Probabilistic Forecasting via Modern Hopfield Networks

Kashif Rasul
Morgan Stanley
rasul.kashif@morganstanley.com

Pablo Vicente
Morgan Stanley
pablo.vicente@morganstanley.com

Anderson Schneider
Morgan Stanley
anderson.schneider@morganstanley.com

Alexander März
alex.maerz@gmx.net

Abstract

Hopfield networks, originally introduced as associative memory models, have shown promise in pattern recognition, optimization problems, and tabular datasets. However, their application to time series data has been limited. We introduce a temporal version that leverages the associative memory properties of the Hopfield architecture while accounting for temporal dependencies present in time series data. Our results suggest that the proposed model demonstrates competitive performance compared to state-of-the-art probabilistic forecasting models.

1 Introduction

Time series forecasting is a challenging machine learning task given the need to model potentially intricate, non-linear temporal patterns across extended time horizons. Over time, time series models have evolved tremendously from classical statistical methods such as ARIMA [3] to deep learning-based approaches [2]. Such neural forecasting methods are generally comprised of two components: one that learns representations from the history of a time series and a second that, based on the representations, learns an emission model, either point-based or probabilistic. Older architectures for learning such historical representations include Recurrent Neural Networks (RNNs) such as LSTM [10] or GRU [6] and Causal Convolutional Networks [5]. Recently, methods based on Transformers [25] have presented state-of-the-art performance, surpassing both classical autoregressive approaches as well its neural counterparts. This is in large part due to the Transformer's strong inductive bias [26], which allows it to look back over the entire context history of a time series, without suffering from limited temporal field or issues of forgetting [19].

In this work, we investigate an alternative approach for neural time series modeling. Motivated by the observation that time series data typically display seasonal or trend behavior, incorporating Associative Memories [15], which can effectively link sets of past features to generate robust representations for forecasting, can provide a beneficial inductive bias for modeling time series data. More specifically, we explore the potential of modern Hopfield networks [21] in capturing past temporal information for probabilistic forecasting tasks. It is worth noting that the attention mechanism used in Transformers is a specific instance of these models.

2 Probabilistic Forecasting

The task of probabilistic time series forecasting in the *univariate* setting consists of training on a dataset of $D \geq 1$ time series $\mathcal{D}_{\text{train}} = \{x^i_{1:T^i}\}$ where $i \in \{1, \ldots, D\}$ and at each time point t, we have $x_t^i \in \mathbb{R}$ or in $\mathbb{N}$. We are tasked with forecasting the potentially complex distribution of the next $P > 1$ time steps into the future, and we are given a test set $\mathcal{D}_{\text{test}} = \{x^i_{T^i+1:T^i+P}\}$. Each time index t is in reality a date-time value that increments regularly based on the frequency of the dataset in question and the last training point T^i for each time series may or may not be the same date-time. Autoregressive models like [9, 23] estimate the prediction density by decomposing the joint distribution of all P points via the chain rule of probability as:

$$p_{\mathcal{X}}(x^i_{T^i+1:T^i+P}) \approx \Pi_{t=1}^{P} p(x^i_{T^i+t} | x^i_{1:T^i-1+t}, \mathbf{c}^i_{1:T^i+P}; \theta),$$

parameterized by some model with trained weights θ. This requires the next time point being conditioned on *all* the past and covariates vectors $\mathbf{c}_t^i$ (detailed in Section A.3). RNN-based models like `DeepAR` [23] typically resort to the seq-to-seq paradigm [24] and considers some context window of *fixed-size* C sampled randomly from the complete time series history to learn historical representation, up to a time point, and use this representation in the decoder to learn the distribution of the subsequent P time points:

$$\Pi_{t=1}^{P} p(x^i_{C+t} | x^i_{1:C-1+t}, \mathbf{c}^i_{1:C+P}; \theta).$$

Convolutional models, though not autoregressive, can similarly learn some representation from their temporal receptive fields and the distribution for a fixed P. Finally, attention-based models allow representation to be learned over the entire context window, which due to the quadratic complexity of the attention layer, can restrict the size of C. In the case of encoder-decoder architecture, N layers of an encoder can be used to learn a context-window sized sequence of `d_model` sized representations denoted by:

$$\{\mathbf{h}_t\}_{t=1}^{C-1} = \mathrm{Enc}_N \circ \cdots \circ \mathrm{Enc}_1(\{\texttt{proj}(\texttt{concat}(x_t^i, \mathbf{c}^i_{t+1}); \theta_p)\}_{t=1}^{C-1}; \theta_1).$$

Afterward, M layers of a *causal* or masked decoder can be used to model the distribution of the subsequent P future time points conditioned on the encoding representations and covariates as:

$$\Pi_{t=C}^{C+P-1} p(x^i_{t+1} | x^i_{t:C}, \mathbf{c}^i_{t+1:C+1}, \mathbf{h}_1, \ldots, \mathbf{h}_{C-1}; \theta).$$

For example, if we assume the data comes from a Gaussian then the outputs from the M decoders' representation can be passed to a layer (Distribution Head) that returns appropriately signed parameters of a Gaussian whose log-likelihood, given by

$$\sum_{t=C}^{C+P-1} \log p_{\mathcal{N}}(x^i_{t+1} | x^i_{t:C}, \mathbf{c}^i_{t+1:C+1}, \mathbf{h}_1, \ldots, \mathbf{h}_{C-1}; \theta),$$

can be maximized for all i and t from the $\mathcal{D}_{\text{train}}$ using stochastic gradient descent (SGD). During inference time we pass the very last context length window for each time series i to the encoder and the resulting predicted distribution is used to sample predictions for the next time point which together with the future covariates are autoregressive passed to the decoder P times (greedy decoding). In this manner, we can obtain our desired empirical probability intervals of interest and evaluation metrics with respect to the unseen $\mathcal{D}_{\text{test}}$.

3 Mordern Hopfield Networks

Associative Memory or Hopfield Networks [11] are networks with memory storage capacity which is important for machine learning applications. Modern Hopfield Networks (MHN) or Dense Associative Memories [7] uses an exponential pattern interaction function giving them exponential storage capacity in the latent dimension. [21] introduces a new energy function to update associate inputs to their most similar patterns which can be given or learned. These networks generalize the energy functional to continuous value patterns and maintain the high storage capacity. The attention mechanism is a special case of these models where the function is given by the softmax. Due to their continuous nature, these layers are differentiable and can be integrated into deep learning architectures, endowing them with associative memories. This extension allows us to learn associations to memories from temporal vectors given by a sequence of $\{\texttt{concat}(x_t^i, \mathbf{c}_{t+1}^i)\}_1^{T^i}$ as Dense Associative Memory models are naturally designed for completing patterns. The key difference between transformers is that Associative Memories are recurrent networks whereas transformers are not typically considered a dynamical system.

4 `HopfieldTS`

The `HopfieldTS` consists of an encoder-decoder architecture as depicted in Figure 1, where N unmasked `Hopfield` [21] encoders are fed the context length sequence of projected inputs and the subsequent prediction length window is passed to M *masked* or causal `Hopfield` decoders. The loss with respect to the specified log-likelihood is calculated via the by-one-shifted prediction window. This setup has the advantage that at inference time we only need to run the encoder over some long context length sequence once and run the decoder at most P times, which is typically a smaller number. The disadvantage of this approach is that we are throwing away potential learning signals from the context-length window of training data and back-propagate via a prediction length window for each SGD step.

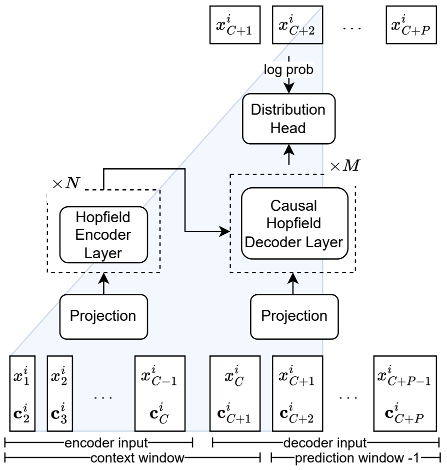

Figure 1: A depiction of the `HopfieldTS` model where during training a context-sized sequence of inputs is projected and passed to N encoder `Hopfield` layers and the subsequent prediction length input similarly projected and passed to M *causal* decoder `Hopfield` layers.

5 Related Works

The use of MHNs in sequential modeling has been recently explored in the General Sequential Episodic Memory Model (GSEMM) [13] that encodes memories and their sequential relationships by leveraging an adiabatic approach on the energy landscape of Hopfield Networks. The Long Sequence Hopfield Memory [4] works directly with the update rule for the state of the Hopfield network allowing for an analytical derivation for the sequential capacity of the model and showcasing its ability to store and recall sequences of correlated patterns. The recent `HopCPT` [1] leverages a Hopfield network to forecast using the Conformal Prediction approach of quantifying uncertainty instead of some chosen parametric distribution.

6 Experiments

We use the following open datasets: `Exchange` [16], `Solar` [16], `Electricity`[1], `Traffic`[2], `Taxi`[3], and `Wikipedia`[4] preprocessed exactly as in [22] for our probabilistic forecasting experiments. The dataset properties are listed in Table 2. For real-valued datasets we use a Student-T head, while for integer-valued datasets we resort to learning the parameters of a Negative-Binomial distribution. We compare against the following deep learning-based *univariate probabilistic* models: `DeepAR` [23], `Transformer`, `TFT` [17] and `Informer` [26]. We report the mean scale interval score [8] (MSIS[5]) for a 95% prediction interval, the 50th and 90th quantile percentile loss (QL50 and QL90, respectively), as well as the Continuous Ranked Probability Score (CRPS) [8] score. The CRPS is a *proper* scoring rule. The point-forecasting performance of models is measured by the normalized root mean square error (NRMSE), the mean absolute scaled error (MASE) [12], and the symmetric mean absolute percentage error (sMAPE) [20]. For pointwise metrics, we use sample *medians* except for NRMSE, where we report using sample *means*. Table 1 shows the results.

Table 1: Forecasting-error metrics (lower is better) with Student-T (`-t`) or Negative Binomial (`-nb`) on the open datasets. Best metrics are in bold.

Dataset	Method	CRPS	QL50	QL90	MSIS	NRMSE	sMAPE	MASE
	`DeepAR-t`	0.0504	0.0631	**0.0338**	**5.8148**	**0.5350**	**0.1007**	**0.6945**
	`Transformer-t`	0.0554	0.0690	0.0398	6.2246	0.6470	0.1076	0.7460
`Electricity`	`TFT-t`	0.0661	0.0819	0.0473	8.2548	0.6763	0.1223	0.9484
	`Informer-t`	**0.0503**	**0.0629**	0.0341	5.9355	0.5684	0.1036	0.7020
	`HopfieldTS-t`	**0.0503**	0.0632	0.0336	6.0535	0.5622	0.1039	0.7152
	`DeepAR-t`	0.0183	0.0222	0.0095	53.0098	0.0325	0.0230	4.0972
	`Transformer-t`	0.0345	0.0410	0.0150	84.2023	0.0556	0.0383	6.5162
`Exchange`	`TFT-t`	0.0217	0.0256	0.0087	79.4968	0.0374	0.0281	4.9519
	`Informer-t`	0.0082	0.0103	**0.0040**	**14.0588**	0.0168	**0.0126**	**1.7254**
	`HopfieldTS-t`	**0.0078**	**0.0100**	0.0041	14.3792	**0.0152**	0.0129	1.8572
	`DeepAR-t`	0.4386	0.5340	0.3139	12.4606	**1.0897**	1.3929	**1.2536**
	`Transformer-t`	**0.4261**	**0.5336**	0.3235	9.8072	1.1023	1.4029	1.2545
`Solar`	`TFT-t`	0.4478	0.5623	0.3590	10.9283	1.1641	1.4342	1.3216
	`Informer-t`	0.4358	0.5465	0.2399	9.0558	1.1297	1.3900	1.2855
	`HopfieldTS-t`	0.4293	0.5410	**0.2314**	**8.2744**	1.1161	**1.3842**	1.2706
	`DeepAR-nb`	**0.2875**	**0.3641**	**0.1944**	**5.4164**	**0.5787**	**0.5706**	**0.7436**
	`Transformer-nb`	0.3037	0.3820	0.2166	6.0592	0.6187	0.5913	0.7761
`Taxi`	`TFT-nb`	0.2969	0.3755	0.2020	5.6185	0.6086	0.5759	0.7644
	`Informer-nb`	0.3046	0.3850	0.2059	5.8538	0.6217	0.5860	0.7822
	`HopfieldTS-nb`	0.3021	0.3815	0.2079	5.8184	0.6135	0.5910	0.7762
	`DeepAR-t`	**0.1194**	**0.1423**	**0.0987**	**6.7620**	**0.4121**	**0.1403**	**0.5428**
	`Transformer-t`	0.1230	0.1471	0.1009	6.7683	0.4242	0.1433	0.5577
`Traffic`	`TFT-t`	0.1320	0.1547	0.1049	7.5259	0.4531	0.1518	0.5910
	`Informer-t`	0.1216	0.1430	0.0996	7.0771	0.4244	0.1571	0.5450
	`HopfieldTS-t`	0.1274	0.1494	0.1020	7.3987	0.4334	0.1713	0.5710
	`DeepAR-nb`	0.4688	0.4532	0.7826	59.5015	3.1795	0.3385	2.3928
	`Transformer-nb`	**0.3204**	**0.3563**	**0.3494**	**26.4558**	**2.1732**	**0.3060**	**1.7312**
`Wikipedia`	`TFT-nb`	0.4163	0.4397	0.5899	43.9931	2.7214	0.3684	2.1223
	`Informer-nb`	0.4740	0.5467	0.5648	50.2443	2.4364	0.6929	2.7806
	`HopfieldTS-nb`	0.4181	0.5228	0.3596	26.6816	2.2919	0.6316	2.6045

From Table 1 we see that model performance varies with data characteristics, in particular with the size D and frequency. Notably, our Hopfield-TS model stands out for smaller datasets, as suggested by its performance on the `Electricity`, `Exchange`, and `Solar` datasets. On the other hand, the vanilla encoder-decoder Transformer [25] shows superior results on the largest dataset, `Wikipedia`. Meanwhile, `DeepAR` [23] performs well for medium to large datasets like `Traffic` and `Taxi`.

During our experiments, we further investigated the role of the β parameter within the Hopfield model. Our results indicate a relationship between the dataset's size and the influence of the β parameters on both the encoder and decoder's performance. Specifically, when β is set to 1, patterns are predominantly averaged, leading to a decrease in the model's effectiveness on larger datasets. Conversely, our Hopfield model demonstrates enhanced performance on larger datasets as β increases.

[1]`https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014`

[2]`https://github.com/laiguokun/multivariate-time-series-data#traffic-usage`

[3]`https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page`

[4]`https://github.com/mbohlkeschneider/gluon-ts/tree/mv_release/datasets`

[5]`http://www.unic.ac.cy/test/wp-content/uploads/sites/2/2018/09/M4-Competitors-Guide.pdf`

On smaller datasets, such as `Solar` and `Electricity`, we observe worse CRPS as we increase β. Similarly, for datasets with similar temporal patterns like `Traffic`, we observe the performance decreasing for larger β. As can be seen from Figure 2 we get SOTA ÇRPS metrics with respect to the methods compared, on all the datasets except `Taxi` by tuning the β hyperparameter.

7 Summary

Our temporal Hopfield architecture offers a novel approach to time series modeling, leveraging the associative memory properties of the Hopfield network while accounting for temporal dependencies. Our extension opens up new avenues for the application of Hopfield-like networks in time series analysis, forecasting, and anomaly detection. Future work will explore the scalability of our approach and its applicability to multivariate time series data.

References

[1] Andreas Auer, Martin Gauch, Daniel Klotz, and Sepp Hochreiter. Conformal prediction for time series with modern hopfield networks, 2023.

[2] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, François-Xavier Aubet, Laurent Callot, and Tim Januschowski. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Comput. Surv.*, apr 2022.

[3] George.E.P. Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1976.

[4] Hamza Tahir Chaudhry, Jacob A. Zavatone-Veth, Dmitry Krotov, and Cengiz Pehlevan. Long sequence hopfield memory, 2023.

[5] Yitian Chen, Yanfei Kang, Yixiong Chen, and Zizhuo Wang. Probabilistic forecasting with temporal convolutional neural network. *Neurocomputing*, 399:491–501, 2020.

[6] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[7] Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, May 2017.

[8] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[9] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[10] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997.

[11] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[12] Rob J. Hyndman and Anne B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.

[13] Arjun Karuvally, Terrence Sejnowski, and Hava T Siegelmann. General sequential episodic memory model. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15900–15910. PMLR, 23–29 Jul 2023.

[14] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022.

[15] Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. 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.

[16] 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*, SIGIR '18, pages 95–104, New York, NY, USA, 2018. ACM.

[17] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[18] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-Stationary Transformers: Exploring the stationarity in time series forecasting. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 9881–9893. Curran Associates, Inc., 2022.

[19] Shivangi Mahto, Javier Turek, Vy Vo, and Alexander Huth. Multi-timescale representation learning in lstm language models. In *International Conference on Learning Representations, (ICLR)*, 2021.

[20] Spyros Makridakis. Accuracy measures: theoretical and practical concerns. *International Journal of Forecasting*, 9(4):527–529, 1993.

[21] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2021.

[22] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank Gaussian Copula Processes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 6824–6834. Curran Associates, Inc., 2019.

[23] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2019.

[24] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc., 2014.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[26] 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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, May 2021.

A Experimental Setup

This appendix provides additional details on the models, datasets, covariates, as well as hyper-parameters used during our experiments.

A.1 Models

We compare against the following deep learning-based *univariate probabilistic* models:

- `DeepAR` [23]: an RNN-based probabilistic model which learns the parameters of some chosen distribution for the next time point;
- `Transformer` [25]: the vanilla encoder-decoder transformer;
- `TFT` [17]: an auto-regressive attention based Seq-to-Seq model with variable selection network for selecting relevant inputs;
- `Informer` [26]: an efficient transformer and full horizon predictor model;

A.2 Dataset

We evaluate the different models on the following commonly used datasets:

- `Electricity`: hourly time series of the electricity consumption of 371 customers.
- `Exchange`: daily exchange rate between 8 currencies.
- `Solar`: hourly photo-voltaic production of 137 stations in Alabama State.
- `Taxi`: spatio-temporal traffic time series of New York taxi rides taken at 1,214 locations every 30 minutes.
- `Traffic`: hourly occupancy rate of 862 San Francisco car lanes.
- `Wikipedia`: daily page views of 9,535 Wikipedia pages.

Each dataset is divided into training and test sets, where rolling windows are employed for the evaluation on the test sets. Table 2 gives a description of the characteristics of each dataset.

Table 2: Dataset Description.

Dataset	D	Domain.	Frequency	Time Step	Prediction Length
`Electricity`	321	$\mathbb{R}^{\geq 0}$	hour	$15,782$	24
`Exchange`	8	$\mathbb{R}^{\geq 0}$	day	$6,071$	30
`Solar`	137	$\mathbb{R}^{\geq 0}$	hour	$7,009$	24
`Taxi`	$1,214$	$\mathbb{N}^{\geq 0}$	30-min	$1,488$	24
`Traffic`	862	$(0,1)$	hour	$14,036$	24
`Wikipedia`	$9,535$	$\mathbb{N}^{\geq 0}$	day	762	30

A.3 Covariates

We encode covariates via date-time features with regards to a time series' frequency, e.g. for a particular time point t of the time series i, we can create hour-of-day, day-of-week, week-of-month, etc. features as a vector, which we denote by $\mathbf{c}_t^i$. The covariates have to be known for the time points we wish to predict. Additional covariates can be constructed by embedding the identity i of each time series in a dataset via embedding layers, as done in the `DeepAR` method. Univariate time series signals can also be vectorized via lag indices designed for the frequency of the data. For example for hourly time series, the last $x_{t-24}, x_{t-24\times 7}, \ldots$ time steps can be copied into the covariate vector $\mathbf{c}_t$. As time series data can have arbitrary magnitude we can normalize the data based on the mean and standard deviation of the context window and pass these statistics as static covariates. During training or inference, we can either transform the distribution or samples back to the original scale as proposed in [23]. Note that this technique predates the recent related `RevIN` [14] method and Non-stationary Transformers [18]. *All* the methods considered receive the same covariates as input.

A.4 Hyperparameters

We initialize the different models we consider so that they have approximately the same number of trainable parameters of around $63K$. The main hyperparameters are listed in Table 3.

Table 3: Hyperparmeters used in the experiments and their descriptions.

Parameter	Value	Description
`d_model`	64	The latent dim of the Transformer/RNN vector
`context_len`	$2 * P$	The context length is twice the size prediction length
N	1	The number of encoder layers
M	1	The number of decoder layers
`dim_ff`	32	The linear layers hidden size
`emb_dim`	4	Size of the time series i embedding vector
β_{enc}	1	The β parameter of `Hopfield` encoder's metastable states
β_{dec}	1	The β parameter of the `Hopfield` decoder's metastable states
`batch_size`	32	The SGD batch size
`num_batches_per_epoch`	100	The number of batches that constitute an epoch
`max_epochs`	100	The maximum number of epochs

B Ablation

The β parameters of the `Hopfield` layers control the size of the metastable states patterns can converge to. In other words, it controls how many patterns are averaged. As β increases the more pattern clusters there are. We check the test-set performance of `HopfieldTS` as we increase the β for the encoder and decoder and plot the resulting test metrics together with an indication of the best CRPS metric:

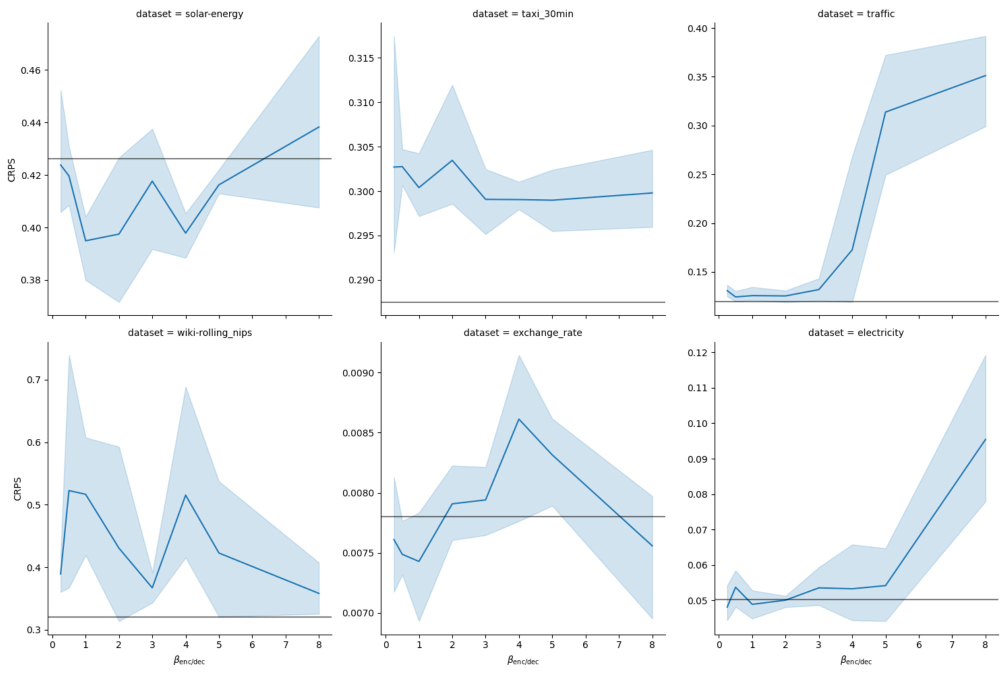

Figure 2: CRPS on test for different datasets according to β with the best CRPS metric (where lower is better) denoted by the horizontal line.