FORECASTPFN: UNIVERSAL FORECASTING FOR HEALTHCARE

Gurnoor Khurana*, **Samuel Dooley***, **Siddartha Naidu, Colin White**
Abacus.AI
San Francisco, CA 94105, USA
{gurnoor,samuel,siddartha,colin}@abacus.ai

ABSTRACT

With the proliferation of time-series analysis in the healthcare sector, each new time series requires a new model to be fit and trained on those specific data. The vast majority of time-series forecasting approaches require a training dataset. There is very recent work on zero-shot forecasting—pretraining on one series and evaluating on another—yet its performance is inconsistent depending on the training dataset. In this work, we take a different approach and devise ForecastPFN, the first universal zero-shot model, pretrained purely on synthetic data. Drawing inspiration from TabPFN, a recent breakthrough in tabular data, ForecastPFN is the first forecasting model to approximate Bayesian inference. To accomplish this, we design a synthetic time-series distribution with local and global trends, and noise. Through experiments on multiple datasets, we show that ForecastPFN achieves competitive performance without ever seeing the training datasets, compared to popular methods that were fully trained on the training dataset.

1 INTRODUCTION

Time-series forecasting problems are an important class of problems with applications in healthcare, such as diagnosing diseases, predicting disease progression, and online monitoring (Harutyunyan et al., 2019; Chimmula & Zhang, 2020), in addition to other areas such as forecasting financial metrics, supply and demand, and cloud computation monitoring (Sezer et al., 2020; Krollner et al., 2010). The vast majority of time-series forecasting approaches require training on a training dataset, however, many real-world forecasting applications have very few initial observations, sometimes just one hundred or less. While there is recent work on zero-shot forecasting (Oreshkin et al., 2021) by training on one time series and evaluating on another, its performance is inconsistent depending on the dataset on which it is trained. As in prior work (Oreshkin et al., 2021), we use the term "zero-shot forecasting" when the number of available observations for the target time series is so small (such as 100 or less) that training a deep learning model is not possible.

In this work, we take a different approach and devise ForecastPFN, the first *universal zero-shot method*, pretrained purely on synthetic data. We draw inspiration from TabPFN (Hollmann et al., 2022), a recent breakthrough in tabular data, which is trained offline to approximate Bayesian inference using prior-data fitted networks (PFNs). However, there are significant challenges when designing a general and flexible PFN in the forecasting setting. First, we design a novel synthetic time-series distribution with global trends, multi-scale seasonal trends, and noise. Next, we design a scheme for dynamically scaling across a wide range of time-series values. Finally, we design a flexible transformer architecture that can query values at any timestep in the future.

We show that by carefully designing a prior-data fitted network for forecasting, unlike TabPFN, our model achieves competitive performance *without ever seeing the training datasets*, compared to popular methods that were fully trained on the training datasets. This remarkable result is due to the structure of our synthetic priors: by encoding multi-scale seasonal trends, global trends, and noise, across a variety of parameters, our model is able to learn how to forecast general time series. Our codebase and all raw results will be open-sourced after the double-blind review process.

*Equal contribution.

2 RELATED WORK

Time-series forecasting problems have numerous applications in healthcare (Harutyunyan et al., 2019; Chimmula & Zhang, 2020) and other areas (Sezer et al., 2020; Krollner et al., 2010). Already in the 1970s, ARIMA builds an autoregressive model based on Markov processes (Box & Pierce, 1970). A popular method in practice is Prophet (Taylor & Letham, 2018), which incorporates non-linear trends and multi-scale seasonality via traditional methods. Recently, researchers have turned to deep learning to model highly nonlinear time series. DeepAR (Salinas et al., 2020) is a popular deep learning method which makes use of autoregressive methods and RNNs. Recently, many works have applied transformer models to time-series forecasting (Zhou et al., 2021; Kitaev et al., 2020; Wu et al., 2021), following the success of transformers on NLP (Vaswani et al., 2017). FEDformer (Zhou et al., 2022) is a popular recent method that incorporates Fourier transforms and the seasonal trend decomposition method into a transformer architecture. For a survey on deep learning methods for time-series forecasting, see Mahmoud & Mohammed (2021).

There is much less work on zero-shot time-series forecasting. Oreshkin et al. (2021) use an MLP with residual connections to train on a (single) real-world time series and test on a different time series. However, it achieves substantially different performance based on which training dataset is used. Recently, Adriaensen et al. (2022) designed a PFN for learning curve extrapolation; however, the setting for learning curve extrapolation is significantly different and arguably a special case of time-series forecasting. To the best of our knowledge, we are the first to design a *single, universal* model for zero-shot time-series forecasting, trained purely on synthetic data.

3 PRIOR-DATA FITTED NETWORKS FOR FORECASTING

In this section, we introduce prior-data fitted networks (PFNs) for forecasting, and then we introduce our ForecastPFN model. PFNs were first introduced by Hollmann et al. (2022); Müller et al. (2022) for classification and by Adriaensen et al. (2022) for learning curve extrapolation. We extend the idea of PFNs to time-series forecasting.

We model time series as the combination of an underlying time series hypothesis with the addition of noise. Each hypothesis $\varphi \in \Phi$ is a function $\varphi : \mathbb{N} \to \mathbb{R}$ which generates an *underlying* time series — defined as a time series with seasonal and trend components but without noise. In order to generate a time series from a hypothesis, we add noise. That is, a function φ is used to generate time series $D = \{(t, y_t)\}_{t=1}^{T}$, where $y_t = \varphi(t) \cdot z_t$ and z_t is sampled from a noise distribution with a mean of 1. We define Φ as a set of hypotheses.

The *posterior predictive distribution* (PPD) for a point y is the distribution of its values given time t and dataset D, $p\left(\cdot \mid t, D\right)$, and it is computed for a particular point y by integrating over the set of hypotheses Φ as follows: $p(y \mid t, D) \propto \int_{\Phi} p(y \mid t, \varphi) p(D \mid \varphi) p(\varphi) d\varphi$.

We conduct synthetic prior-fitting as in Müller et al. (2022); Adriaensen et al. (2022), adapted to the time-series forecasting domain. Prior-fitting is the training of a prior-data fitted network (PFN) to approximate the PPD. As in (Hollmann et al., 2022), we use a prior-sampling scheme $p(D)$ which first samples a hypothesis $\varphi \sim p(\varphi)$ and then a dataset $D \sim p(D \mid \varphi)$. In order to train the PFN, we iteratively sample a prior φ and dataset $D = \{(t, y_t)\}_{t=1}^{T}$ using the above method and split the dataset into an input set and a test set, $D_{\text{in}} = \{(t, y_t)\}_{t=1}^{\ell}$ and $D_{\text{test}} = \{(t, y_t)\}_{t=\ell}^{T}$. We use the PFN to predict D_{test} given D_{in} and compute the training loss as the mean squared error for the predicted D_{test}, which trains the PFN to approximate Bayesian inference (Müller et al., 2022).

ForecastPFN. The majority of existing time-series forecasting methods work only for a fixed series length, horizon length, and frequency. In contrast, we design a PFN for forecasting, ForecastPFN, which can handle a wide variety of time series. We use an encoder-based transformer, consisting of a multi-head attention layer and two feedforward layers. This is in stark contrast to prior work on zero-shot forecasting (Oreshkin et al., 2021), which used a residual network.

The transformer accepts data in the form of tokens (t, y_t). The transformer takes in a set of ℓ contiguous tokens, along with a query consisting of a future date without a value, and then it predicts the PPD of this query. This is in contrast to existing transformer models for forecasting, which are

only set up to predict the next N steps in the current sequence. The date, t, is represented in terms of time features corresponding to the year, month, day, day of the week, and day of the year.

When standard forecasting models are trained on the training data, it is typical to apply transforms to put the data in a fixed range. However, ForecastPFN is designed to handle scaling *dynamically*: first, outliers are removed by scaling based on the 99th percentile of the input. Second, to give the model a high dynamic range, the following transform is applied for input x: $\tanh\left(x \cdot 4^{[-4\ldots,4]}\right)$. In other words, this splits the input into multiple channels, sensitive to scales of different order of magnitude, allowing the model to concentrate on different scales in the same time series.

Synthetic Data Distribution. Unlike all prior work in forecasting, our model is not trained on any real-world data; it is trained purely on synthetic data.

We model our synthetic data with the simple premise that time series data have two independent components: underlying (φ) and noise (z_t). Further, we model the underlying time series as being comprised of a seasonal and a trend component. We see these as three independent aspects of time series data and model them as below, where the time series is the product of a trend and seasonality with an additional noise factor. The trend component is made up of a linear and exponential component with coefficients $m_{\text{lin}}, c_{\text{lin}}, m_{\text{exp}}, c_{\text{exp}}$ sampled from normal distributions such that $m_{\text{lin}}, m_{\text{exp}} \sim N(-.01, 0.5), c_{\text{lin}} \sim N(0, 0.01), c_{\text{exp}} \sim N(1, 0.005)$. The seasonal component has a week, month, and year component where each comprises of of coefficients $m_{\text{week}}, m_{\text{month}}, m_{\text{year}}, c_i, d_i$ where the $c_i, d_i \sim N(0, 1/i)$. Finally, the noise in our model is derived from a Weibull distribution and is modeled such that the expected value of the noise model is 1, meaning in expectation the noise does not contribute to the seasonality or trend of the time series. We set $p(\varphi)$ based on the following equation, with random draws to c_i, d_i, and z.

$$
\begin{aligned}
y_t = \varphi(t) \cdot z_t &= \text{trend}(t) \cdot \text{seasonal}(t) \cdot z_t, \text{ where} \\
z_t &= 1 + m_{\text{noise}}\left(z - \bar{z}\right), \text{ where } z \sim \text{Weibull}(1, k),\ \bar{z} = (\ln 2)^{1/k} \\
\text{trend}(t) &= \left(1 + m_{\text{lin}} \cdot t + c_{\text{lin}}\right)\left(m_{\text{exp}} \cdot c_{\text{exp}}^{t}\right) \\
\text{seasonal}(t) &= \text{seasonal}_{\text{week}}(t) \cdot \text{seasonal}_{\text{month}}(t) \cdot \text{seasonal}_{\text{year}}(t), \text{ where} \\
\text{seasonal}_{\text{per}}(t) &= 1 + m_{\text{per}} \sum_{i=1}^{k} \left[c_i \sin\left(2\pi i \frac{t \text{ Mod } p_{\text{per}}}{p_{\text{per}}}\right) + d_i \cos\left(2\pi i \frac{t \text{ Mod } p_{\text{per}}}{p_{\text{per}}}\right)\right], \\
&\text{where per} \in \{\text{week}, \text{month}, \text{year}\} \text{ and } p_{\text{week}} = 7,\ p_{\text{month}} = 30.5,\ p_{\text{year}} = 365.25.
\end{aligned}
$$

4 EXPERIMENTS

We compare ForecastPFN to various forecasting models on different datasets. We compare against two high-performing traditional methods: ARIMA (Box & Pierce, 1970) and Prophet (Taylor & Letham, 2018), and three state-of-the-art transformer-based method: FEDformer (Zhou et al., 2022), Autoformer (Wu et al., 2021), and Informer (Zhou et al., 2021). All experiments were done using *the same* pretrained ForecastPFN model (the one described in the previous section).

We evaluate our method on two datsets. Illness is a dataset [1]containing influenza-like illness patients in the United States. It contains 966 datapoints with 8 dimensions, spaced weekly. Exchange is a currency exchange dataset (Lai et al., 2018) across 8 countries, consisting of 7588 datapoints with 9 features, spaced daily. Training procedures for ForecastPFN can be found in Appendix A.

Experimental setup and results. Following prior work (Zhou et al., 2022; Wu et al., 2021), for the Illness dataset, we fix the input length to be 36, and we predict the next sequences up to length 24, 36, 48, and 60. For the Exchange dataset, we fix the input length to 96, and we predict the next sequences up to length 96, 192, 336, and 720. We focus on the univariate case, leaving the mutlivariate case for future work. Additionally, we scale our datasets using the approach of (Zhou et al., 2022) where all data is scaled to zero mean and unit variance of the training data. The same transformation is then applied to the test data. At inference on the test set, the data may not be scaled such that it has zero mean and unit variance, for example if the test data trends higher or lower than

[1] `https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html`

Table 1: Forecasting results on Illness and Exchange datasets with two budgets, 1 second and 1 minute, for training. A lower MSE indicates better performance. A blue bold is for the best performance on 1 second budget, and black bold is for the best performance on 1 minute budget. Configurations which took twice than their budget to converge are omitted from this table.

Datasets		Illness				Exchange			
Methods	Budget	24	36	48	60	96	192	336	720
ForecastPFN	1s	**1.011**	**0.799**	**0.881**	**1.047**	**0.237**	**0.351**	**0.577**	1.231
	1m	1.011	0.799	0.881	1.047	0.237	**0.351**	**0.577**	1.231
FEDformer-w	1s	1.087	1.137	1.243	1.533	0.331	0.416	0.579	**1.155**
	1m	**0.690**	**0.548**	**0.662**	**0.796**	0.236	0.407	0.637	1.439
Autoformer	1s	2.042	2.291	2.188	2.124	1.697	1.136	1.684	1.951
	1m	0.712	0.590	0.715	0.904	**0.227**	0.664	0.900	1.171
Informer	1s	4.586	4.036	4.177	4.289	1.256	1.092	1.471	1.800
	1m	5.039	4.052	4.210	4.688	0.966	1.577	3.188	1.673
Prophet	1s	3.439	3.384	3.321	3.345	1.303	1.309	1.328	1.379
	1m	3.439	3.384	3.321	3.345	1.303	1.309	1.328	1.379
Arima	1s	–	–	–	–	–	–	–	–
	1m	3.749	3.663	3.643	3.726	1.057	1.053	1.066	**1.077**

the training data. However, if a model expects input to be within the unit interval, we then scale the input sequence of the test sequence to be as such and then inverse the scaling on the prediction.

We compare a low budget (1 second) and moderate budget (1 minute) for training the baseline models. During this budget, we allow for each comparison baseline method to train for that budgeted amount of time on the dataset. Explicitly, we allow a baseline, like Prophet, to fit to the training data for 1 second (or 1 minute) by learning from the training dataset. Training runs which take twice the allocated budget are omitted from our analysis. We expect, and show, that the performance on the resource-constrained, small budget should yield worse performance (higher MSE) than performance when there is a longer training time. This contrasts to ForecastPFN which does not train on the given dataset and so performance on both budgets is the same.

We report mean squared error (MSE) for the evaluation as the mean MSE over five training runs for each configuration. Our main results are presented in Table 1. We see that at the lowest budget (1 second of training time) and for every prediction length (except for one – 720 on the Exchange dataset), ForecastPFN achieves the lowest MSE. This confirms that ForecastPFN's strength rests in the ability to quickly perform well on a wide range of forecasting tasks, without ever seeing any part of the training time series for a given dataset. We also see that with a 1 minute budget, ForecastPFN still achieves competitive results, particularly on the Exchange dataset where it is the top-performing method at prediction length 96, 192, and 336. We also note that the simple Prophet algorithm outperforms all transformer-based methods in the 1 minute setting.

5 Conclusions and Future Work

In this work, we introduced ForecastPFN, the first universal zero-shot model, trained purely on synthetic data. To accomplish this, we designed a novel synthetic time-series distribution and a new dynamic scaling method. Through experiments on multiple datasets, we show that ForecastPFN achieves competitive performance *without ever seeing the training datasets*, compared to popular methods that were trained on the training data.

In the domain of health, we see that ForecastPFN has wide potential to be applied to a variety of applications, particularly in low-resource settings. Given that ForecastPFN can achieve competitive results even without access to the training data, we propose further work in this area to extend its application to more healthcare applications. Desdigning ForecastPFN for multivariate time series, and training with a mix of synthetic and real datasets, are both exciting direction for future work.

ACKNOWLEDGMENTS AND DISCLOSURE OF FUNDING

This work was completed while all authors were employed at Abacus.AI.

REFERENCES

Steven Adriaensen, Herilalaina Rakotoarison, Samuel Müller, and Frank Hutter. Efficient bayesian learning curve extrapolation using prior-data fitted networks. In *Sixth NeurIPS Workshop on Meta-Learning*, 2022.

George EP Box and David A Pierce. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American statistical Association*, 65(332):1509–1526, 1970.

Vinay Kumar Reddy Chimmula and Lei Zhang. Time series forecasting of covid-19 transmission in canada using lstm networks. *Chaos, Solitons & Fractals*, 135:109864, 2020.

Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):96, 2019.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*, 2022.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Bjoern Krollner, Bruce J Vanstone, Gavin R Finnie, et al. Financial time series forecasting with machine learning techniques: a survey. In *ESANN*, 2010.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Amal Mahmoud and Ammar Mohammed. A survey on deep learning for time-series forecasting. *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges*, pp. 365–392, 2021.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. Meta-learning framework with applications to zero-shot time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2020.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90:106181, 2020.

Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

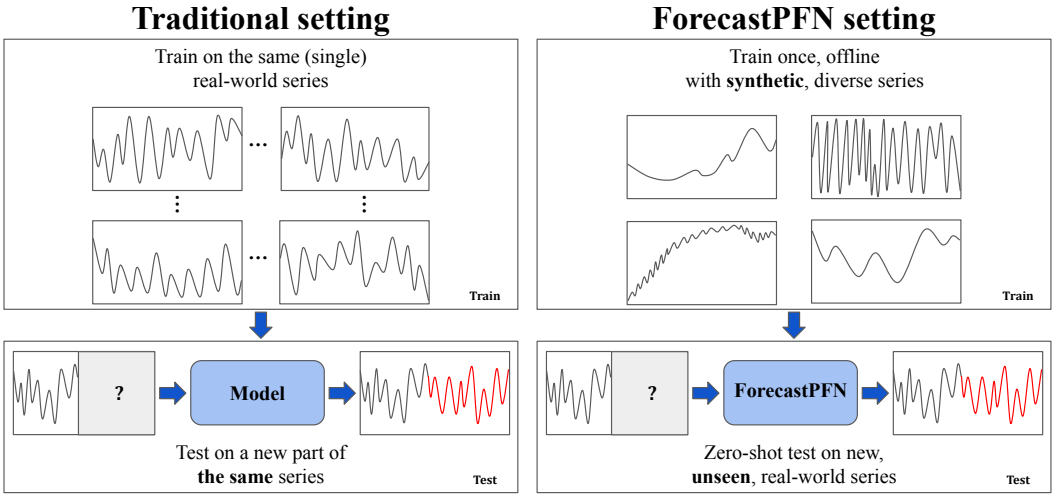

Figure 1: Left: standard setting for time-series forecasting. Right: ForecastPFN setting; zero-shot forecasting by training on synthetic time series.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

A Training Procedure for ForecastPFN

We train ForecastPFN with a novel prior for time-series forecasting, described in the next section. We generate 100 000 synthetic daily, weekly, and monthly series, each of length 200, and we use a sliding window of size 100 to obtain 101 prediction tasks per series. We sample 1024 tasks in a training step, and there are 1000 training steps in an epoch. We trained the transformer for 300 epochs on a single Tesla V100 16GB GPU, which took 16 hours. Note that this training is done offline, and only once. The trained ForecastPFN was used for all experiments in our paper and can be used in the future on any new time series.