The Tabular Foundation Model TabPFN Outperforms Specialized Time Series Forecasting Models Based on Simple Features

Shi Bin Hoo[1] **Samuel Müller**[1] **David Salinas**[1] **Frank Hutter**[2,1]

[1]University of Freiburg
[2]ELLIS Institute Tübingen
Correspondence to `{hoos,muellesa}@cs.uni-freiburg.de`

Abstract

Foundation models have become popular in forecasting due to their ability to make accurate predictions, even with minimal fine-tuning on specific datasets. In this paper, we demonstrate how the newly released regression variant of TabPFN, a general tabular foundation model, can be applied to time series forecasting. We propose a straightforward approach, TabPFN-TS, which pairs TabPFN with simple feature engineering to achieve strong forecasting performance. Despite its simplicity and with only 11M parameters, TabPFN-TS outperforms Chronos-Mini, a model of similar size, and matches or even slightly outperforms Chronos-Large, which has 65-fold more parameters. A key strength of our method lies in its reliance solely on artificial data during pre-training, avoiding the need for large training datasets and eliminating the risk of benchmark contamination. To encourage reproducibility, we provide a Colab Notebook[1] to demonstrate our approach.

1 Introduction

Time series forecasting has received a lot of attention due to its large set of high-impact applications, in areas such as energy, finance and logistics. Recently, deep learning has gained popularity in forecasting for its ability to integrate covariates and custom likelihoods [Benidis et al., 2022]. However, these methods typically require large amounts of training data to outperform simpler approaches. To address this, several lines of work have explored pre-training foundation models on large collections of time series datasets, followed by zero-shot or few-shot fine-tuning.

In this work, we demonstrate that the tabular foundation model TabPFN [Hollmann et al., 2023][2] performs on par with, or slightly outperforms state-of-the-art time-series foundation models *out-of-the-box* in forecasting. This shows that TabPFN is sufficiently general, eliminating the need for time-series-specific priors [Dooley et al., 2024] or extensive pre-training on real-world time series datasets as in Ansari et al. [2024].

[1]`https://bit.ly/tabpfn-ts`

[2]We use a recent version of TabPFN, for which a formal publication is not yet available, can be accessed at `https://github.com/automl/tabpfn-client`.

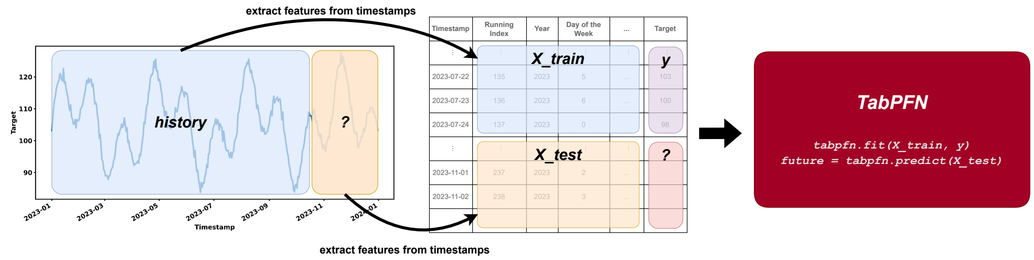


Figure 1: Overview of TabPFN-TS. Given a time series, we derive features from the timestamps to form both `X_train` and `X_test`. The target values of the history are used as `y_train`. These three variables are then used by TabPFN to predict the target values of the future timestamps.

2 Related work

Traditional forecasting methods, such as ARIMA and ETS [Hyndman, 2018], are widely used but are often outperformed by deep learning models when ample training data is available [Salinas et al., 2020].

Recently, foundation models for time series have been developed, particularly suited for smaller datasets (fewer than a few million time steps). These models, pre-trained on real-world time-series datasets, are applied to new time series through zero-shot prediction, without fine-tuning. Rasul et al. [2023] introduced an auto-regressive model trained on large datasets that performs well in zero-shot settings and improves further with fine-tuning. Other works have explored similar foundational approaches [Woo et al., 2024, Dooley et al., 2024], as surveyed by Liang et al. [2024].

Another line of work involves adapting architectures from other domains and modalities to create time series foundation models. Gruver et al. [2024] and Ansari et al. [2024] demonstrated strong forecasting performance using models designed for language tasks, while Yang et al. [2024] applied Vision Transformers to time series forecasting.

In this work, we extend the tabular foundation model from Hollmann et al. [2023] to time series forecasting, notably without requiring pre-training on real-world or synthetic time series datasets.

3 Method

We frame time series forecasting as a tabular regression problem, where each time series is treated as an independent table, as shown in Figure 1. Tabular regression uses the training data (in this case, the history of the series) to predict future target values. Unlike auto-regressive methods, our approach generates multi-step-ahead predictions by relying solely on historical information. Moreover, each time series is processed independently, with no information shared between series. As a result, our method is a local, multi-step-ahead forecasting approach.

3.1 Featurizing Time Series Data for TabPFN

Leveraging TabPFN for forecasting requires capturing temporal relationships through appropriate feature engineering. We derive features directly from the timestamps, excluding lagged and auto-regressive features (e.g., moving averages and lag terms), as they rely on future values and are therefore unsuitable for non-auto-regressive, multi-step-ahead prediction settings. All of our features describe the current time stamp, independent of other time steps.

Sine and Cosine Encoding To capture the cyclical nature of most calendar-based features (excluding the year), we apply sine and cosine transformations. This replaces a feature with two new features representing its sine and cosine values, with the period set to match the feature's natural cycle (e.g. 24 hours for the hour of the day, 7 days for the day of the week).

Calendar Features From each timestamp, we extract several calendar-based features: the year, the hour of the day (sine and cosine), the day of the week (sine and cosine), the day of the month

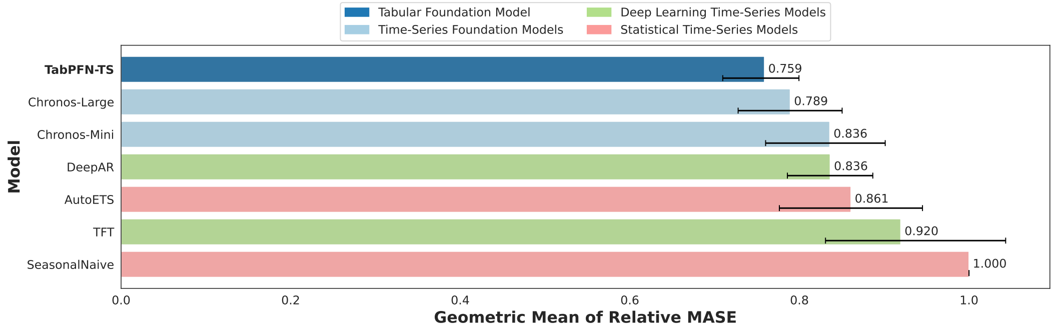


Figure 2: Forecasting performance of various models on 24 datasets. MAE scores are normalized using the scores of Seasonal Naive to compute Relative MASE, then aggregated via geometric mean over the datasets. 95% confidence interval is included. Lower is better.

(sine and cosine), the day in the year (sine and cosine), the week of the year (sine and cosine), and the month of the year (sine and cosine).

Running Index To introduce a temporal reference within the timeline, we include the index of each time step as a feature (e.g., 0 for the first time step in the time series, 4 for the fifth). This provides a straightforward and effective way to track the progression of time across the observations.

3.2 Forecasting with TabPFN

For each time series, we transform the sequence into a table with the aforementioned features, as outlined in Figure 1. This table is then fed to TabPFN as an "i.i.d." regression task. Since the available TabPFN implementation does not support batched inference, we process each time series individually.

4 Experiments

In this section, we aim to rigorously assess the point forecast accuracy of TabPFN-TS. All evaluations are conducted with AutoMLBenchmark [Gijsbers et al., 2024], following the same settings used in the evaluation of AutoGluon-TS [Shchur et al., 2023].

Datasets We utilize 24 of the 29 datasets from the AutoGluon-TS evaluation, excluding 5 datasets due to their large size, which prevented TabPFN-TS from completing within the 4 hour time limit. Adhering to this constraint ensures a fair comparison with the results reported in AutoGluon-TS, where we reference baseline results. Despite the exclusions, the remaining datasets span a wide range of application domains and exhibit diverse time series characteristics. Table A.1 outlines the datasets and their respective statistics.

TabPFN Configuration We use a recent TabPFN implementation from the following hosted endpoint: [3]. TabPFN internally models the full distribution of the target values, allowing for flexible aggregation into point prediction (explained in detail in Appendix A.2.1). Given that our evaluation metric (MASE, described in 4) is a scaled variant of mean absolute error (MAE), we configure TabPFN to use the median prediction, which minimizes MAE [Schwertman et al., 1990]. All other settings are kept at their defaults. Additional configuration details are provided in Appendix A.2.2.

Baselines We evaluate the performance of TabPFN-TS against a diverse set of baselines, including statistical, deep-learning, and pre-trained models. From the statistical forecasting literature Hyndman [2018], we include SeasonalNaive, AutoETS, AutoARIMA and AutoTheta. For neural forecasting baselines, we compare against DeepAR and TFT [Lim et al., 2021], while the pre-trained models include Chronos-Mini and Chronos-Large. Implementation details are provided in Appendix A.3.

[3] `https://github.com/automl/tabpfn-client`.

Evaluation Metrics We follow the evaluation protocol outlined by Ansari et al. [2024] and Shchur et al. [2023]. Point forecast accuracy is assessed using the mean absolute scaled error (MASE) [Hyndman and Koehler, 2006], which scales the absolute forecast error by the historical seasonal error of the time series. Consistent with Ansari et al. [2024], we aggregate the relative scores using the geometric mean.

4.1 Main Results

TabPFN-TS outperforms all baselines (see Figure 2). With only 11M parameters, it surpasses Chronos-Mini (20M) by 7.7% and shows a modest improvement over Chronos-Large (710M, with 65 times more parameters) by 3.0%. For further insights, we provide complementary information in the Appendix A.4, including raw MASE scores for individual datasets (Table 2), visualizations of TabPFN-TS' predictions (Figure 6 and 7), and a latency comparison across models (Table 3).

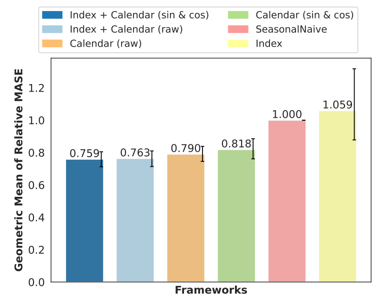


Figure 3: TabPFN-TS performance with different feature combinations. Seasonal Naive is included for reference.

4.2 Ablations

In this section, we conduct a series of ablations to better understand the surprisingly strong performance of TabPFN-TS.

Which features are the best? We experimented with various features for time series forecasting with TabPFN, including a running index, raw calendar features (e.g. day of the week represented as 0-6), and sine-cosine transformed calendar features. The results, outlined in Figure 3, show that using only the index, similar to how Chronos is prompted, yields subpar performance. In contrast, TabPFN performs significantly better when calendar features are present.

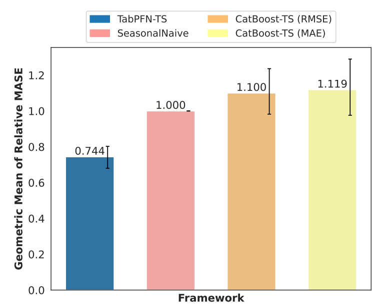


Figure 4: Performance comparision of TabPFN-TS, CatBoost-Median, CatBoost-Mean, and Seasonal Naive on a subset of 16 datasets out of 24.

Can Any Tabular Regressor Achieve This? To assess whether the effectiveness of our method stems from general tabular regression or from TabPFN, we replaced TabPFN with the default CatBoost regressor, keeping the rest of the pipeline unchanged. As shown in Figure 4, CatBoost falls short of our performance and is even outperformed by Seasonal Naive. While boosted trees have shown strong results in forecasting [Januschowski et al., 2022], they are typically used as global model and rely on lag and aggregate features. This suggests TabPFN's unique capability as a tabular foundation model for time series forecasting.

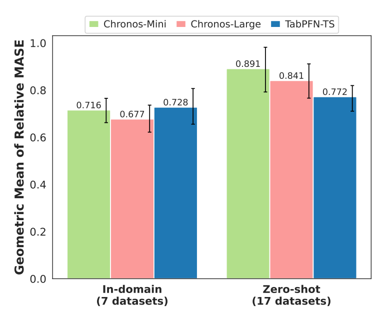


Figure 5: Forecasting performance grouped by Chronos' in-domain vs zero-shot datasets split.

Chronos' Zero-shot vs In-domain Performance Unlike TabPFN, Chronos is pre-trained on real-world time series data, with overlap in our evaluation datasets. To better compare the performances, we grouped the results into in-domain and zero-shot categories based on the data split from Chronos's paper. As shown in Figure 5, Chronos outperforms TabPFN-TS on the datasets it was pre-trained on, suggesting that additional dataset-specific training can improve performance when computational resources are available. However, in zero-shot settings — where Chronos has not been trained on the dataset — TabPFN-TS significantly outperforms Chronos, underscoring its strength as a foundation model for time series forecasting.

5 Conclusion

In this paper, we presented evidence suggesting that tabular foundation models, like TabPFN, may be general enough to be the incumbent for time series datasets. By using a simple set of timestamp-derived features, our approach matches or slightly outperforms Chronos-T5 (Large), which, to our knowledge, is one of the strongest time series foundation models. This demonstrates the potential of tabular foundation models in time series forecasting, though further research is needed to confirm their broader applicability.

Acknowledgements

Frank Hutter acknowledges the financial support of the Hector Foundation. We acknowledge the funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant number 417962828 and Emmy Noether grant HU1900/2-1. We also acknowledge the funding by the European Union (via ERC Consolidator Grant DeepLearning 2.0, grant no. 101045765). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Funded by
the European Union

References

A. Alexandrov, K. Benidis, M. Bohlke-Schneider, V. Flunkert, J. Gasthaus, T. Januschowski, D. C. Maddix, S. Rangapuram, D. Salinas, J. Schulz, L. Stella, A. C. Türkmen, and Y. Wang. GluonTS: Probabilistic Time Series Modeling in Python. *arXiv preprint arXiv:1906.05264*, 2019.

A. F. Ansari, L. Stella, C. Turkmen, X. Zhang, P. Mercado, H. Shen, O. Shchur, S. S. Rangapuram, S. P. Arango, S. Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

K. Benidis, S. S. Rangapuram, V. Flunkert, Y. Wang, D. Maddix, C. Turkmen, J. Gasthaus, M. Bohlke-Schneider, D. Salinas, L. Stella, et al. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6):1–36, 2022.

S. Dooley, G. S. Khurana, C. Mohapatra, S. V. Naidu, and C. White. Forecastpfn: Synthetically-trained zero-shot forecasting. *Advances in Neural Information Processing Systems*, 36, 2024.

P. Gijsbers, M. L. P. Bueno, S. Coors, E. LeDell, S. Poirier, J. Thomas, B. Bischl, and J. Vanschoren. Amlb: an automl benchmark. *Journal of Machine Learning Research*, 25(101):1–65, 2024. URL `http://jmlr.org/papers/v25/22-0493.html`.

N. Gruver, M. Finzi, S. Qiu, and A. G. Wilson. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36, 2024.

N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023.

R. Hyndman. *Forecasting: principles and practice*. OTexts, 2018.

R. J. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL `https://www.sciencedirect.com/science/article/pii/S0169207006000239`.

T. Januschowski, Y. Wang, K. Torkkola, T. Erkkilä, H. Hasson, and J. Gasthaus. Forecasting with trees. *International Journal of Forecasting*, 38(4):1473–1481, 2022.

Y. Liang, H. Wen, Y. Nie, Y. Jiang, M. Jin, D. Song, S. Pan, and Q. Wen. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6555–6565, 2024.

B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

K. Rasul, A. Ashok, A. R. Williams, A. Khorasani, G. Adamopoulos, R. Bhagwatkar, M. Biloš, H. Ghonia, N. V. Hassen, A. Schneider, et al. Lag-llama: Towards foundation models for time series forecasting. *arXiv preprint arXiv:2310.08278*, 2023.

D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

N. C. Schwertman, A. Gilks, and J. Cameron. A simple noncalculus proof that the median minimizes the sum of the absolute deviations. *The American Statistician*, 44(1):38–39, 1990.

O. Shchur, C. Turkmen, N. Erickson, H. Shen, A. Shirkov, T. Hu, and Y. Wang. AutoGluon-TimeSeries: AutoML for probabilistic time series forecasting. In *International Conference on Automated Machine Learning*, 2023.

G. Woo, C. Liu, A. Kumar, C. Xiong, S. Savarese, and D. Sahoo. Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592*, 2024.

L. Yang, Y. Wang, X. Fan, I. Cohen, J. Chen, Y. Zhao, and Z. Zhang. Vitime: A visual intelligence-based foundation model for time series forecasting, 2024. URL `https://arxiv.org/abs/2407.07311`.

A Appendix

A.1 Dataset

Table A.1 provides the complete list of datasets used in our empirical evaluation. All datasets are sourced from Alexandrov et al. [2019].

Table 1: Datasets used for evaluation and their respective statistics.

Dataset	Domain	Freq.	Prediction Length	Num. Series	Series Length		
					min	avg	max
car_parts	retail	M	12	2674	51	51	51
cif_2016	banking	M	12	72	28	98	120
covid_deaths	healthcare	D	30	266	212	212	212
electricity_weekly	energy	W	8	321	156	156	156
fred_md	economics	M	12	107	728	728	728
hospital	healthcare	M	12	767	84	84	84
kdd_cup_2018	nature	H	48	270	9504	10897	10920
m1_monthly	various	M	18	617	48	90	150
m1_quarterly	various	Q	8	203	18	48	114
m1_yearly	various	A	6	181	15	24	58
m3_monthly	various	M	18	1428	66	117	144
m3_other	various	A	8	174	71	76	104
m3_quarterly	various	Q	8	756	24	48	72
m3_yearly	various	A	6	645	20	28	47
m4_daily	various	D	14	4227	107	2371	9933
m4_hourly	various	H	48	414	748	901	1008
m4_weekly	various	W	13	359	93	1035	2610
nn5_daily	finance	D	56	111	791	791	791
nn5_weekly	finance	W	8	111	113	113	113
pedestrian_counts	finance	H	48	66	576	47459	96424
tourism_monthly	finance	M	24	366	91	298	333
tourism_quarterly	various	Q	8	427	30	99	130
tourism_yearly	various	A	4	518	11	24	47
vehicle_trips	transport	D	7	329	70	128	243

A.2 Technical Overview of TabPFN

A.2.1 A brief overview of TabPFN's working principle

TabPFN approaches tabular regression by predicting a probability distribution over possible target values, rather than a single deterministic output. In the context of time series forecasting, when provided with a future timestamp, TabPFN generates a probability distribution for the corresponding target value.

This probabilistic approach allows flexibility in obtaining point forecasts. Users can aggregate the distribution using methods such as the mean or median, depending on the forecasting objective. The use of a full probability distribution enables better uncertainty quantification and provides a more robust forecast compared to single-point predictions.

Additionally, TabPFN is naturally suited to quantile prediction in forecasting, as it can directly predict the probability of different quantiles. However, in this paper, we focus on point accuracy, leaving quantile accuracy for future work.

A.2.2 Implementation of TabPFN

The implementation of TabPFN, available through a hosted endpoint[4] supports datasets with up to 10K data points and 500 features. It allows users to configure various internals, such as pre-processing, model selection, and ensembling.

For our experiments, we selected the `2noar4o2` model due to its superior empirical performance and configured the regressor to perform median prediction. The following code snippets demonstrate this setup.

```
from tabpfn_client import TabPFNRegressor

tabpfn = TabPFNRegressor(model_path="2noar4o2")

tabpfn.fit(X_train, y_train)
pred = tabpfn.predict_full(y_train)["median"]
```

A.3 Baselines Implementation

After verifying that our results for Seasonal Naive aligned with those reported by Shchur et al. [2023], we sourced the remaining baseline results, except for the Chronos variants, from their paper. We re-evaluated Chronos-Mini and Chronos-Large on an NVIDIA V100 machine for further comparison.

For Seasonal Naive, Chronos-Mini, and Chronos-Large, we utilized the AutoGluon forecasting library [Shchur et al., 2023] with default settings.

[4] https://github.com/automl/tabpfn-client

A.4 Additional Results

This section complements the main result (4.1) by providing additional details to the experimental results.

A.4.1 Mean Absolute Scaled Error (MASE) Scores

Table 2 presents the raw MASE scores for all models across the datasets. Additionally, we report the average rank of each model, with lower ranks indicating better overall performance.

Table 2: MASE scores of all models on various time-series datasets. Lower is better.

	Tabular Foundation Model	Time-Series Foundation Model		Deep Learning Time-Series Model		Statistical Time-Series Model			
	TabPFN-TS	Chronos-Large	Chronos-Mini	DeepAR	TFT	AutoARIMA	AutoETS	AutoTheta	SeasonalNaive
car_parts	0.796	0.823	0.821	**0.749**	0.751	1.118	1.133	1.208	1.127
cif_2016	**0.885**	1.000	1.040	1.278	1.372	1.069	0.898	1.006	1.289
covid_deaths	6.471	7.580	7.569	7.166	**5.192**	6.029	5.907	7.719	8.977
electricity_hourly	1.335	1.119	**1.113**	1.251	1.389	-	1.465	-	1.230
electricity_weekly	**1.704**	1.723	1.865	2.447	2.861	3.009	3.076	3.113	3.037
fred_md	0.521	0.499	**0.469**	0.634	0.901	0.478	0.505	0.564	1.101
hospital	**0.757**	0.808	0.813	0.771	0.814	0.820	0.766	0.764	0.921
kdd_cup_2018	**0.727**	0.734	0.728	0.841	0.844	-	0.988	1.010	0.975
m1_monthly	**1.040**	1.093	1.186	1.117	1.534	1.152	1.083	1.092	1.314
m1_quarterly	**1.664**	1.735	1.794	1.742	2.099	1.770	1.665	1.667	2.078
m1_yearly	3.684	4.390	5.106	3.674	4.318	3.870	3.950	**3.659**	4.894
m3_monthly	**0.853**	0.861	0.903	0.960	1.062	0.934	0.867	0.855	1.146
m3_other	2.123	2.023	2.092	2.061	1.926	2.245	**1.801**	2.009	3.089
m3_quarterly	**1.096**	1.203	1.282	1.198	1.176	1.419	1.121	1.119	1.425
m3_yearly	2.696	3.060	3.462	2.694	2.818	3.159	2.695	**2.608**	3.172
m4_daily	1.290	**1.118**	1.122	1.145	1.176	1.153	1.228	1.149	1.452
m4_hourly	0.790	**0.694**	0.762	1.484	3.391	1.029	1.609	2.456	1.193
m4_weekly	2.058	**2.039**	2.146	2.418	2.625	2.355	2.548	2.608	2.777
nn5_daily	**0.764**	0.832	0.923	0.812	0.789	0.935	0.870	0.878	1.011
nn5_weekly	**0.878**	0.945	0.970	0.915	0.884	0.998	0.980	0.963	1.063
pedestrian_counts	0.318	**0.262**	0.300	0.309	0.373	-	0.553	-	0.369
tourism_monthly	**1.432**	1.758	1.936	1.461	1.719	1.585	1.529	1.666	1.631
tourism_quarterly	1.587	1.665	1.812	1.599	1.830	1.655	**1.578**	1.648	1.699
tourism_yearly	3.066	3.686	4.176	3.476	**2.916**	4.044	3.183	2.992	3.552
vehicle_trips	**1.147**	1.170	1.260	1.162	1.227	1.427	1.301	1.284	1.302
Average Rank	**2.500**	4.083	5.417	4.083	5.583	5.955	4.500	4.739	7.750

A.4.2 Visualization of Prediction on Real Time-series Datasets

We visualize TabPFN-TS's predictions on 12 datasets selected for their high variance in MASE scores, representing significant differences in model performance. For each dataset, we choose the time series where TabPFN-TS's MASE score falls closest to the 50%, 75%, and 95% percentiles of the MASE distribution.

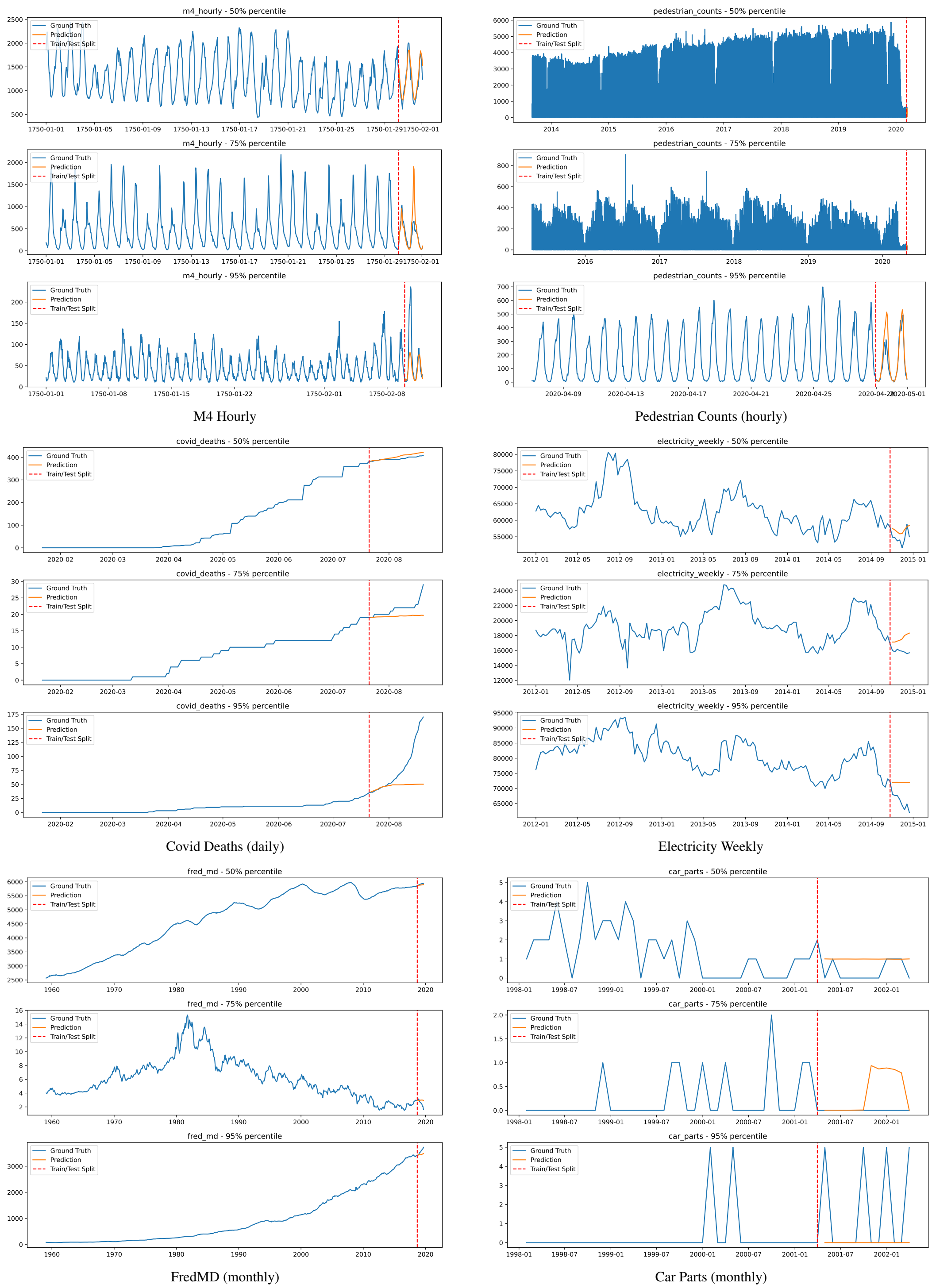
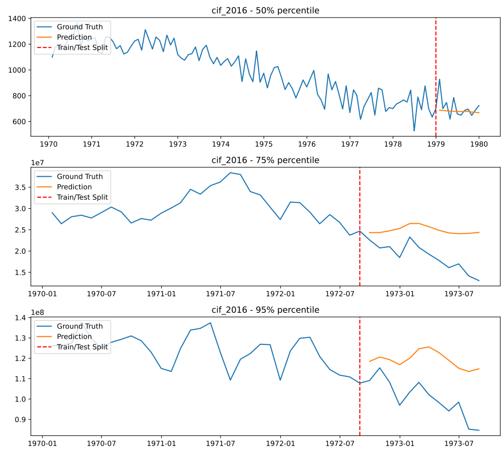
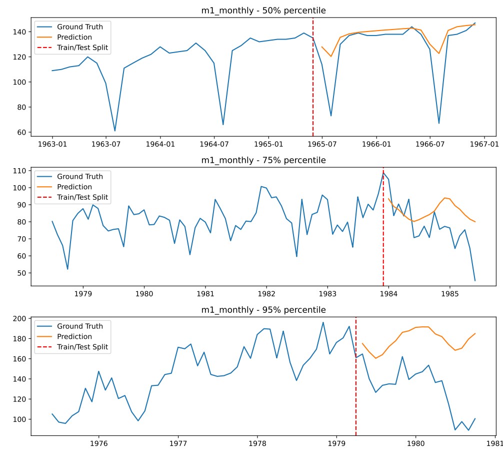


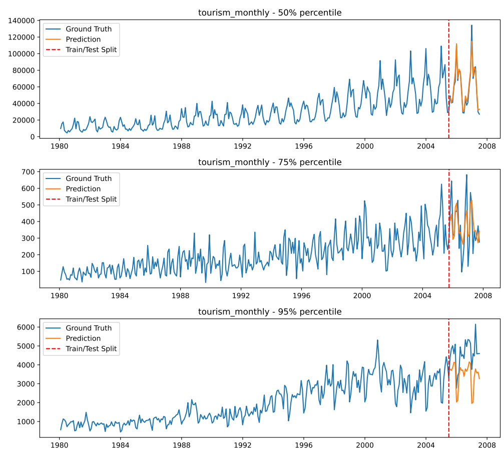
Figure 6: Visualization of TabPFN-TS's predictions on M4 Hourly, Pedestrian Counts, Covid Deaths, Electricity Weekly, FredMD, and Car Parts.



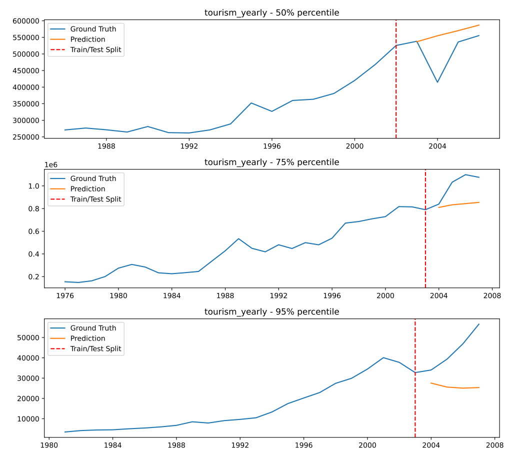
CIF 2016 (monthly)



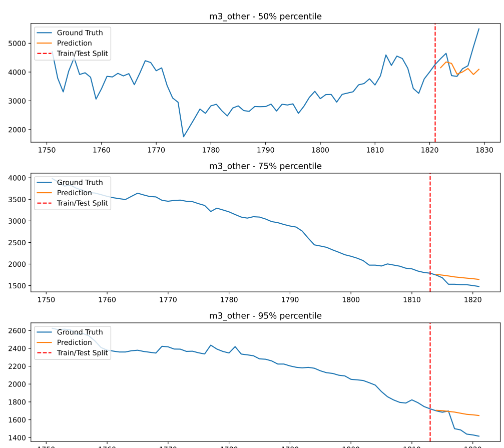
M1 Monthly



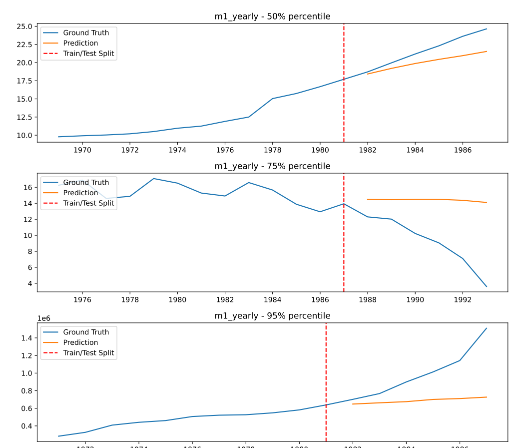
Tourism Monthly



Tourism Yearly



M3 Other (yearly)



M1 Yearly

Figure 7: Visualization of TabPFN-TS's predictions on CIF 2016, M1 Monthly, Tourism Monthly, Tourism Yearly, M3 Other, and M1 Yearly.

A.4.3 Comparison of Forecast Latency

Table 3 shows the time taken by each model to complete evaluation on each dataset. For pre-trained models, this primarily reflects inference time, while for deep learning and statistical model, it includes both training (or statistical computation) and inference time.

This comparison reveals that, despite requiring no training or fine-tuning, TabPFN-TS takes significantly longer to perform inference across all time series data. This is mainly due to TabPFN's lack of batch inference capability for time series data, where the training set (or history) is not fixed. As a result, each time series must be processed individually. Reducing forecast latency by enabling batch inference is a key area for future improvement.

Table 3: Latency comparison of models, measured in seconds. Lower is better.

	Tabular Foundation Model	Time-Series Foundation Model		Deep Learning Time-Series Model		Statistical Time-Series Model			
	TabPFN-TS	Chronos-Large	Chronos-Mini	DeepAR	TFT	AutoARIMA	AutoETS	AutoTheta	SeasonalNaive
car_parts	6155	250	19	416	555	146	35	42	0
cif_2016	152	15	6	246	372	27	32	39	0
covid_deaths	563	107	10	475	529	86	29	40	0
electricity_weekly	833	39	7	188	395	19	27	28	0
fred_md	338	38	7	406	331	146	41	33	1
hospital	1570	85	9	277	458	56	42	42	0
kdd_cup_2018	2910	239	22	746	711	-	981	1367	1
m1_monthly	1318	116	11	331	369	92	50	43	0
m1_quarterly	434	24	6	352	326	20	31	40	0
m1_yearly	325	17	6	252	313	16	26	27	0
m3_monthly	3287	250	18	306	355	239	60	45	1
m3_other	318	22	6	302	358	16	27	26	0
m3_quarterly	1624	61	8	274	360	32	35	42	0
m3_yearly	1147	41	8	354	321	21	27	27	0
m4_daily	13302	1355	90	407	503	1708	1979	1516	2
m4_hourly	1363	334	28	554	657	5093	107	49	0
m4_weekly	1087	116	12	334	468	38	32	79	0
nn5_daily	330	106	12	437	655	151	31	35	0
nn5_weekly	234	16	6	219	384	16	27	27	0
pedestrian_counts	12131	61	11	810	999	-	291	-	1
tourism_monthly	1107	126	13	266	457	615	46	42	0
tourism_quarterly	850	46	7	218	378	56	34	41	0
tourism_yearly	901	27	6	211	347	20	27	27	0
vehicle_trips	761	67	8	306	439	65	37	41	0
Average	2210	148	14	362	460	394	169	161	0