Delphyne-L (trained without financial data). We record the average Mean Absolute Error (MAE) across forecast lengths of 96 and 192. Initially, Delphyne-A incurs a higher MAE than Delphyne-L, but after fine-tuning, both models achieve comparable MAE. So the strength of a pre-trained time-series model lies in it's ability to swiftly adapt to new task distributions by efficiently "unlearning" pre-training biases and "adapting" to the specific characteristics of downstream tasks.

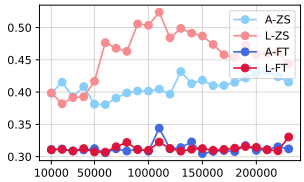

Figure 2: Average MAE on ETTh2 across training epochs.

3 Delphyne Model

Problem Formulation For both single and multi-variate times-series, we assume that each dataset $\mathcal{D} = \{\mathbf{Y}^{(i)}\}_{i=1}^{M}$ has M data points, while each $\mathbf{Y}^{(i)} \in \mathbb{R}^{l(i) \times T_{\mathbf{Y}^{(i)}}}$, contains $l(i)(l(i) \geq 1)$ variates and $T_{\mathbf{Y}^{(i)}}$ time-steps. For each variate j, the future $h_j \geq 0$ time-steps are modeled as $P(\mathbf{Y}_{T_\mathbf{Y}-\mathbf{h}:T_\mathbf{Y}}|\phi)$, where ϕ is the output distribution of the time-series forecasting model and $\mathbf{h}$ is a vector comprised of h_j time-steps. Time-series data, particularly in finance, exhibits unique characteristics that present distinct challenges: (1) **Multivariate Nature**: multiple interrelated time series, e.g, US stocks are often quite correlated with the S&P 500 index. (2) **Nowcasting Data**: The estimation of current value of a time-series based on it's own history and current values of other variables. (3) **Multifrequency and Missing Data**: Each $\mathbf{Y}$ variable can be collected at varying time granularities or may contain missing entries due to irregular sampling intervals. (4) **Extended Context Length**: Financial time-series data often span thousands of timesteps, requiring models to handle significantly longer temporal dependencies.

3.1 Study of Transformer Architecture for Time-series

Following recent approaches [47, 18], we adopt a transformer encoder structure (see details in Appendix B). Fig. 3 shows the overall pipeline. Below we describe some of the architectural changes:

Missing and Forecast Masking Before Patching. Recent work shows that patching time series allows the model to attend to significantly longer contexts [30]. Therefore, Delphyne breaks the flattened time series into disjoint patches after right-padding the shorter time-series, fixing the patch

size to 32. Contrary to prior work, we create a `[FORECAST MASK]` to identify the target timesteps for forecast as well as a `[MISSING MASK]` that indicates where data is unobserved due to the sampling procedure (e.g., daily stock prices not being recorded on holidays). Simply ignoring these gaps can misalign multifrequency data, while backfilling or zero-filling can distort the original distribution. Missing data and forecast masks are treated similarly during training; however, missing values are excluded from the forecast process. We apply both masks alongside the time-series data to learn a trainable linear projection embedding that incorporates both data and missingness information.

Context Length and Masking Ratio We perform ablations studies to explore the impact of context length and masking ratio on training efficiency. This ratio is crucial as it affects input context length, impacting generalization and fine-tuning. Table 2 reports the NLL for zero-shot and fine-tuning results from ablation study (details in Appendix K.1). While pre-training with a context length of 32 yields the best zero-shot performance, a longer context length of $64, 128$ improves fine-tuning, especially with fewer fine-tuning (10-100) samples. This suggests that for effective fine-tuning, pre-trained time-series models benefit from longer context lengths during pre-training. Table 3 illustrates the results from masking ratio ablation study (details in Appendix K.2). The model consistently outperforms when masking less aggressively. During model pre-training, we apply independent masking to each variate (average masking ratio of 30%), so the model can better adapt to nowcasting scenarios where variates can have different context lengths.

Table 2: NLL(↓) for zero-shot and fine-tuning with varying sample sizes.

Model	Zero-Shot	1 Sample	10 Samples	100 Samples	1000 Samples
Medium-16	-0.0483	-0.0893	-0.1542	-0.1767	-0.1897
Medium-32	**-0.1330**	**-0.1486**	-0.1673	-0.1792	-0.1847
Medium-64	-0.0793	-0.1146	**-0.1612**	**-0.1873**	-0.1875
Medium-128	-0.1020	-0.1113	-0.1711	-0.1843	**-0.1899**

Table 3: NLL(↓) for varying pre-training masking ratios.

	Sample Size	Masking Ratio 0.25	0.5	0.99
Pred. Len. 32	1	**-0.441**	-0.187	0.416
	10	**-0.394**	-0.071	0.496
	100	**-0.580**	-0.391	0.321
	1000	-0.666	-0.662	**-0.676**
Pred. Len. 64	1	**-0.263**	-0.077	0.122
	10	**-0.236**	-0.077	0.144
	100	**-0.296**	-0.158	0.088
	1000	**-0.318**	-0.314	-0.318

Multivariate Data Multivariate time-series modeling poses unique challenges, particularly in capturing correlations between variates. Many pre-trained models adopt different approaches like channel-independence, channel-mixing or any-variate attention [30, 18, 10, 2, 8, 34, 47] (see Appendix K.3). Table 4 reports that multivariate models outperform univariate ones on correlated Wavelet data, highlighting the value of modeling inter-variable dependencies.To handle both strongly and weakly correlated financial returns, Delphyne uses an any-variate attention mechanism that integrates cross-channel information without forced mixing.

Output Distribution For probabilistic forecasting, previous studies assume a fixed output distribution [35]. However, when the data presents varying distributions and supports, this may be insufficient. We use a mixture of Student's T distributions to model the output which helps model the fat-tail scenarios in financial tasks (details in Appendix K.4)

Table 4: NLL(↓) for zero-shot and fine-tuning with varying sample sizes, for different modeling multivariate methods.

	Model	Zero-Shot	1 Sample	10 Samples	100 Samples	1000 Samples
Corr.	Univariate	-0.0978	-0.0842	-0.1681	-0.1873	-0.1898
	Channel Mixing	**-0.1531**	**-0.1650**	**-0.1797**	-0.1873	**-0.1913**
	Any-variate Attention	-0.1513	-0.1507	-0.1794	**-0.1892**	-0.1895
Uncorr.	Univariate	**-0.0978**	-0.0842	-0.1681	-0.1873	**-0.1898**
	Channel Mixing	-0.0928	-0.1323	-0.1722	**-0.1892**	-0.1874
	Any-variate Attention	-0.0922	**-0.1386**	**-0.1879**	-0.1879	-0.1886

3.2 Training Details

Training Data Delphyne is trained on carefully sampled public LOTSA data (see Appendix C.1, C.3) and financial data, allowing Delphyne to generalize well to daily time-series forecast tasks as well as financial time-series tasks. Our financial dataset contains data for companies, stocks, ETFs, currencies (exchange rates), and commodities with multiple frequencies (including intraday and monthly). To ensure that there is no lookahead bias in our downstream tasks, we pre-train with data only until the end of 2019. We provide a detailed breakdown of the dataset in Appendix C.2. We streamline our preprocessing with variate subsampling and truncating timesteps to 512×32.

Model Parameters We train with 12 layers and 768 dimensional attention with 12 heads. Dropout is set to 0.2, and the model is trained on negative log-likelihood (NLL) loss. We pretrained for 1 million gradient updates with a fixed patch size of 32 and a sequence length of 512×32 steps. Using a batch size of 256, we optimize with AdamW (learning rate = $1e-4$, weight decay = 0.1, $\beta_1 = 0.9$, $\beta_2 = 0.98$) and apply a learning rate scheduler with $10,000$ steps of linear warmup and cosine annealing to $1e-5$. Training was conducted on 8 H100 GPUs over 4 days with mixed-precision.

4 Experiments

We train three models for downstream evaluations. Delphyne-A trained on the LOTSA dataset [47] only, Delphyne-F trained on finance data only, and Delphyne-A trained on both. We compare the zero-shot (ZS) forecasts and fine-tuning (FT) performances on various standard and financial time-series tasks and datasets. For conciseness, we discuss only the financial examples here; the other experiments (Monash short-term forecasting, out-of-distribution long-term forecasting, probability quantification and UCR anomaly detection) are reported in Appendices F–I.

Stocks. While a commonly evaluated task for finance is forecasting the returns, modeling the distribution of stock returns is equally important where it can be used for stress-testing scenarios, and conducting risk analysis [40]. We use the daily stock returns of 14 major stocks from SPX Index from 2021-01-04 to 2023-12-29. For fine-tuning, we use data from 1996-07-01 to 2019-12-31 for training, 2020-01-02 to 2020-12-31 for validation. Since most methods, except MOIRAI [47], forecast point estimates, we conduct two experiments: forecasting next-day stock returns' variance (volatility) to compare MSE, and evaluating NLL for the forecasted returns distribution. We also compare against GARCH with Student's T, a standard financial baseline [40] and PatchTST [30] (adapted the output to a mixture of four Student's T). Table 5 and 6 show the overall results which indicate that while utilizing only financial data in pre-training brings best zero-shot performance (Delphyne-F), Delphyne-A achieves the best results. Additional coverage statistics, R^2 calculation and comparison to the Fama-French factor model (Appendix E) confirm Delphyne's superior performance.

Bars. We use the intraday bars data, which contains the log of volume traded in five-minute intervals to test the different methods' performances in long-sequence modeling. For 4 different ETFs, we use the past 15 days data (context length 15×78) to forecast the log-volume in the next day's trading hours (e.g., forecast length of 78) and compare their mean-squared errors. Table 7 shows the results for 2021-01-04 to 2021-01-11. For fine-tuning, data from 2008-01-24 to 2019-12-30 is used for training, and 2019-12-31 to 2020-12-31 for validation. All Delphyne models significantly improve metrics after fine-tuning, effectively capturing the seasonal component in bar log-volume data. Delphyne-A-ZS outperforms Delphyne-F-ZS and Delphyne-L-ZS due to the data's seasonal nature, similar to electricity and weather datasets. However, with fine-tuning, Delphyne-F achieves the best performance across all methods.

Table 5: Likelihood results for next-day stock returns risk analysis.

Model	NLL ZS	NLL FT
Delphyne-A	1.762	**1.741**
Delphyne-F	1.750	<u>1.746</u>
Delphyne-L	1.775	1.757
MOIRAI	1.776	1.788
GARCH	-	1.752
PatchTST	-	1.751

Table 6: MSE for next-day stock squared returns (variance).

Model	MSE ZS	MSE FT
Delphyne-A	37.792	37.810
Delphyne-F	<u>37.653</u>	38.616
Delphyne-L	**37.591**	38.246
MOIRAI	41.428	40.502
MOMENT	46.006	37.785
TTM	44.918	44.360
PatchTST	-	51.705
GARCH	-	41.517

Table 7: MSE for bars log-volume data. (78 timestep predictions of 5-minute intervals)

Model	MSE ZS	MSE FT
Delphyne-A	0.728	0.551
Delphyne-F	0.965	**0.530**
Delphyne-L	0.930	0.557
MOIRAI	0.767	0.620
MOMENT	0.775	0.838
TTM	0.714	0.601
PatchTST	-	<u>0.534</u>
Avg past values	0.602	-

Table 8: Nowcasting results for zero-shot vs. fine-tuning for company sales growth

Model	MAE ZS	MAE FT
Delphyne-A	0.099	**0.071**
Delphyne-F	0.128	0.079
Delphyne-L	0.101	<u>0.073</u>
MOIRAI	0.091	0.093
Baseline	0.100	-

Nowcasting Company Revenue. We use consumer transaction data to test nowcasting performance (e.g., when we have contemporaneous data), forecasting year-over-year (YoY) sales growth for 211 U.S. companies based on previous quarter's YoY sales growth, previous YoY transactions growth, and current quarter's YoY growth in transactions. Due to the quarterly nature, the context length is short (4-8). We make rolling forecasts for Q3 2022 to Q1 2023. For fine-tuning, data from 2018 Q1 to 2021 Q1 is used for training, and 2021 Q2 to 2022 Q2 for validation. We compare against a statistical baseline built by the providers of the consumer transaction data and MOIRAI [47]. Table 8 shows MSE results: Delphyne-A with fine-tuning outperforms all methods, and Delphyne-L ranks second despite not seeing the data during pre-training. We attribute this to Delphyne-L's independent masking of variates, which enhances handling of contemporaneous data.

5 Conclusions

We illustrate the presence of negative transfer effect in pre-trained time-series models especially when pre-trained with time-series data from various domains, contrasting them with LLMs for language tasks. Our experiments emphasize the role of fine-tuning to counter this effect which helps pre-trained time-series models to adapt to diverse downstream tasks with few training samples and minimal iterations. We introduce various architectural modifications, supported by ablation studies, to handle continuous, noisy, multivariate and multifrequency nature of time-series data. Delphyne is the first pre-trained model excelling in both general time-series and diverse financial and economic tasks such as nowcasting.

References

[1] 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`.

[2] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Syndar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the Language of Time Series. *arXiv preprint arXiv:2403.07815*, 2024.

[3] Gustavo Silva Araujo and Wagner Piazza Gaglianone. Machine Learning Methods for Inflation Forecasting in Brazil: New Contenders versus Classical Models. *Latin American Journal of Central Banking*, 4(2):100087, 2023. doi: 10.1016/j.latcb.2023.100087.

[4] V. Assimakopoulos and K. Nikolopoulos. The Theta Model: A Decomposition Approach to Forecasting. *International Journal of Forecasting*, 16(4):521–530, 2000. doi: 10.1016/S0169-2070(00)00066-2.

[5] Tim Bollerslev. Generalized Autoregressive Conditional Heteroskedasticity, journal = Journal of Econometrics. 31(3):307–327, 1986. doi: 10.1016/0304-4076(86)90063-1.

[6] David Campos, Miao Zhang, Bin Yang, Tung Kieu, Chenjuan Guo, and Christian S. Jensen. LightTS: Lightweight Time Series Classification with Adaptive Ensemble Distillation. *Proc. ACM Manag. Data*, 1(2):171:1–171:27, 2023. doi: 10.1145/3589316.

[7] Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

[8] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10148–10167. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/das24c.html`.

[9] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. CatBoost: Gradient Boosting with Categorical Features Support. *CoRR*, abs/1810.11363, 2018. URL `http://arxiv.org/abs/1810.11363`.

[10] Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam H. Nguyen, Wesley M. Gifford, Chandra Reddy, and Jayant Kalagnanam. Tiny Time Mixers (TTMs): Fast Pre-trained Models for Enhanced Zero/Few-Shot Forecasting of Multivariate Time Series, 2024. URL `https://arxiv.org/abs/2401.03955`.

[11] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning. *Neural Networks (Special issue on reinforcement learning)*, 107:3–11, 2018. doi: 10.1016/j.neunet.2017.12.012.

[12] Patrick Emami, Abhijeet Sahu, and Peter Graf. BuildingsBench: A Large-Scale Dataset of 900K Buildings and Benchmark for Short-Term Load Forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL `https://openreview.net/forum?id=c5rqd6PZn6`.

[13] Azul Garza and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

[14] Tilmann Gneiting and Adrian E Raftery. Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[15] R. W. Godahewa, C. Bergmeir, G. I. Webb, R. Hyndman, and P. Montero-Manso. Monash Time Series Forecasting Archive. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL `https://openreview.net/forum?id=wEc1mgAjU-`.

[16] Achintya Gopal. Neuralfactors: A novel factor learning approach to generative modeling of equities. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 99–107, 2024.

[17] Mononito Goswami, Cristian Ignacio Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. Unsupervised Model Selection for Time Series Anomaly Detection. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gOZ_pKANaPW.

[18] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A Family of Open Time-series Foundation Models, 2024. URL https://arxiv.org/abs/2402.03885.

[19] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. Large Language Models Are Zero-Shot Time Series Forecasters. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 19622–19635. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/3eb7ca52e8207697361b2c0fb3926511-Paper-Conference.pdf.

[20] Matthew D. Hoffman and Andrew Gelman. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47): 1593–1623, 2014. URL http://jmlr.org/papers/v15/hoffman14a.html.

[21] Blanka Horvath, Aitor Muguruza, and Mehdi Tomas. Deep learning volatility. *arXiv preprint arXiv:1901.09647*, 2019.

[22] Rob J Hyndman. Errors on Percentage Errors, 4 2014. URL https://robjhyndman.com/hyndsight/smape/.

[23] Rob J Hyndman and Anne B Koehler. Another Look at Measures of Forecast Accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.

[24] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time Series Forecasting by Reprogramming Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Unb5CVPtae.

[25] Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, LEI BAI, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=loOw3oyhFW.

[26] Yuxin Liu, Jimin Lin, and Achintya Gopal. Neuralbeta: Estimating beta using deep learning. *arXiv preprint arXiv:2408.01387*, 2024.

[27] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1): 54–74, 2020.

[28] Haitao Mao, Zhikai Chen, Wenzhuo Tang, Jianan Zhao, Yao Ma, Tong Zhao, Neil Shah, Mikhail Galkin, and Jiliang Tang. Position: Graph Foundation Models Are Already Here. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 34670–34692. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/mao24a.html.

[29] S. Mouatadid, P. Orenstein, G. E. Flaspohler, M. Oprescu, J. Cohen, F. Wang, S. E. Knight, M. Geogdzhayeva, S. J. Levang, E. Fraenkel, and L. Mackey. SubseasonalclimateUSA: A Dataset for Subseasonal Forecasting and Benchmarking. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=pWkrU6raMt.

[30] 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.

[31] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-BEATS: Neural Basis Expansion Analysis for Interpretable Time Series Forecasting. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1ecqn4YwB.

[32] Youngsuk Park, Danielle Maddix, François-Xavier Aubet, Kelvin Kan, Jan Gasthaus, and Yuyang Wang. Learning Quantile Functions without Quantile Crossing for Distribution-Free Time Series Forecasting. In *International Conference on Artificial Intelligence and Statistics*, pages 8127–8150. PMLR, 2022.

[33] Alessio Petrozziello, Luigi Troiano, Angela Serra, Ivan Jordanov, Giuseppe Storti, Roberto Tagliaferri, and Michele La Rocca. Deep Learning for Volatility Forecasting in Asset Management. *Soft Computing*, 26(17):8553–8574, 2022. doi: 10.1007/s00500-022-07161-1.

[34] 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.

[35] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. *International Journal of Forecasting*, 36 (3):1181–1191, 2020. doi: 10.1016/j.ijforecast.2019.07.001.

[36] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. *International Journal of Forecasting*, 36 (3):1181–1191, 2020. doi: 10.1016/j.ijforecast.2019.07.001.

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

[38] 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.

[39] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomput.*, 568(C), March 2024. doi: 10.1016/j.neucom.2023.127063.

[40] Ruslan Tepelyan and Achintya Gopal. Generative Machine Learning for Multivariate Equity Returns. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, ICAIF '23, page 159–166, New York, NY, USA, 2023. Association for Computing Machinery. doi: 10.1145/3604237.3626884.

[41] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. *CoRR*, abs/1609.03499, 2016. URL http://arxiv.org/abs/1609.03499.

[42] J. Wang, J. Jiang, W. Jiang, C. Han, and W. X. Zhao. Towards Efficient and Comprehensive Urban Spatial-Temporal Prediction: A Unified Library and Performance Benchmark. *arXiv preprint arXiv:2304.14343*, 2023.

[43] Zehong Wang, Zheyuan Zhang, Chuxu Zhang, and Yanfang Ye. Subgraph Pooling: Tackling Negative Transfer on Graphs. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 5153–5161. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/570. Main Track.

[44] Zhixian Wang, Qingsong Wen, Chaoli Zhang, Liang Sun, Leandro Von Krannichfeldt, Shirui Pan, and Yi Wang. Benchmarks and Custom Package for Electrical Load Forecasting, 2024. URL https://openreview.net/forum?id=gjB7qqPJbv.

[45] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and Avoiding Negative Transfer. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11285–11294, 2019. doi: 10.1109/CVPR.2019.01155.

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

[47] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified Training of Universal Time Series Forecasting Transformers. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 53140–53164. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/woo24a.html.

[48] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ju_Uqw384Oq.

[49] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On Layer Normalization in the Transformer Architecture. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10524–10533. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/xiong20b.html.

[50] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/85422afb467e9456013a2a51d4dff702-Paper.pdf.

[51] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/zhou22g.html.

[52] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One Fits All: Power General Time Series Analysis by Pretrained LM. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=gMS6FVZvmF.

A Any-variate Attention

Any-variate attention is first proposed by [47] to allow binary attention biases to encode variate indices for a flattened multi-variate time series. The attention score between the (i, m)-th query and (j, n)-th query (j and i represent the time-steps, and n and m encode the variate index) is calculated as the following:

$$E_{ij,mn} = \left(\mathbf{W}^Q \mathbf{x}_{i,m}\right)^T \mathbf{R}_{i-j} \left(\mathbf{W}^K \mathbf{x}_{j,n}\right) + u^{(1)} * \mathbb{1}_{\{m=n\}} + u^{(2)} * \mathbb{1}_{\{m \neq n\}}, \tag{1}$$

$$A_{ij,mn} = \frac{\exp(E_{ij,mn})}{\sum_{k,o} \exp(E_{ik,mo})}, \tag{2}$$

where $\mathbf{W}^Q \mathbf{x}_{i,m}, \mathbf{W}^K \mathbf{x}_{j,n} \in \mathbb{R}^{d_h}$ are the query and key vectors. $u^{(1)}$ and $u^{(2)}$ are learnable scalars as the attention biases. These binary attention biases component enables differentiation between variates, satisfies permutation equivariance and invariance with respect to variate ordering, and is scalable to any number of variates.

B Model Overview

We utilize pre-normalization [49], rotary positional embedding [39], any-variate attention (Sec. **??**), Silu activation function [11] and gated linear unit (GLU) [37] to replace FFN.

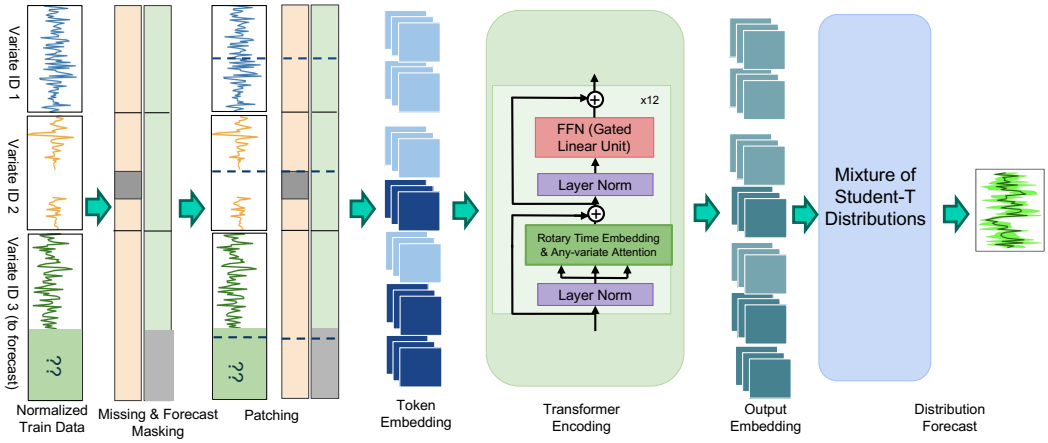

Figure 3: Delphyne Overview

C Pre-training Data

Table 9: Sampling dataset probability (%) across LOTSA domains and finance data.

LOTSA	Energy	Transport	CloudOps	Climate	Econ/fin	Web	Sales	Nature	Healthcare	**Total**
	17.8	25.5	8.6	11.9	5.7	6.1	5.4	3.9	0.2	**85**
Finance Data		ETFs	Tickers	Commodities	Currency	Stock	Company	Intraday Bars		**Total**
		2.8	2.8	0.3	0.9	2.8	1.8	2.8		**15**

C.1 LOTSA

1. **BuildingsBench** BuildingsBench [12] comprises of datasets detailing energy consumption in residential and commercial buildings. These include the real-world BDG-2 datasets, Low Carbon London, SMART, IDEAL, Sceaux, and Borealis, which capture energy usage from diverse sources. BuildingsBench introduces the Buildings-900K dataset, a large-scale simulation of 900K buildings, while both the training and testing splits are included in LOTSA. Electricity is omitted in LOTSA and used for out-of-distribution evaluation.

2. **ClimateLearn** This dataset includes both ERA5 and CMIP6 [30], which contain various climate-related variables like temperature and humidity across different pressure levels. In LOTSA, we observe that ERA5 and CMIP6 are divided into several data folders across different years. To address this, we reduce the probability of their appearance by treating all directories spanning multiple years as single datasets.
3. **CloudOps** CloudOps-TSF, introduced by [46], provides three large-scale time series datasets that capture variables like CPU and memory utilization. Only training dataset is included in LOTSA.
4. **GluonTS [1]** For this dataset, only Taxi, Uber TLC Daily, Uber TLC Hourly, Wiki-Rolling, and M5 are included. The rest of the datasets are already included in the Monash dataset.
5. **LargeST [25]** This dataset contains traffic datasets from California Department of Transportation Performance Measurement System (PeMS). PeMS includes PEMS03, PEMS04, PEMS07, PEMS08, PEMS Bay, and the well-known Traffic dataset.
6. **LibCity [42]** This is a collection of urban spatio-temporal datasets, while the spatial aspect is dropped.
7. **Monash** The Monash Time Series Forecasting Repository [15] is a comprehensive collection of diverse time series datasets. The test data for each dataset is the final horizon as the test set, while the forecast horizon is defined for each individual dataset. LOTSA includes the training data of Monash dataset, holding out the testing for in-distribution evaluation. Several datasets are included entirely in LOTSA: London Smart Meters, Wind Farms, Wind Power, Solar Power, Oikolab Weather, Covid Mobility, Extended Web Traffic, Kaggle Web Traffic Weekly, M1 Yearly, M1 Quarterly, M3 Yearly, M3 Quarterly, M4 Yearly, M4 Quarterly, Tourism Yearly. In our experiment evaluation, we do not fine-tune Delphyne on several datasets in Monash, due to that their training data is very short (< 20 time steps) after splitting the data into test data and validation data.
8. **ProEnFo** ProEnFo [44] is a dataset for load forecasting. Its data contains various covariates such as temperature, humidity, and wind speed.
9. **SubseasonalClimateUSA [29]** This dataset offers climate time series data for subseasonal forecasting. LOTSA contains Subseasonal Precipitation, containing precipitation data from 1948 to 1978, and Subseasonal, which includes both precipitation and temperature data from 1979 to 2023.
10. **Other** LOTSA also contains datasets from miscellaneous sources, spanning from energy, econ/finance, sales and healthcare. Refer to Table 17 in [47] for details.

C.2 Financial Data

By design, our data sampling samples time-series from the same dataset with the same sample ID; because of this, some of our datasets have the same time-series but are used in different contexts (sampled with different time series).

1. **Single Currency Daily** This dataset includes 12 exchange rates, and each exchange rate is treated as a separate sample. For each exchange rate, we include the time series of the exchange rate and its returns, forward rates, and implied volatilities. We use 1W, 1M, 3M, 6M, 9M, 1Y, 18M, and 2Y for the tenors, and 0.1, 0.15, 0.25, 0.35, 0.5, 0.65, 0.75, 0.85, and 0.9 for the deltas. This dataset trains the model to properties across the spot, forward, and volatility surface.
2. **Joint Currencies Daily** This data includes 68 currency pairs, the exchange rate and returns are the time series. This dataset is to allow our model to learn correlations across currencies; to this end, there is only one sample.
3. **Currencies Monthly** This dataset includes 43 exchange rates, and each exchange rate is treated as a separate sample. We use the same columns as in Single Currency Daily except the returns are monthly returns.
4. **Commodities Daily** This dataset includes 29 commodities, and each commodity is treated as a separate sample. Similar to exchange rates, we include the price and returns of the commodity, and the implied volatilities for 1M, 2M, 3M, 6M, 1Y, 18M, and 2Y for the tenors, and 90%, 95.0%, 97.5%, 100.0%, 102.5%, 105.0%, and 110.0% for the moneyness.

Table 10: List of commodities tickers

BO1	HO1	QS1
CC1	JO1	SB1
CL1	KC1	SI1
CO1	KO1	S
CT1	LA1	TZT1
CU1	LB1	UXA1
C	LP1	W
FN1	MO1	XB1
GC1	NG1	XW1
HG1	PL1	

Table 11: List of company financials fields

is_sg&a_expense	net_income	is_cogs_to_fe_and_pp_and_g
is_sales_and_services_revenues	is_other_operating_expenses	is_cog_and_services_sold
is_operating_expn	ebit	sales_rev_turn
ebitda	short_and_long_term_debt	bs_tot_asset
bs_cur_liab	bs_cur_asset_report	bs_gross_fix_asset
total_equity	bs_pfd_eqty_&_hybrid_cptl	bs_inventories
bs_cash_near_cash_item	bs_lt_invest	bs_net_fix_asset
bs_acct_note_rcv	cash_and_marketable_securities	enterprise_value
net_debt	sales_to_net_fix_asset	gross_profit
num_of_employees	gross_margin	historical_market_cap
avg_age_of_assets_in_years	cf_cap_expend_prpty_add	cf_cash_from_inv_act

Table 12: List of consumer transactions fields

observed_sales	observed_transactions	observed_unique_customers
average_transaction_value	transactions_per_customer	sales_per_customer

5. **Commodities Monthly** This dataset is identical to 'Commodities Daily' except that the data is resampled to monthly level where we take monthly returns and the last value per month for the rest of the columns.
6. **Joint Stock Returns** Similar to 'Joint Currencies Daily', this dataset is to allow our model to learn correlations across stocks. This data includes the returns of 10,511 stocks. To ensure the correlations learned are more meaningful, we partitioned the stocks into 53 exchanges.
7. **Daily ETFs Returns** Similar to 'Joint Stock Returns', this dataset is to allow our model to learn correlations across ETFs. This data includes the returns of 28,837 ETFs. To ensure the correlations learned are more meaningful, we partitioned the stocks into 76 exchanges.
8. **Company Data** Similar to 'Single Currency Daily', this dataset is to allow our model to learn correlation across stock features. This data includes 10,511 stocks. For variates, we include stock returns, volume traded, quarterly company financials, consumer transaction data, forward rates for the same tenors as 'Single Currency Daily', and implied volatilities for the same moneynesses as 'Commodities Daily' as well as 80% and 120% moneyness.
9. **Intraday Bars** We include 15,817 global securities and for each five-minute interval (bar), we include the open, high, low, and closing price, the volume traded, and the number of trades. Since the securities have different open and close hours, to normalize the data, we drop specific five-minute intervals (a specify day of the week and time) for which the ticker has had zero trades through its life.

C.3 Sampling

The LOTSA dataset is significantly imbalanced, necessitating subsampling to ensure more balanced representation during training. We carefully identify the few datasets that dominate in size and reduce their likelihood of being sampled to avoid overrepresentation. For any dataset, we first

compute the total number of observations (across samples, variates, and timesteps) within the dataset, $|\mathcal{D}_k| = \sum_{i=1}^{M} \sum_{j=1}^{l+k} T_{i,j}$. Then, we normalize the scores to sum to one and cap the minimum weight to 0.001, to obtain the final sampling probabilities. Overall, our training data consists of 85% from LOTSA and 15% from financial data (Table 9). We sample the number of variates (≤ 128) using a beta-binomial distribution ($\alpha = 2$ and $\beta = 5$).

D Monash Time Series Forecast

D.1 Comparison Methods

Pre-trained Models. For pre-trained models we report the zero-shot performance of MOIRAI [47]. MOIRAI is a unified pre-trained foundation model for time-series analysis. Across the three versions of MOIRAI, we report it performance across $\text{MOIRAI}_{\text{Base}}$, which is roughly the same amount of parameters as our Delphyne models.

Baselines. Several traditional and statistical methods serve as the reported baseline for Monash, using the last observed value for the forecast. SES (Single Exponential Smoothing) applies a weighted average to past observations, with exponentially decreasing weights for older data points. Theta [4] fits θ-lines with exponential smoothing. Exponential Smoothing (ETS) is also a traditional statistical method.

Non-deep Methods. The non-deep learning methods include CatBoost (gradient boosting on decision trees) [9], (DHR)-ARIMA (dynamic harmonic regression), PR.

Deep Methods. Methods that including training neural networks include N-BEATS [31], feed-forward neural network (FFMM), DeepAR [36], N-BEATS [31], WaveNet [41] and Transformer (Trans).

GPT3.5 & Llama2. GPT3.5 and Llama2 are two versions of LLMTime. For GPT-3.5, we report the reproduced results by [47], as well as the original results by [19] run on Llama2.

D.2 Full Comparison Results

See Table 13 for a comparison across baselines and Table 14 for comparison across different versions of Delphyne.

Table 13: Full results of Monash Time Series Forecasting Benchmark. MAE is reported. The best result is in **bold**. "**Aggregated**" means that we take the geometric mean of the MAE of each dataset divided by the MAE of the Naive approach (for zero-shot models only, fine-tuned performances are reported in Table 14).

Dataset	Delphyne-A-ZS	Delphyne-A-FT	MOIRAI	Naive	SES	Theta	TBATS	ETS	(DHR-)ARIMA	PR	CatBoost	FFNN	DeepAR	N-BEATS	WaveNet	Trans.	GPT3.5	LLaMA2
M1 Monthly	2153.37	-	2068.63	2707.75	2259.04	2166.18	2237.50	1905.28	2080.13	2088.25	2052.32	2162.58	1860.81	**1820.37**	2184.42	2723.88	2562.84	-
M3 Monthly	649.77	-	658.17	837.14	743.41	**623.71**	630.59	626.46	654.8	692.97	732	692.48	728.81	648.6	699.3	798.38	877.97	-
M3 Other	202.44	-	198.62	278.43	277.83	215.35	**189.42**	194.98	193.02	234.43	318.13	240.17	247.56	221.85	245.29	239.24	300.30	-
M4 Monthly	597.43	**560.47**	592.09	671.27	625.24	563.58	589.52	582.6	575.36	596.19	611.69	612.52	615.22	578.48	655.51	780.47	728.27	-
M4 Weekly	378.36	306.06	328.08	347.09	336.82	333.32	296.15	335.66	321.61	293.21	364.65	338.37	351.78	**277.73**	359.46	378.89	518.44	-
M4 Daily	223.54	181.89	192.66	180.83	178.27	178.86	**176.6**	193.26	179.67	181.92	231.36	177.91	299.79	190.44	189.47	201.08	266.52	-
M4 Hourly	218.85	211.93	**209.87**	1218.06	1218.06	1220.97	386.27	3358.10	1310.85	257.39	285.35	385.49	886.02	425.75	393.63	320.54	576.06	-
Tourism Quarterly	9487.13	-	17196.86	15845.10	15014.19	**7656.49**	9972.42	8925.52	10475.47	9902.58	10267.97	8881.04	9511.37	8640.56	9137.12	9521.67	16918.86	9311.98
Tourism Monthly	2615.26	2488.41	2862.06	5636.83	5302.10	4996.60	6940.08	5804.51	6022.21	5536.70	6071.62	5315.74	**1871.69**	2003.02	5998.22	4057.97	5608.81	3145.48
CIF 2016(E+5)	**4.67**	-	5.39	5.78	5.81	7.15	8.56	6.42	4.69	5.63	6.04	14.96	32.00	6.79	59.98	40.58	5.99	6.84
Aus. Elec.	235.49	248.62	**201.39**	659.6	659.6	655.14	370.74	1282.99	1045.92	247.18	241.77	258.76	302.41	213.83	227.5	231.45	760.81	560.48
Bitcoin(E+18)	2.15	1.44	1.87	0.78	5.53	5.53	0.99	1.10	3.62	**0.66**	1.93	1.45	1.95	1.06	2.46	2.61	1.74	8.57
Pedestrian Counts	52.99	44.50	23.17	170.88	170.87	170.96	222.38	216.5	635.16	44.18	43.41	46.41	44.78	66.84	46.46	47.29	97.77	65.92
Vehicle Trips	17.68	-	21.85	31.42	29.98	30.76	**21.21**	30.95	30.97	27.24	22.61	22.93	22	28.16	24.15	28.01	31.48	-
KDD cup	30.87	**29.96**	39.09	42.13	42.04	42.06	39.2	44.88	52.2	36.85	34.82	37.16	48.98	49.1	37.08	44.46	42.72	-
Weather	2.05	**1.79**	1.8	2.36	2.24	2.51	2.3	2.35	2.45	8.17	2.51	2.09	2.02	2.34	2.29	2.03	2.17	2.09
NN5 Daily	**3.57**	3.65	4.26	8.26	6.63	3.8	3.7	3.72	4.41	5.47	4.22	4.06	3.94	4.92	3.97	4.17	7.10	6.67
NN5 Weekly	15.00	**14.28**	16.42	16.71	15.66	15.3	14.98	15.7	15.38	14.94	15.29	15.02	14.69	14.19	19.34	20.34	15.76	15.60
Carparts	0.65	-	0.47	0.65	0.55	0.53	0.58	0.56	0.56	0.41	0.53	**0.39**	**0.39**	0.98	0.4	0.39	0.44	-
FRED-MD	3806.16	2907.86	2679.29	2825.67	2798.22	3492.84	1989.97	2041.92	2957.11	8921.94	2475.68	2339.57	4264.36	2557.8	2508.4	4666.04	2804.64	**1781.41**
Traffic Hourly	0.02	0.02	0.02	0.03	0.03	0.03	0.04	0.03	0.04	0.02	0.02	**0.01**	**0.01**	0.02	0.02	**0.01**	0.03	0.02
Traffic Weekly	1.13	1.13	1.14	1.19	**1.12**	1.13	1.17	1.14	1.22	1.13	1.17	1.15	1.18	1.11	1.2	1.42	1.15	1.15
Rideshare	1.12	**1.11**	1.39	6.29	6.29	7.62	6.45	6.29	3.37	6.3	6.07	6.59	6.28	5.55	2.75	6.29	6.28	-
Hospital	19.08	-	19.4	24.07	21.76	18.54	**17.43**	17.97	19.6	19.24	19.17	22.86	18.25	20.18	19.35	36.19	25.68	22.75
COVID Deaths	174.35	137.76	126.11	353.71	353.71	321.32	96.29	85.59	87.77	347.98	475.15	144.14	201.98	158.81	1049.48	408.66	653.31	**66.14**
Temperature Rain	6.27	5.14	**5.08**	9.39	8.18	8.22	7.14	8.21	7.19	6.13	6.76	5.56	5.37	7.28	5.81	5.24	6.37	-
Sunspot	3.51	0.41	**0.08**	3.93	4.93	4.93	2.57	4.93	2.57	3.83	2.27	7.97	0.77	14.47	0.17	0.13	5.07	0.28
Saugeen	25.41	**21.55**	24.4	21.5	21.5	21.49	22.26	30.69	22.38	25.24	21.28	22.98	23.51	27.92	22.17	28.06	34.84	23.01
US Births	462.16	442.71	614.3	1152.67	1192.20	586.93	**399**	417.93	526.33	574.92	441.7	557.87	422	504.4	452.87	1374.99	638.82	
Aggregated	0.68	-	**0.58**	1.0	1.03	0.96	0.78	0.91	0.93	0.81	0.78	0.77	0.76	0.79	0.77	0.82	1.01	-

Table 14: Delphyne model results of Monash Time Series Forecasting Benchmark. MAE is reported; for fine-tuning, the MAE is taken over 3 experimental runs and we report the mean ± std. The best result is in **bold**. "**Aggregated (Fine-tune)**" means that we take the geometric mean of the MAE of each fine-tuned dataset divided by the MAE of the Naive approach.

Dataset	Delphyne-L-ZS	Delphyne-F-ZS	Delphyne-A-ZS	Delphyne-L-FT	Delphyne-F-FT	Delphyne-A-FT
M1 Monthly	2252.507	2298.433	**2153.370**	-	-	-
M3 Monthly	**643.874**	817.461	649.765	-	-	-
M3 Other	209.257	493.340	**202.444**	-	-	-
M4 Monthly	620.923	697.144	597.434	569.709±7.662	563.145±9.496	**560.472±2.022**
M4 Weekly	340.878	379.246	378.363	**296.282±4.909**	319.959±38.095	306.055±5.195
M4 Daily	210.110	201.094	223.536	**174.349±0.727**	182.327±3.155	181.884±5.742
M4 Hourly	234.718	1369.664	218.845	263.368±16.039	331.591±29.941	**211.930±0.657**
Tourism Quarterly	**9268.650**	17143.935	9487.128	-	-	-
Tourism Monthly	2458.334	6074.311	2615.256	**2360.693±123.850**	2460.906±277.598	2488.407±205.380
CIF 2016 (E+6)	5.408	**2.826**	4.670	-	-	-
Aus. Elec.	203.034	1795.900	235.490	**199.854±0.200**	235.263±5.601	248.615±0.959
Bitcoin (E+18)	1.928	1.871	2.152	1.255±0.174	**1.185±0.861**	1.437±0.332
Pedestrian Counts	44.367	170.285	52.987	**41.917±0.532**	58.494±5.365	44.505±1.217
Vehicle Trips	18.327	20.227	**17.682**	-	-	-
KDD cup	30.045	35.768	30.868	30.029±0.019	**28.822±0.110**	29.964±0.039
Weather	2.053	2.495	2.052	**1.776±0.014**	1.807±0.036	1.792±0.010
NN5 Daily	3.596	3.622	3.574	3.566±0.044	**3.474±0.017**	3.648±0.063
NN5 Weekly	14.899	15.945	14.999	**14.109±0.045**	14.189±0.047	14.280±0.087
Carparts	0.656	0.662	**0.652**	-	-	-
FRED-MD	2868.493	3510.298	3806.159	**2720.094±101.289**	4008.423±198.970	2907.856±200.807
Traffic Hourly	0.018	0.038	0.018	**0.015±0.000**	0.016±0.000	0.017±0.001
Traffic Weekly	1.121	1.163	1.125	**1.108±0.007**	1.123±0.005	1.131±0.010
Rideshare	1.256	1.676	1.123	1.221±0.010	**1.110±0.001**	1.111±0.001
Hospital	**19.029**	22.427	19.081	-	-	-
COVID Deaths	154.175	385.451	174.354	**135.791±21.305**	180.857±23.128	137.716±4.510
Temperature Rain	6.794	7.515	6.267	5.341±0.160	5.482±0.112	**5.142±0.006**
Sunspot	**3.163**	9.458	3.507	0.328±0.035	0.455±0.072	0.410±0.015
Saugeen	24.281	23.757	25.410	24.780±0.088	23.258±0.209	**21.552±0.169**
US Births	443.349	462.390	463.157	**365.175±42.777**	425.495±9.022	442.705±0.801
Aggregated Fine-tune	0.623	0.929	0.626	**0.514**	0.562	0.536

E Financial Tasks

The hyperparameters we tuned are in Table 15. Note, for MOMENT [18], we use patch length of 8 and the context length to 512 since these are fixed by the model. Similarly, for TTM [10], we used a context length of 512 since this is fixed by the model; we also used a head dropout of 0.7 (as suggested in the paper).

Table 15: Hyperparameter search values for financial tasks.

	Hyperparameter	**Values**
Delphyne	learning rate	$\{1e-5, 5e-4, 1e-4\}$
	dropout	$\{0.1, 0.2, 0.3\}$
MOIRAI	learning rate	$\{1e-5, 5e-4, 1e-4\}$
	patch size	$\{16, 32\}$
TTM	learning rate	$\{1e-5, 5e-4, 1e-4, 1e-3\}$
PatchTST	patch size	$\{1, 4, 16, 32\}$
	hidden size	$\{64, 128, 256\}$
	dropout	$\{0.0, 0.1, 0.2\}$

E.1 Calibration Analysis for Stocks Task

In addition to evaluating the models with NLL, we can also evaluate the coverage statistics: **given a forecasted quantile q, what percentage of the observations are less than that value?** In Table 16, we present the results. We see that Delphyne-A-FT performs the best or second-best in nearly all of the quantiles.

Table 16: Results of stock risk analysis for zero-shot versus fine-tuning with stocks coverage statistic. We **bold** the results that are closest in absolute error to the optimal coverage.

Model	Q10	Q25	Q50	Q75	Q90
Optimal	0.100	0.250	0.500	0.750	0.900
Delphyne-A-ZS	0.109	0.239	<u>0.502</u>	<u>0.749</u>	0.892
Delphyne-F-ZS	<u>0.104</u>	**0.252**	0.515	0.777	0.907
Delphyne-L-ZS	**0.097**	0.238	0.525	0.790	0.906
Delphyne-A-FT	0.106	<u>0.246</u>	**0.501**	**0.750**	**0.900**
Delphyne-F-FT	0.086	0.216	0.487	0.768	<u>0.903</u>
Delphyne-L-FT	0.089	0.216	0.510	0.794	0.916
MOIRAI-ZS	0.108	0.215	0.445	0.732	0.877
MOIRAI-FT	0.083	0.205	0.493	0.794	0.917
GARCH	0.111	0.269	0.561	0.790	0.913
PatchTST	0.114	0.264	0.509	0.748	0.896

Table 17: Full results for zero-shot versus fine-tuning for predicting next-day stock squared-returns (variance) data. MSE is reported; for fine-tuning, the MSE is taken over 3 experimental runs and we report the mean $\pm$ std.

Model	MSE ZS	MSE FT
Delphyne-A	37.792	37.810$\pm$0.105
Delphyne-F	<u>37.653</u>	38.616$\pm$1.566
Delphyne-L	**37.591**	38.246$\pm$0.598
MOIRAI	41.428	40.502$\pm$0.046
MOMENT	46.006	37.935$\pm$0.179
TTM	44.918	44.360$\pm$0.004
PatchTST	-	51.705$\pm$11.467
GARCH	41.517	-

Table 18: Full results for zero-shot versus fine-tuning for next-day stock-returns risk analysis. NLL is reported; for fine-tuning, the NLL is taken over 3 experimental runs and we report the mean $\pm$ std.

Model	NLL ZS	NLL FT
Delphyne-A	1.762	**1.741$\pm$0.002**
Delphyne-F	1.750	<u>1.746$\pm$0.001</u>
Delphyne-L	1.775	1.757$\pm$0.005
MOIRAI	1.776	1.788$\pm$0.001
GARCH	1.752	-
PatchTST	-	1.751$\pm$0.005

Table 19: Full results for zero-shot versus fine-tuning for predicting bars log-volume data (longer horizon, 78 timesteps prediction for 5-minute intervals). MSE is reported; for fine-tuning, the MSE is taken over 3 experimental runs and we report the mean$\pm$std.

Model	MSE ZS	MSE FT
Delphyne-A	0.728	0.551$\pm$0.017
Delphyne-F	0.965	**0.530$\pm$0.01**
Delphyne-L	0.930	0.557$\pm$0.002
MOIRAI	0.765	0.621$\pm$0.003
MOMENT	0.775	0.838$\pm$0.028
TTM	0.714	0.600$\pm$0.001
PatchTST	-	<u>0.534</u>
Last Value	0.602	-

Table 20: Nowcasting results for zero-shot vs. fine-tuning for company sales growth data. MAE is reported; for fine-tuning, the MAE is taken over 3 experimental runs and we report the mean ± std.

Model	MAE ZS	MAE FT
Delphyne-A	0.099	**0.071±0.002**
Delphyne-F	0.128	0.079±0.003
Delphyne-L	0.101	0.073±0.001
MOIRAI	0.091	0.093±0.001
Baseline	0.100	-

E.2 R^2 Measurement

For utilizing the R^2 measure for finance time series, we have provided the R^2 for next-day stock squared returns (variance) prediction and bars log-volume data predictions, as shown in Table 21 and Table 22 below. Delphyne out-performs existing benchmarks for both ML approaches, time-series deep models and traditional AR approaches (GARCH).

Table 21: R^2 for next-day stock squared returns (↑)

Model	R^2 (Zero-shot)	R^2 (Finetune)
Delphine-A	0.0057	0.0052
Delphine-F	0.0093	0.0102
Delphine-L	**0.0110**	0.0086
MOIRAI	-0.0900	-0.0656
MOMENT	-0.2104	0.0019
TTM	-0.1818	-0.1671
PatchTST	-	-0.3604
GARCH	-0.0923	-

Table 22: R^2 for log-volume data (↑)

Model	R^2 (Zero-shot)	R^2 (Finetune)
Delphine-A	0.0582	0.2872
Delphine-F	-0.2484	**0.3144**
Delphine-L	-0.2031	0.2794
MOIRAI	0.0103	0.1966
MOMENT	-0.0026	-0.0841
TTM	0.0763	0.2238
PatchTST	-	0.3092
Avg. past values	0.2212	-

E.3 Comparison to Factor Model

Since many benchmarked time-series models can't handle multi-variate or nowcasting style features, we designed our financial tasks mostly as univariate prediction tasks so that we can make a fair comparison with other models. But the Delphyne model is designed keeping in mind all these nuances of financial tasks and thus it can easily incorporate diverse features arriving at different frequencies including contemporaneous features along with the target time-series which makes it suitable for comparison with other financial models like GARCH, Fama French factor model, etc.

To provide a comparison with factor models, we have tested our model against the 3 factor Fama french model where the risk adjusted stock returns are predicted as a function of the market, SMB (small minus big factor) and HML (high minus low factor) using OLS regression. For a fair comparison, we added the factor data as features along with the past values of stock's squared returns as input to Delphyne and then compared the predicted squared stock returns in terms of MSE and R^2 statistic.

Table 23 below provides results for the same where finetuned Delphyne model is able to outperform factor model.

Table 23: Comparison to Fama–French 3 Factor Model

Factor Model	R^2 Stocks Squared Returns	MSE Stocks Squared Returns
Factor Model	0.026	37.073
Delphine-ZS	0.0084	37.687
Delphine-A-FT	0.0189	37.290
Delphine-F-FT	0.0176	37.340
Delphine-L-FT	**0.0315**	**36.812**

F Short-term Forecasting on Monash Dataset

We conduct an evaluation using the Monash dataset [15], which spans multiple domains like demand forecasting, traffic, and weather, with various data granularities. We follow the train-test split outlined in [47], evaluating performance only on the hold-out test set to ensure a fair in-distribution comparison. We report both zero-shot and finetuning results in Table 13 and comparison across versions of Delphyne in Table 14. Fig. 4 and Fig. 5 provide the aggregate geometric mean of normalized MAE.

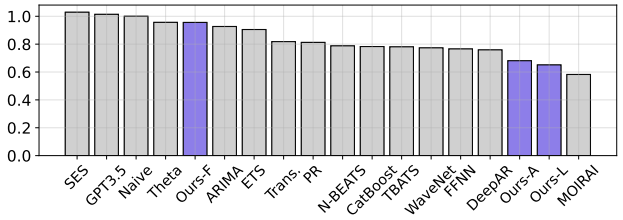

Figure 4: Aggregated geometric mean of normalized MAEs on the Monash Time-Series Forecasting Benchmark. On average, Delphyne zero-shot models perform better than existing models, falling into second place behind MOIRAI.

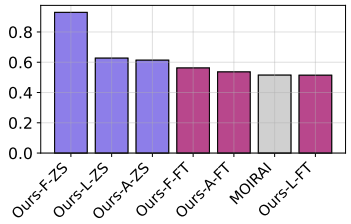

Figure 5: Aggregated geometric mean of normalized MAEs for Delphyne zero-shot (ZS) and fine-tuned(FT) models.

G Long-term Forecasting Experiment

For long-term forecasting, we evaluate Delphyne and benchmarks on popular datasets with forecast lengths of $96, 192, 336$, and 720. We report the average performance across various forecast lengths for zero-shot, linear probing, fine-tuning, and full-shot methods in Table 24, with the full results in Table 25. Delphyne-A demonstrates strong performance after fine-tuning, a crucial step for improving out-of-distribution forecasting and overcoming negative transfer effect. Its architectural design and training paradigm make it particularly adaptable.

Table 24: Average MAE and MSE across forecast lengths $\{96, 192, 336, 720\}$ for Delphyne and other baseline methods. The best are highlighted in **bold** and the runner-up is underlined.

		Zero-shot					Linear-Probing			Fine-tuned			Full-Shot				
		TimeMOE-ZS	TTM	MOIRAI-ZS	TimesFM	Delphyne-A-ZS	TimeLLM	MOMENT	GPT4TS	Delphyne-A-FT	MOIRAI-FT	TimeMOE-FT	PatchTST	Dlinear	TimesNet	FEDFormer	Stationary
ETTh1	MSE	0.394	0.402	0.434	0.476	0.449	0.408	0.418	0.428	0.440	0.675	**0.375**	0.413	0.423	0.457	0.440	0.570
	MAE	0.419	-	0.439	0.451	0.450	0.423	0.436	0.426	0.441	0.658	**0.404**	0.431	0.437	0.449	0.460	0.537
ETTh2	MSE	0.405	**0.327**	0.346	0.404	0.375	0.334	0.352	0.355	0.352	0.387	0.361	0.330	0.431	0.414	0.437	0.526
	MAE	0.415	-	0.382	0.406	0.404	0.383	0.395	0.395	**0.356**	0.412	0.386	0.379	0.447	0.427	0.449	0.516
ETTm1	MSE	0.376	0.338	0.382	0.420	0.501	0.329	0.354	0.352	0.364	0.389	**0.322**	0.351	0.357	0.400	0.448	0.481
	MAE	0.405	-	0.388	0.408	0.429	0.372	0.391	0.383	**0.365**	0.395	0.371	0.381	0.379	0.406	0.452	0.456
ETTm2	MSE	0.316	0.264	0.272	0.350	0.323	0.251	0.256	0.266	**0.250**	0.253	0.284	0.255	0.267	0.291	0.305	0.306
	MAE	0.361	-	0.321	0.353	0.363	0.313	0.270	0.326	**0.257**	0.275	0.332	0.315	0.334	0.333	0.350	0.347
ECL	MSE	-	0.160	0.188	**0.156**	0.202	0.158	0.165	0.167	0.170	0.203	-	0.162	0.166	0.193	0.214	0.193
	MAE	-	-	0.274	0.246	0.293	0.252	0.260	0.263	**0.203**	0.212	-	0.253	0.264	0.295	0.327	0.296
Weather	MSE	0.270	0.233	0.235	0.232	0.369	0.225	0.230	0.237	**0.221**	0.287	0.234	0.226	0.249	0.259	0.309	0.288
	MAE	0.300	-	0.263	0.257	0.348	0.257	0.261	0.271	**0.235**	0.258	0.273	0.264	0.300	0.286	0.360	0.314

For fine-tuning Delphyne, we use a learning rate of 5e-5, dropout of 0.2, batch size of 64, and a linear warmup for the learning rate of 50 steps. For all datasets, we use a context length of 1000. We use early-stopping based on the validation loss. Due to Electricity and Weather being large datasets, we

randomly sample 32×500 rows from the validation set for early-stopping. For long-term forecasting experiment, we do not conduct any additional hyperparameter searching, although that could lead to improved performances.

G.1 Comparison Methods

Zero-Shot Methods. For zero-shot methods, we report TTM_A, the best and largest model presented in [10]. Since the authors have only published models with a forecast length of 96, we are limited to reporting the MSE based on their reported results. For MOIRAI [47], we again report the performance of $\text{MOIRAI}_\text{Base}$, which has similar number of parameters as our model. For TimesFM [8], we follow their demonstration [1] and report the MSE and MAE results for their checkpoint "google/timesfm-1.0-200m" in Huggingface. For Time-MOE [38], we take the numbers from Time-MOE large model. While Time-MOE is one of the best models, it utilizes a mixture of experts and model size is significantly larger than Delphyne.

Linear-Probing. We directly report the linear probing results from MOMENT's experiments [18], which include baseline results for GPT4TS [52]. For Time-LLM [24], we also take the results from the paper.

Fine-tuning. we also finetune MOIRAI with the same procedure as our Delphyne model and we report the MAE, MSE for Delphyne and other baseline methods for long-term performance. We search learning rate in $\{1e-1, 1e-3, 5e-5, 1e-5\}$, linear warmup in $\{0, 50\}$ dropout in $\{0, 0.2, 0.4\}$, and various weight decays. However, we do not see an improvement in fine tuning performance for MOIRAI. This validates our hypothesis that Delphyne is better at adapting to new tasks quickly with few gradient updates.

Full-Shot. The full-shot results are obtained from [18]. Within the full-shot results, PatchTST [30], DLinear [11], TimesNet [48], FEDFormer [51], Stationary [52], LightTS [6] and N-BEATS [31] are reported.

G.2 Full Comparison Results

Table 25 shows an comparison across different models and Table 26 shows the comparison across different versions of Delphyne.

Table 25: Zero-shot and Full-shot Results for Delphyne and Other Models

Dataset	Horizon	Zero-shot								Linear-Probing						Fine-tuned		Full-shot															
		TTM		MOIRAI		TimesFM		Delphyne-A-ZS		MOMENT		Time-LLM		GPT4TS		Delphyne-A-FT		PatchTST		DLinear		TimesNet		FEDFormer		Stationary		LightTS		N-BEATS	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.359	-	0.384	0.402	0.405	0.432	0.398	0.417	0.387	0.410	0.408	0.429	0.376	0.397	0.376	0.390	0.370	0.399	0.375	0.399	0.384	0.402	0.376	0.419	0.513	0.491	0.424	0.432	0.399	0.428
	192	0.389	-	0.425	0.429	0.459	0.432	0.440	0.444	0.410	0.426	-	-	0.416	0.418	0.432	0.440	0.413	0.421	0.405	0.416	0.436	0.429	0.420	0.448	0.534	0.504	0.475	0.462	0.451	0.464
	336	0.405	-	0.456	0.450	0.534	0.458	0.474	0.464	0.422	0.437	-	-	0.442	0.433	0.452	0.481	0.422	0.436	0.439	0.443	0.491	0.469	0.459	0.465	0.588	0.535	0.518	0.488	0.498	0.500
	720	0.448	-	0.470	0.473	0.513	0.482	0.482	0.479	0.454	0.472	0.523	0.514	0.477	0.456	0.498	0.454	0.447	0.466	0.472	0.490	0.521	0.500	0.506	0.507	0.643	0.616	0.547	0.533	0.608	0.573
	Avg	0.402	-	0.434	0.439	0.476	0.451	0.449	0.450	0.418	0.436	-	-	0.428	0.426	0.440	0.441	0.413	0.431	0.423	0.437	0.457	0.449	0.440	0.460	0.570	0.537	0.491	0.479	0.489	0.491
ETTh2	96	0.264	-	0.277	0.327	0.311	0.344	0.303	0.352	0.288	0.345	0.285	0.348	0.285	0.342	0.281	0.300	0.274	0.336	0.289	0.353	0.340	0.374	0.358	0.397	0.476	0.458	0.397	0.437	0.327	0.387
	192	0.321	-	0.340	0.374	0.395	0.393	0.371	0.397	0.349	0.386	-	-	0.354	0.389	0.342	0.351	0.339	0.379	0.383	0.418	0.402	0.414	0.429	0.439	0.512	0.493	0.520	0.504	0.400	0.435
	336	0.351	-	0.371	0.402	0.384	0.402	0.400	0.420	0.369	0.408	-	-	0.373	0.407	0.382	0.393	0.329	0.380	0.448	0.465	0.452	0.452	0.496	0.487	0.552	0.551	0.626	0.559	0.747	0.599
	720	0.395	-	0.394	0.426	0.526	0.484	0.428	0.446	0.403	0.439	-	-	0.406	0.441	0.401	0.381	0.379	0.422	0.605	0.551	0.462	0.468	0.463	0.474	0.562	0.560	0.863	0.672	1.454	0.847
	Avg	0.327	-	0.346	0.382	0.404	0.406	0.375	0.404	0.352	0.395	-	-	0.355	0.395	0.352	0.356	0.330	0.379	0.431	0.447	0.414	0.427	0.437	0.449	0.526	0.516	0.602	0.543	0.732	0.568
ETTm1	96	0.318	-	0.335	0.360	0.332	0.351	0.439	0.399	0.293	0.349	0.384	0.403	0.292	0.346	0.318	0.320	0.290	0.342	0.299	0.343	0.338	0.375	0.379	0.419	0.386	0.398	0.374	0.400	0.318	0.367
	192	0.354	-	0.366	0.379	0.387	0.389	0.483	0.425	0.326	0.368	-	-	0.332	0.372	0.339	0.354	0.332	0.369	0.335	0.365	0.374	0.387	0.426	0.441	0.459	0.444	0.400	0.407	0.355	0.391
	336	0.376	-	0.391	0.394	0.427	0.420	0.512	0.445	0.352	0.384	-	-	0.366	0.394	0.377	0.384	0.366	0.392	0.369	0.386	0.410	0.411	0.445	0.459	0.495	0.464	0.438	0.438	0.401	0.419
	720	0.398	-	0.434	0.419	0.532	0.472	0.572	0.445	0.445	0.462	0.437	0.429	0.417	0.421	0.421	0.402	0.416	0.420	0.425	0.421	0.478	0.450	0.543	0.490	0.585	0.516	0.527	0.502	0.448	0.448
	Avg	0.338	-	0.382	0.388	0.420	0.408	0.501	0.429	0.354	0.391	-	-	0.352	0.383	0.364	0.365	0.351	0.381	0.357	0.379	0.400	0.406	0.448	0.452	0.481	0.456	0.435	0.437	0.381	0.406
ETTm2	96	0.169	-	0.195	0.269	0.189	0.257	0.211	0.294	0.243	0.170	0.181	0.269	0.173	0.262	0.159	0.179	0.165	0.255	0.167	0.269	0.187	0.267	0.203	0.287	0.192	0.274	0.209	0.308	0.197	0.271
	192	0.223	-	0.247	0.303	0.284	0.314	0.278	0.338	0.279	0.285	-	-	0.229	0.301	0.216	0.233	0.220	0.292	0.224	0.303	0.249	0.309	0.269	0.328	0.280	0.339	0.311	0.382	0.285	0.328
	336	0.276	-	0.291	0.333	0.467	0.396	0.343	0.377	0.227	0.297	-	-	0.286	0.341	0.271	0.298	0.274	0.329	0.281	0.342	0.321	0.351	0.325	0.366	0.334	0.361	0.442	0.466	0.338	0.366
	720	0.342	-	0.355	0.377	0.460	0.445	0.459	0.441	0.275	0.328	0.366	0.388	0.378	0.401	0.352	0.317	0.362	0.385	0.397	0.421	0.408	0.403	0.421	0.415	0.417	0.413	0.675	0.587	0.395	0.419
	Avg	0.264	-	0.272	0.321	0.350	0.353	0.323	0.363	0.256	0.270	-	-	0.266	0.326	0.250	0.257	0.255	0.315	0.267	0.334	0.291	0.333	0.305	0.350	0.306	0.347	0.409	0.436	0.304	0.347
Electricity	96	0.152	-	0.158	0.248	0.116	0.210	0.164	0.260	0.136	0.233	-	-	0.139	0.238	0.143	0.176	0.129	0.222	0.140	0.237	0.168	0.272	0.193	0.308	0.169	0.273	0.207	0.307	0.131	0.228
	192	0.179	-	0.174	0.263	0.138	0.229	0.181	0.276	0.152	0.247	-	-	0.153	0.251	0.159	0.192	0.157	0.240	0.153	0.249	0.184	0.289	0.201	0.315	0.182	0.286	0.213	0.316	0.153	0.248
	336	0.193	-	0.191	0.278	0.157	0.251	0.202	0.296	0.167	0.264	-	-	0.169	0.266	0.173	0.207	0.163	0.259	0.169	0.267	0.198	0.300	0.214	0.329	0.200	0.304	0.230	0.333	0.170	0.267
	720	0.243	-	0.229	0.307	0.211	0.292	0.260	0.340	0.205	0.295	-	-	0.206	0.297	0.206	0.238	0.197	0.290	0.203	0.301	0.220	0.320	0.246	0.355	0.222	0.321	0.265	0.360	0.208	0.298
	Avg	0.160	-	0.188	0.274	0.156	0.246	0.202	0.293	0.165	0.260	-	-	0.167	0.263	0.170	0.203	0.162	0.253	0.166	0.264	0.193	0.295	0.214	0.327	0.193	0.296	0.229	0.329	0.166	0.260
Weather	96	0.159	-	0.167	0.203	0.117	0.156	0.188	0.248	0.154	0.209	-	-	0.162	0.212	0.140	0.157	0.149	0.198	0.176	0.237	0.172	0.220	0.217	0.296	0.173	0.223	0.182	0.242	0.152	0.210
	192	0.203	-	0.197	0.248	0.165	0.205	0.265	0.309	0.209	0.214	-	-	0.204	0.248	0.187	0.204	0.194	0.241	0.220	0.282	0.219	0.261	0.276	0.336	0.245	0.285	0.227	0.287	0.199	0.260
	336	0.247	-	0.256	0.276	0.256	0.276	0.409	0.374	0.246	0.285	-	-	0.254	0.286	0.242	0.255	0.245	0.282	0.265	0.319	0.280	0.306	0.339	0.380	0.321	0.338	0.282	0.334	0.258	0.311
	720	0.314	-	0.321	0.323	0.388	0.392	0.612	0.460	0.315	0.336	-	-	0.326	0.337	0.316	0.323	0.314	0.334	0.333	0.362	0.365	0.359	0.403	0.428	0.414	0.410	0.352	0.386	0.331	0.359
	Avg	0.233	-	0.235	0.263	0.232	0.257	0.369	0.348	0.230	0.261	-	-	0.237	0.271	0.221	0.235	0.226	0.264	0.249	0.300	0.259	0.286	0.309	0.360	0.288	0.314	0.261	0.312	0.235	0.286

[1] We use the following script and set different forecast lengths in `https://github.com/google-research/timesfm/blob/master/notebooks/finetuning.ipynb`.

Table 26: Full results of long sequence forecasting experiments for zero-shot versus fine-tuning. Delphyne-F underperforms both model due to lack of similar dataset in pre-training. Both Delphyne-A and Delphyne-L perform comparatively well after fine-tuning.

Dataset		Delphyne-L-**ZS**		Delphyne-F-**ZS**		Delphyne-A-**ZS**		Delphyne-L-**FT**		Delphyne-F-**FT**		Delphyne-A-**FT**	
		MSE	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
ETTh1	96	0.388	0.418	1.179	0.690	0.398	0.417	0.384±0.001	0.400±0.023	0.403±0.003	0.409±0.011	**0.376±0.001**	**0.390±0.020**
	192	0.437	0.452	1.405	0.769	0.440	0.444	0.445±0.009	0.457±0.010	0.456±0.023	0.493±0.032	**0.432±0.011**	**0.440±0.006**
	336	0.502	0.483	1.677	0.857	0.474	**0.464**	0.470±0.001	0.490±0.029	0.487±0.040	0.484±0.036	**0.452±0.006**	0.481±0.035
	720	0.598	0.526	2.203	1.025	**0.482**	0.479	0.534±0.013	0.489±0.067	0.517±0.013	0.484±0.042	0.498±0.023	**0.454±0.039**
ETTh2	96	0.297	0.354	0.440	0.433	0.303	0.352	0.283±0.005	0.302±0.032	0.328±0.013	0.351±0.045	**0.281±0.006**	**0.300±0.031**
	192	0.368	0.405	0.612	0.513	0.371	0.397	0.352±0.007	0.364±0.014	0.402±0.014	0.393±0.012	**0.342±0.001**	**0.351±0.014**
	336	0.412	0.434	0.799	0.592	0.400	0.420	0.383±0.002	**0.389±0.008**	0.406±0.017	0.420±0.012	**0.382±0.009**	0.393±0.007
	720	0.448	0.462	1.075	0.689	0.428	0.446	0.422±0.015	0.403±0.043	0.433±0.011	0.417±0.028	**0.401±0.004**	**0.381±0.028**
ETTm1	96	0.599	0.455	1.498	0.711	0.439	0.399	**0.295±0.004**	**0.314±0.028**	0.313±0.001	0.324±0.014	0.318±0.009	0.320±0.013
	192	0.715	0.502	1.730	0.780	0.483	0.425	0.348±0.006	**0.350±0.007**	0.342±0.005	0.350±0.014	**0.339±0.005**	0.354±0.021
	336	0.795	0.534	2.006	0.861	0.512	0.445	**0.363±0.004**	**0.382±0.024**	0.385±0.014	0.417±0.035	0.377±0.008	0.384±0.016
	720	0.919	0.581	3.030	1.090	0.572	0.478	0.433±0.014	0.408±0.048	0.482±0.030	0.446±0.069	**0.421±0.012**	**0.402±0.039**
ETTm2	96	0.245	0.313	0.288	0.347	0.211	0.294	**0.157±0.002**	0.179±0.028	0.159±0.002	0.181±0.029	0.159±0.003	**0.179±0.024**
	192	0.351	0.375	0.422	0.418	0.278	0.338	0.221±0.002	0.239±0.024	0.222±0.002	0.246±0.033	**0.216±0.002**	**0.233±0.022**
	336	0.452	0.427	0.583	0.495	0.343	0.377	0.273±0.000	0.305±0.046	0.283±0.007	0.312±0.048	**0.271±0.006**	**0.298±0.035**
	720	0.554	0.480	0.918	0.625	0.459	0.441	0.360±0.007	0.318±0.052	0.373±0.010	0.328±0.059	**0.352±0.005**	**0.317±0.053**
Weather	96	0.197	0.256	0.315	0.295	0.188	0.248	0.141±0.001	0.158±0.026	0.147±0.002	0.165±0.024	**0.140±0.002**	**0.157±0.023**
	192	0.292	0.324	0.416	0.353	0.265	0.309	0.189±0.003	0.207±0.023	0.191±0.002	0.211±0.031	**0.187±0.003**	**0.204±0.026**
	336	0.433	0.390	0.576	0.426	0.409	0.374	**0.240±0.002**	**0.252±0.018**	0.246±0.006	0.260±0.025	0.242±0.004	0.255±0.021
	720	0.633	0.483	1.035	0.576	0.612	0.460	**0.311±0.008**	**0.319±0.012**	0.322±0.006	0.332±0.009	0.316±0.008	0.323±0.018
Electricity	96	0.161	0.259	2.006	1.077	0.164	0.260	0.145±0.002	0.178±0.046	0.164±0.010	0.198±0.041	**0.143±0.001**	**0.176±0.046**
	192	0.180	0.275	2.399	1.160	0.181	0.276	**0.159±0.002**	**0.191±0.047**	0.173±0.005	0.205±0.049	**0.159±0.002**	0.192±0.045
	336	0.203	0.295	3.151	1.304	0.202	0.296	**0.172±0.002**	**0.204±0.044**	0.191±0.003	0.222±0.046	0.173±0.001	0.207±0.047
	720	0.268	0.342	4.664	1.592	0.260	0.340	**0.203±0.003**	**0.235±0.044**	0.233±0.001	0.264±0.044	0.206±0.002	0.238±0.046

H Probability Quantification

We assess probability quantification on six diverse datasets. We report the Continuous Ranked Probability Score (CRPS) and Mean Scaled Interval Score (MSIS) metrics in Table 27, and additional deterministic metrics are shown in Table 28. We use a rolling evaluation setup where the stride matches the forecast length.

For fine-tuning Delphyne, we use a learning rate of 5e-5, dropout of 0.2, batch size of 128, and a linear warmup for the learning rate of 50 steps. For all datasets, we use a context length of 1000 except Walmart, for which we use 50-100. We use early-stopping based on the validation loss. Similarly, we stick to the default hyperparameters without additional searching.

For evaluation, we use CRPS [14], MSIS [27], symmetric mean absolute percentage error (sMAPE) [22], mean absolute scaled error (MASE) [23], normalized deviation (ND), and normalized root mean squared error (NRMSE) [50].

The CRPS [14] is a probabilistic forecasting evaluation metric, given a forecasted distribution with c.d.f. F and ground truth y, it is defined as:

$$\text{CRPS} = \int_0^1 2\Lambda_\alpha(F^{-1}(\alpha), y)d\alpha$$
$$\Lambda_\alpha(q, y) = (\alpha - \mathbf{1}_{\text{y<q}})(y - q),$$

where Λ_α is the α-quantile loss, also known as the pinball loss at quantile level α. To compute a normalized metric, the mean weighted sum quantile loss [32], defined as the average of K quantiles:

$$\text{CRPS} \approx \frac{1}{K}\sum_{k=1}^{K} \text{wQL}[\alpha_k]$$
$$\text{wQL}[\alpha] = 2\frac{\sum_t \Lambda_\alpha(\hat{q}_t(\alpha), y_t)}{\sum_t |y_t|},$$

where $\hat{q}_t(\alpha)$ is the forecasted α-quantile at time step t. We take $K = 9, \alpha_1 = 0.1, \alpha_2 = 0.2, \ldots, \alpha_9 = 0.9$ in practice.

The MSIS [27] is a metric to evaluate uncertainty around point forecasts. Given an upper bound forecast U_t (0.975 quantile) and lower bound forecast L_t (0.025 quantile) the MSIS is defined as:

$$\text{MSIS} = \frac{1}{h}\frac{\sum_{t=1}^{h}(U_t - L_t) + \frac{2}{a}(L_t - Y_t)\mathbb{I}_{\{Y_t<L_t\}} + \frac{2}{a}(Y_t - U_t)\mathbb{I}_{\{Y_t>U_t\}}}{\frac{1}{n-m}\sum_{t=m+1}^{n}|Y_t - Y_{t-m}|}$$

where $a = 0.05$ is the significance level for a 95% forecast interval, over a forecast horizon of length h, and m is the seasonal factor.

Table 27: Full results for probabilistic forecasting experiments. The best results are highlighted in **bold**, and the second best results are underlined. (The baseline results are taken from [47].)

		Zero-shot		Finetuned	Full-shot				Baseline	
		Delphyne-A-ZS	MOIRAI	Delphyne-A-FT	PatchTST	TiDE	TFT	DeepAR	AutoARIMA	Seasonal Naive
Electricity	**CRPS**	0.159	0.055	0.140±0.005	0.052±0.00	**0.048±0.00**	0.050±0.00	0.065±0.01	0.327	0.070
	MSIS	29.293	6.172	21.820±1.383	5.744±0.12	**5.672±0.08**	6.278±0.24	6.893±0.82	29.412	35.251
Solar	**CRPS**	0.905	0.419	1.306±0.103	0.518±0.09	**0.420±0.00**	0.446±0.03	0.431±0.01	1.055	0.512
	MSIS	2.733	7.011	**2.029±0.520**	8.447±1.59	13.754±0.32	8.057±3.51	11.181±0.67	25.849	48.130
Walmart	**CRPS**	0.093	0.093	0.083±0.001	0.082±0.01	**0.077±0.00**	0.087±0.00	0.121±0.00	0.124	0.151
	MSIS	4.741	8.421	**4.559±0.289**	6.005±0.21	6.258±0.12	8.718±0.10	12.502±0.03	9.888	49.458
Weather	**CRPS**	0.064	**0.041**	0.042±0.005	0.059±0.01	0.054±0.00	0.043±0.00	0.132±0.01	0.252	0.068
	MSIS	6.080	5.136	**4.467±0.053**	7.759±0.49	8.095±1.74	7.791±0.44	21.651±17.34	19.805	31.293
Istanbul Traffic	**CRPS**	0.149	0.116	0.212±0.015	0.112±0.00	0.110±0.01	0.110±0.01	**0.108±0.00**	0.589	0.257
	MSIS	9.989	4.461	4.328±0.536	**3.813±0.09**	4.752±0.17	4.057±0.44	4.094±0.31	16.317	45.473
Turkey Power	**CRPS**	0.046	0.040	**0.035±0.001**	0.054±0.01	0.046±0.01	0.039±0.00	0.066±0.02	0.116	0.085
	MSIS	6.269	6.766	**5.384±0.346**	8.978±0.51	8.579±0.52	7.943±0.31	13.520±1.17	14.863	36.256

Table 28: Full results for probabilistic forecasting experiments. The best results are highlighted in **bold**, and the second best results are underlined. (The baseline results are taken from [47].)

		Zero-shot		Finetuned	Full-shot				Baseline	
		Delphyne-A-ZS	MOIRAI	Delphyne-A-FT	PatchTST	TiDE	TFT	DeepAR	AutoARIMA	Seasonal Naive
Electricity	**CRPS**	0.159	0.055	0.140±0.005	0.052±0.00	**0.048±0.00**	0.050±0.00	0.065±0.01	0.327	0.070
	MSIS	29.293	6.172	21.820±1.383	5.744±0.12	**5.672±0.08**	6.278±0.24	6.893±0.82	29.412	35.251
	sMAPE	0.233	0.111	0.215±0.008	0.107±0.00	**0.102±0.00**	0.106±0.01	0.118±0.02	0.318	0.108
	MASE	2.031	0.792	1.839±0.080	0.753±0.01	0.706±0.02	0.747±0.03	0.844±0.16	3.229	0.881
	ND	0.182	0.069	0.164±0.005	0.065±0.00	**0.061±0.00**	0.063±0.00	0.080±0.02	0.357	0.070
	NRMSE	1.084	0.551	0.986±0.007	0.506±0.02	0.514±0.02	0.511±0.02	0.704±0.11	3.296	**0.478**
Solar	**CRPS**	0.905	0.419	1.306±0.103	0.518±0.09	**0.420±0.00**	0.446±0.03	0.431±0.01	1.055	0.512
	MSIS	2.733	7.011	**2.029±0.520**	8.447±1.59	13.754±0.32	8.057±3.51	11.181±0.67	25.849	48.130
	sMAPE	1.650	1.410	1.499±0.009	1.501±0.10	1.400±0.00	1.391±0.01	1.385±0.00	1.685	**0.691**
	MASE	1.710	1.292	**1.076±0.164**	1.607±0.25	1.265±0.02	1.399±0.11	1.222±0.01	2.583	1.203
	ND	**0.263**	0.551	0.290±0.028	0.685±0.11	0.538±0.01	0.594±0.05	0.520±0.00	1.098	0.512
	NRMSE	2.483	1.034	1.060±0.129	1.408±0.26	1.093±0.00	1.236±0.06	**1.033±0.01**	1.784	1.168
Walmart	**CRPS**	0.093	0.093	0.083±0.001	0.082±0.01	**0.077±0.00**	0.087±0.00	0.121±0.00	0.124	0.151
	MSIS	4.741	8.421	**4.559±0.289**	6.005±0.21	6.258±0.12	8.718±0.10	12.502±0.03	9.888	49.458
	sMAPE	0.184	0.168	**0.088±0.003**	0.150±0.01	0.145±0.00	0.172±0.00	0.216±0.00	0.219	0.205
	MASE	**0.645**	0.964	0.660±0.002	0.867±0.09	0.814±0.01	0.948±0.02	1.193±0.02	1.131	1.236
	ND	0.126	0.117	0.101±0.001	0.105±0.01	**0.097±0.00**	0.108±0.00	0.147±0.00	0.141	0.151
	NRMSE	0.270	0.291	0.279±0.003	0.218±0.02	**0.204±0.00**	0.235±0.01	0.298±0.00	0.305	0.328
Weather	**CRPS**	0.064	**0.041**	0.042±0.005	0.059±0.01	0.054±0.00	0.043±0.00	0.132±0.01	0.252	0.068
	MSIS	6.080	5.136	**4.467±0.053**	7.759±0.49	8.095±1.74	7.791±0.44	21.651±17.34	19.805	31.293
	sMAPE	0.906	0.623	0.890±0.214	0.668±0.01	0.636±0.01	0.672±0.01	0.776±0.05	0.770	**0.401**
	MASE	0.701	**0.487**	0.505±0.058	0.844±0.19	0.832±0.13	0.692±0.02	3.170±3.47	0.938	0.782
	ND	0.095	**0.048**	0.057±0.012	0.072±0.02	0.066±0.01	0.051±0.00	0.163±0.15	0.139	0.068
	NRMSE	0.270	0.417	0.212±0.015	0.260±0.01	0.214±0.00	0.211±0.00	0.486±0.43	0.465	0.290
Istanbul Traffic	**CRPS**	0.149	0.116	0.212±0.015	0.112±0.00	0.110±0.01	0.110±0.01	**0.108±0.00**	0.589	0.257
	MSIS	9.989	4.461	4.328±0.536	**3.813±0.09**	4.752±0.17	4.057±0.44	4.094±0.31	16.317	45.473
	sMAPE	0.352	0.284	**0.242±0.017**	0.287±0.01	0.280±0.01	0.287±0.01	0.249±0.01	1.141	0.391
	MASE	0.772	0.644	**0.558±0.018**	0.653±0.02	0.618±0.03	0.620±0.03	0.613±0.03	3.358	1.137
	ND	0.175	0.146	**0.127±0.004**	0.148±0.01	0.140±0.01	0.141±0.01	0.139±0.01	0.758	0.257
	NRMSE	0.273	0.194	0.189±0.010	0.190±0.01	0.185±0.01	0.185±0.01	**0.181±0.01**	0.959	0.384
Turkey Power	**CRPS**	0.046	0.040	**0.035±0.001**	0.054±0.01	0.046±0.01	0.039±0.00	0.066±0.02	0.116	0.085
	MSIS	6.269	6.766	**5.384±0.346**	8.978±0.51	8.579±0.52	7.943±0.31	13.520±1.17	14.863	36.256
	sMAPE	0.176	0.378	0.168±0.002	0.416±0.01	0.389±0.00	0.383±0.00	0.404±0.01	0.244	**0.125**
	MASE	0.891	0.888	**0.790±0.018**	1.234±0.12	0.904±0.02	0.890±0.05	1.395±0.30	1.700	0.906
	ND	0.059	0.051	**0.045±0.001**	0.071±0.01	0.059±0.01	0.049±0.00	0.083±0.02	0.150	0.085
	NRMSE	0.132	0.118	**0.098±0.003**	0.158±0.01	0.139±0.03	0.104±0.01	0.181±0.05	0.383	0.231

I Anomaly Detection

We measure adjusted F1 score for the anomaly detection task, on 44 time-series datasets for the UCR anomaly detection archive, in comparison to popular full-shot models and foundation model MOMENT with anomaly detection head [18]. The aggregated F1 score is provided in Fig. 6. Delphyne-A's versatility allows it to adapt well to anomaly detection tasks after fine-tuning, achieving second place overall in anomaly detection tasks.

I.1 Anomaly Detection Experiment Setup

Our experimental setup is similar to that of Goswami et al. [18]. Following Goswami et al. [17], we used a fixed anomaly detection window size of 512 and downsampled all time-series datasets longer than 2560 timesteps by a factor of 10 to speed up the training and evaluation process. We use the mean squared error between forecasts and observations as the anomaly criterion. We get forecasts

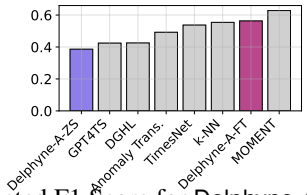

Figure 6: Aggregated Adjusted F1 Score for Delphyne-A vs. comparison baselines.

from our model by masking out nonoverlapping patches of 32 from the window of 512. Even though Delphyne was pre-trained to forecast, we noticed that Delphyne was able impute values in other parts of each time series. For training, to improve imputation performance, each row of the dataset is to forecast a random patches of size 32, not just at the end of the time series.

I.2 Full Comparison Results

Table 29 shows the full adjusted F1 results across UCR Anomaly Archive.

Table 29: Anomaly detection performance measured using adj. best F_1 for a subset of 45 datasets sampled from the UCR Anomaly archive.

Model name	Anomaly Transformer	MOMENT	DGHL	GPT4TS	TimesNet	AnomalyTransformer	Delphyne-A-ZS	Delphyne-A-FT
InternalBleeding4	NaN	NaN	NaN	NaN	NaN	NaN	0.717	0.996
1sddb40	0.030	0.540	0.390	0.190	0.680	0.640	0.818	0.754
BIDMC1	0.990	1.000	1.000	1.000	1.000	0.690	0.390	0.899
CHARISfive	0.010	0.130	0.020	0.020	0.080	0.360	0.017	0.015
CHARISten	0.020	0.110	0.040	0.100	0.030	0.430	0.034	0.040
CIMIS44AirTemperature3	0.060	0.980	0.500	0.180	0.470	0.640	0.167	1.000
CIMIS44AirTemperature5	0.390	0.990	0.960	0.200	0.710	0.780	0.225	1.000
ECG2	1.000	1.000	0.620	0.900	1.000	0.830	0.864	0.772
ECG3	0.360	0.980	0.800	0.840	0.480	0.540	0.142	0.727
Fantasia	0.750	0.950	0.660	0.870	0.550	0.730	0.882	0.833
GP711MarkerLFM5z4	0.930	1.000	0.500	0.640	0.950	0.540	0.837	1.000
GP711MarkerLFM5z5	0.760	0.970	0.310	0.480	0.900	0.690	0.717	1.000
InternalBleeding5	0.940	1.000	1.000	0.920	1.000	0.460	0.883	0.914
Italianpowerdemand	0.010	0.740	0.590	0.010	0.440	0.450	0.087	0.259
Lab2Cmac011215EPG5	0.990	0.980	0.340	0.600	0.990	0.770	0.477	0.672
Lab2Cmac011215EPG6	0.410	0.100	0.260	0.100	0.170	0.700	0.118	0.209
MesoplodonDensirostris	1.000	0.840	0.790	1.000	1.000	0.850	0.532	0.947
PowerDemand1	0.870	0.440	0.490	0.760	0.950	0.720	0.433	0.810
TkeepFirstMARS	0.010	0.150	0.020	0.020	0.230	0.520	0.018	0.061
TkeepSecondMARS	0.830	1.000	0.160	0.120	0.950	0.720	0.057	0.625
WalkingAceleration5	0.990	1.000	0.910	0.870	0.930	0.940	0.634	0.843
apneaecg	0.400	0.200	0.250	0.310	0.260	0.580	1.000	1.000
apneaecg2	0.650	1.000	1.000	1.000	0.650	0.790	0.213	0.213
gait1	0.180	0.360	0.070	0.410	0.520	0.630	0.204	0.144
gaitHunt1	0.080	0.430	0.020	0.100	0.300	0.810	0.008	0.007
insectEPG2	0.120	0.230	0.140	0.810	0.960	0.650	0.093	0.385
insectEPG4	0.980	1.000	0.460	0.210	0.850	0.690	0.068	0.840
ltstdbs30791AS	1.000	1.000	1.000	1.000	1.000	0.780	0.080	0.120
mit14046longtermecg	0.450	0.590	0.530	0.580	0.600	0.790	0.939	0.939
park3m	0.150	0.640	0.200	0.630	0.930	0.630	0.232	0.753
qtdbSel1005V	0.410	0.650	0.400	0.390	0.530	0.520	0.494	0.412
qtdbSel100MLII	0.420	0.840	0.410	0.600	0.870	0.620	0.417	0.402
resperation1	0.000	0.150	0.030	0.010	0.030	0.750	0.006	0.006
s20101mML2	0.690	0.710	0.150	0.050	0.080	0.640	1.000	1.000
sddb49	0.890	1.000	0.880	0.940	1.000	0.660	0.781	0.820
sel840mECG1	0.160	0.660	0.280	0.210	0.360	0.620	0.247	0.235
sel840mECG2	0.150	0.390	0.320	0.280	0.210	0.590	0.484	0.507
tilt12744mtable	0.070	0.240	0.100	0.000	0.030	0.480	0.031	0.080
tilt12754table	0.230	0.640	0.040	0.060	0.050	0.600	0.017	0.084
tiltAPB2	0.920	0.980	0.360	0.830	0.380	0.770	0.416	0.692
tiltAPB3	0.170	0.850	0.030	0.050	0.090	0.680	0.029	0.068
weallwalk	0.000	0.580	0.070	0.130	0.170	0.730	0.041	0.667

J Negative Transfer

For all our synthetic data experiments, we train with 8 layers, the attention is of 1024 dimension shared between 8 heads, following to a maximum width of 4098. We used no dropout. The model is trained on negative log likelihood loss of a Gaussian. The model was trained on 100K steps with a fixed patch size of 1. For optimization, we used a batch size of 64 and employed the AdamW optimizer with the following hyperparameters: lr $= 1e-4$, weight decay $= 1e-5$, β_1 = 0.9, and β_2 = 0.98. A learning rate scheduler was applied, featuring linear warmup for the first 5,000 steps, followed by cosine annealing down to 1e-5.

For our synthetic data, we use wavelet functions:

$$x_t = (d * t/T - c) \sin(a * t + b) + \epsilon_t$$
$$\epsilon_t \sim \mathcal{N}(0, 0.2)$$

where the parameters are $\{a, b, c, d\}$ and T is the number of time steps of the time series, and GARCH:

$$x_t = \mu + \epsilon_t$$
$$\epsilon_t \sim \mathcal{N}(0, \sigma_t)$$
$$\sigma_t^2 = \omega + a x_{t-1}^2 + b \epsilon_{t-1}^2$$

where the parameters are $\{\omega, a, b, \sigma_0, \mu\}$.

J.1 Pre-training with GARCH and Wavelet Data

For generating GARCH data, we use:

$$\mu = 0$$
$$\sigma_0 = \omega \sim \mathrm{U}(0, 1)$$
$$a \sim \mathrm{U}(0, 1)$$
$$b \sim \mathrm{U}(0, 1 - a)$$

For generating wavelet data, we use:

$$a \sim \{0.1, 0.2, 0.6, 0.8\}$$
$$b \sim \{0, 5, 10\}$$
$$c \sim \{0.0, 0.3, 0.6, 0.9\}$$
$$d \sim \{0.5, 0.9\}$$

where the sets denote a uniform sample from those choices.

J.2 Bayesian MCMC

For the wavelet distribution, we use:

$$T = 32$$
$$a \sim \mathrm{U}(0, 1)$$
$$b \sim \mathrm{U}(0, 1)$$
$$c \sim \mathrm{U}(0, 1)$$
$$d \sim \mathrm{U}(0, 1)$$

where U denotes a Uniform distribution.

For the GARCH distribution, we use:

$$\mu = 0$$
$$\sigma_0 = \omega = 0.3^2$$
$$a \sim \mathrm{U}(0, 0.2)$$
$$b \sim \mathrm{U}(0, 0.2)$$

For computing the NLL and the mean, we need to find the probability that a time-series comes from each distribution. For this, we computed the log-likelihood through 10K samples from the prior. For computing the posterior of the parameters given the data for each model, we use the NUTS sampler [20].

J.3 Additional Ablation Experiment on Dataset Sizes

In the additional experiment (see Table 30 below), we alter the amount of GARCH and Wavelet Data, using half GARCH and half Wavelet. The original training data contains 9.6M GARCH and Wavelet functions, we downsample them to include 2.4M (25%) and 0.4M (<5%) training datapoints only. The results show that for a shorter context length, especially, fewer amount of training data provide significantly negative impact on the NLL score. For longer context length, such drop in performance is not that significant.

Table 30: Zero-shot NLL(↓) for models trained on different amount of Wavelet & GARCH

Training data size	Context Len.	Wavelet Pred.	GARCH Pred.
400000	32	-0.0382	0.2316
2400000	32	-0.0635	0.2017
9600000	32	**-0.0732**	0.1176
400000	128	-0.0698	0.1109
2400000	128	-0.0671	**0.1058**
9600000	128	**-0.0733**	0.1137

K Additional Ablation Studies

For pre-training, we use the same setup as in Section J. For fine-tuning, similar to fine-tuning Delphyne, we early-stop based on a validation set.

K.1 Context Length

For these experiments, we use the same wavelet data generation as in Section J.1. We used a finite set of configurations to test how well the model is able to create features specific to a dataset. The intuition is that with a smaller context length, the pre-training does not create features specific to the type of wavelet that generated the data but longer context lengths do.

K.1.1 Architecture for Context Lengths

Small The small model is a transformer encoder of 6 encoder layers, a context length of 128, and a hidden dimension size of 512. It uses 8 attention heads, a feedforward dimension of 2048, and applies a GELU activation function. The model is designed to output attention weights, and features a dropout rate of 0.1 to prevent overfitting. It is trained using the Adam optimizer with a learning rate of 1e-4, betas of 0.9 and 0.98, and a weight decay of 1e-5. The configuration includes training, validation, and test batch sizes of 128, with warmup steps for the learning rate set at 5,000 out of 100,000 total training steps.

Medium The medium model contains 8 encoder layers and a context length of 128, featuring a hidden dimension size of 1024. The model uses 8 attention heads and a feedforward dimension of 4098 with GELU activation, while a dropout rate of 0.1 is applied for regularization. The model outputs attention weights and employs batch sizes of 64 for training, validation, and testing. It uses the Adam optimizer with a learning rate of 1e-4, betas of 0.9 and 0.98, a weight decay of 1e-5, and includes 5,000 warmup steps in a total of 100,000 training steps.

K.2 Masking Ratio

For these experiments, we use the same wavelet data generation as in Section J.1.

K.3 Multivariate

For these experiments, we use the same wavelet data generation as in Section J.1. The main difference is each sample is four time series. We model two scenarios: one where the Wavelet data across rows are correlated, and another where they are uncorrelated. In the correlated scenario, the time-series data is generated using the same Wavelet function, differing only by additive Gaussian noise. In the uncorrelated scenario, the data is generated using different Wavelet functions.

K.4 Output Distribution

We use the same hyperparameter configuration for training Delphyne-A, on three different output distributions: (1) Single Student T, (2) a mixture of Student-T distributions, and (3) a mixture of Normal, Student's-T, Log-normal, and negative binomial distributions. For every 10,000 training steps, we finetune the models on stock NLL task in the experiment section. The overall results are shown in Fig. **??**.

L List of Popular Models

Table 31: Comparison of Pre-trained Time-series Model

Feature	**MOMENT** [18]	**MOIRAI** [47]	**Lag-Llama** [34]	**Chronos** [2]	**TimesFM** [8]	**TimeGPT-1** [13]	**TTM** [10]	**Delphyne** (This paper)
Base Architecture	T5 encoder	Encoder-only transformer	Llama	T5 (encoder-decoder)	Decoder-only	Transformer	MLP-Mixer	Encoder-only transformer
Evaluation Tasks	Forecasting, Classification, Anomaly detection, Imputation	Forecasting	Forecasting	Forecasting	Forecasting	Forecasting	Forecasting	Forecasting, Anomaly detection
Tokenization	Fixed-length patches	Multi-scale Patches	Lag features	Scaling, Quantization	Fixed-length patches	?	Adaptive Patching	Fixed-length patches
Objective	Reconstruction Error	Forecast NLL of mixed-distributions	NLL of Student's t distribution	Cross-entropy loss	Forecast Error	?	Forecasting Error	Forecast NLL of mixture of Student T's distributions
Distribution Prediction / Uncertainty Quantification		✓	✓	✓		✓(post hoc)		✓
Multivariates?	✓(Anyvariate attention w. Channel independence)	✓(Anyvariate attention + Flattening)		✓		✓(?)	(Channel Independence + Mixing)	✓(Anyvariate attention +Flattening)
Context length	512	1000–5000	1024	512	512	?	512	512 x 32

We provide a full table of the foundation models in Table 31. We compare popular models between 2022-2024: MOMENT [18], MOIRAI [43], Lag-Llama [34], Chronos [2], TimesFM [8], TimeGPT-1 [13], TTM [10].

Among these popular models, only MOIRAI, Lag-Llama and TimeGPT-1 are able to provide output distributions and uncertainty quantifications. Specifically, Lag-Llama utilizes a single Student's T distribution which is less ideal to model asymmetries in forecasts, which is shown in our previous experiment. TimeGPT-1 uses a categorical output distribution. While it may potentially model any multi-modal distributions, the output distribution is tied to TimeGPT-1's language model architecture and training objective, offering less flexibility overall. Delphyne utilizes a mixture of Student's T distributions, which are simpler and more stable, as shown in our previous study.

Many existing time-series foundation models excel in modeling single variates, which ignore the potential dependencies between variates (for example, when modeling US stock returns, many stocks in the same sectors are inter-correlated). We use the same any-variate attention mechanism as MOIRAI; we demonstrate in the previous section that any-variate attention performs reasonably well when both the variates are strongly or weakly correlated.

While many pre-trained time-series model aim to adapt to different forecast lengths, TTM has fixed forecast lengths. Its public model has a maximum context length of 1024 and a forecast length of 96, which is limited for various financial tasks. We argue that a good pre-trained time-series model should be agnostic to downstream tasks' forecast lengths and number of variates. In this context, Delphyne offers more flexibility.

Many popular time-series models, such as MOIRAI, Lag-Llama, and TTM, employ various patch sizes or use additional frequency information to capture different frequencies within datasets. We argue that these different patching methods aim to address the negative transfer effect across datasets. Since datasets across domains are collected at varying frequencies, these models leverage frequency information to create distinct embeddings for data at different granularities. In contrast, we believe that fine-tuning, despite being a post-hoc solution, offers the most effective means of mitigating the negative transfer effect.

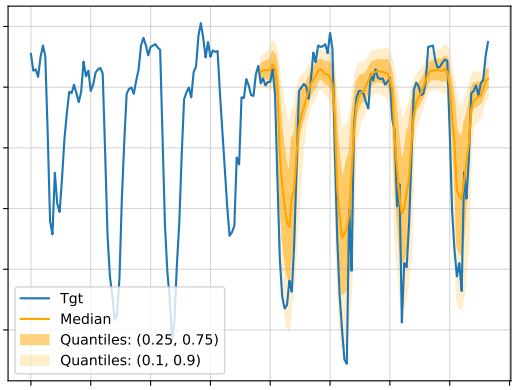

Figure 7: Visualization of fine-tuned forecasts from Delphyne-A on ETTh1 dataset. The quantiles represented are 0.1, 0.25, 0.5, 0.75, and 0.9.

M Visualizations

Fig. 7 shows the visualization on ETTh1. Fig. 8 and Fig. 9 show the fine-tuned forecast visualizations on stock variance and NLL. Fig. 10 shows the forecast of nowcasting company revenue. Fig. 11, Fig. 12 and Fig. 13 show the forecast on bars data using Delphyne, MOMENT and TTM.

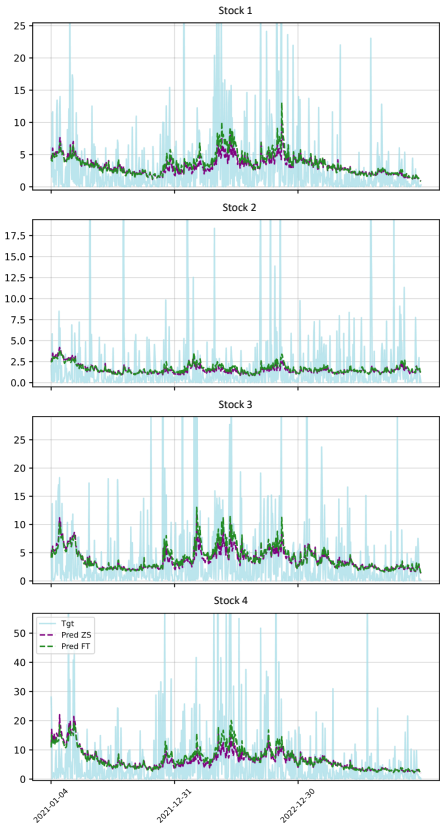

Figure 8: Visualization of fine-tuned forecasts from Delphyne-A on Stock Variance dataset. Note that since sometimes the squared returns are very large, we clip the plot but not the data during training and evaluation.

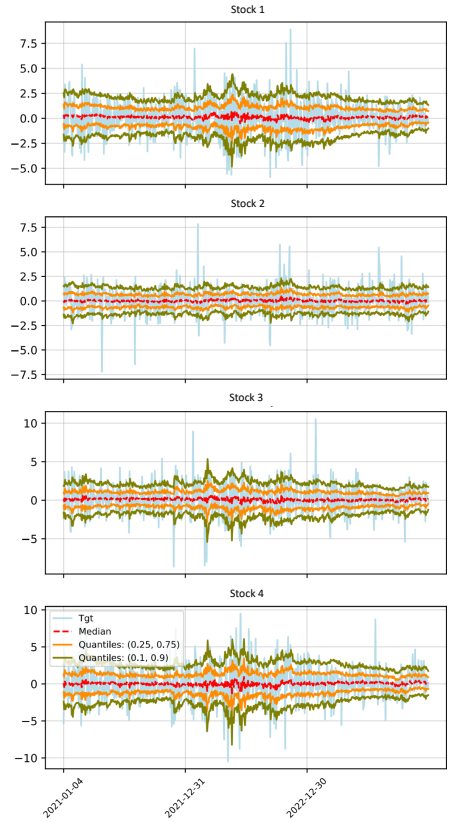

Figure 9: Visualization of fine-tuned probabilistic forecasts from Delphyne-A on Stock NLL dataset.

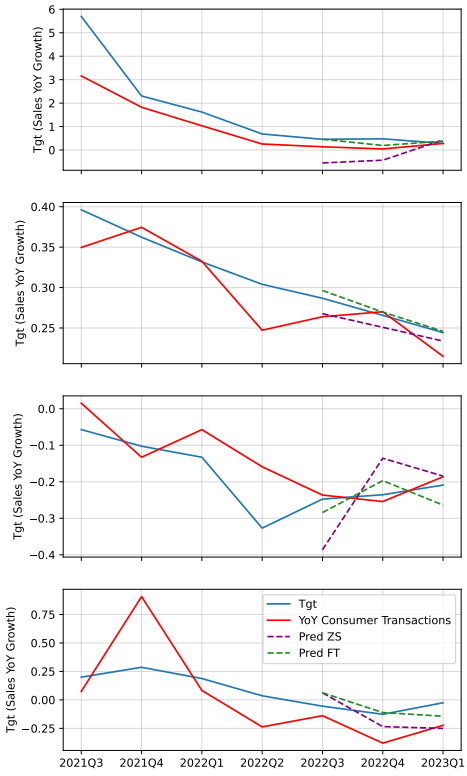

Figure 10: Visualization of fine-tuned forecasts from Delphyne-A on Nowcasting Company Revenue dataset.

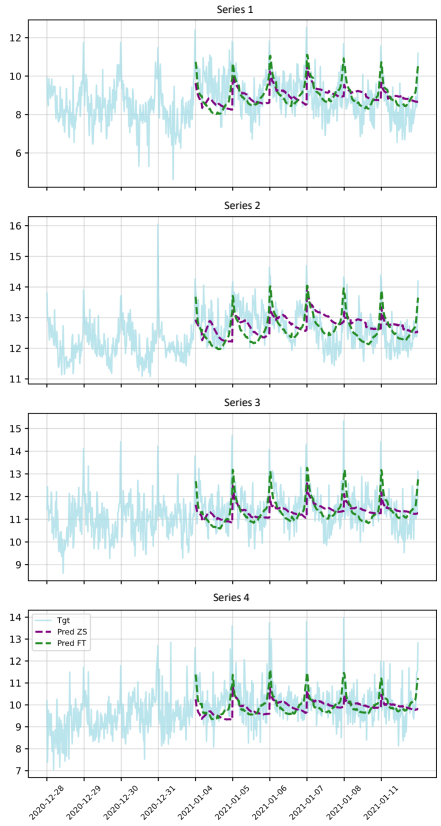

Figure 11: Visualization of fine-tuned forecasts from Delphyne-A on Financial Bars dataset.

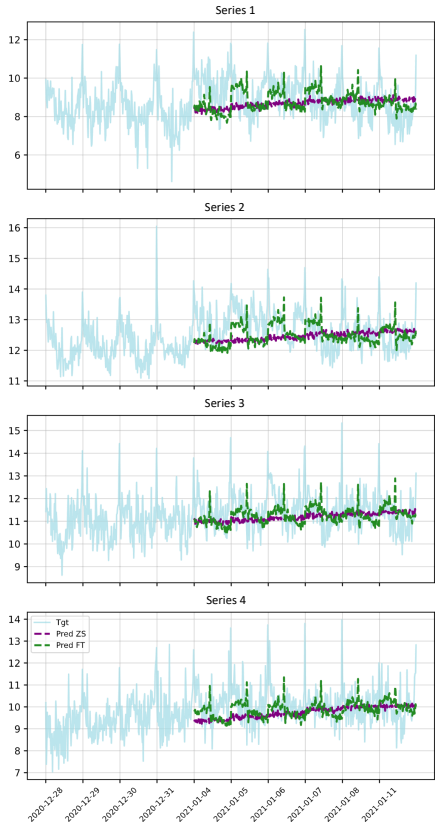

Figure 12: Visualization of fine-tuned forecasts from MOMENT on Financial Bars dataset.

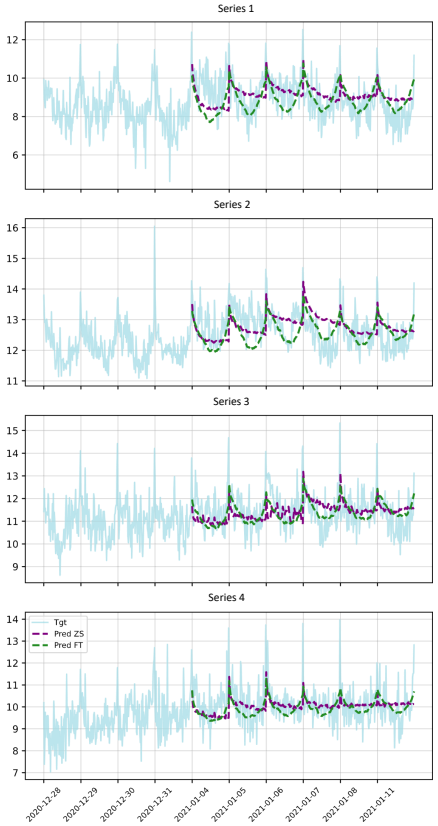

Figure 13: Visualization of fine-tuned forecasts from TTM on Financial Bars dataset.