State-Space Models for Tabular Prior-Data Fitted Networks

Felix Koch [1] **Marcel Wever** [2] **Fabian Raisch** [3] **Benjamin Tischler** [1]

Abstract

Recent advancements in foundation models for tabular data, such as TabPFN, demonstrated that pretrained Transformer architectures can approximate Bayesian inference with high predictive performance. However, Transformers suffer from quadratic complexity with respect to sequence length, motivating the exploration of more efficient sequence models. In this work, we investigate the potential of using Hydra, a bidirectional linear-time structured state space model (SSM), as an alternative to Transformers in TabPFN. A key challenge lies in SSM's inherent sensitivity to the order of input tokens – an undesirable property for tabular datasets where the row order is semantically meaningless. We investigate to what extent a bidirectional approach can preserve efficiency and enable symmetric context aggregation. Our experiments show that this approach reduces the order-dependence, achieving predictive performance competitive to the original TabPFN model.

1. Introduction

Recently, foundation models have shown remarkable performance in tabular classification tasks, particularly in few-shot and small-data regimes (Hollmann et al., 2023; Zeng et al., 2024; Thielmann et al., 2024). A prominent example is the Tabular Prior-Data Fitted Network (TabPFN) (Hollmann et al., 2023; 2025), which employs a pretrained Transformer (Vaswani et al., 2017) to perform in-context probabilistic inference over labeled tabular datasets. By training on a large corpus of synthetic tasks generated from structural priors, TabPFN can deliver highly accurate predictions in milliseconds without the need for gradient-based adaptation or fine-tuning.

[1]University of Applied Sciences Rosenheim, Rosenheim, Germany [2]L3S Research Center, Leibniz University Hannover, Hannover, Germany [3]Technical University of Munich, Munich, Germany. Correspondence to: Felix Koch <felix.koch@th-rosenheim.de>, Marcel Wever <m.wever@ai.uni-hannover.de>.

Proceedings of the 1st ICML Workshop on Foundation Models for Structured Data, Vancouver, Canada. 2025. Copyright 2025 by the author(s).

Despite its success, TabPFN inherits the Transformer's computational inefficiencies. The attention mechanism in Transformers has quadratic complexity with respect to the input sequence limiting its scalability. Mamba (Gu & Dao, 2023) presents an alternative that addresses this problem. Mamba is a recently introduced architecture based on structured state space models (SSMs; Gu et al. (2022)). It achieves linear-time sequence processing while retaining the expressivity of the attention mechanism. It has been shown to match or exceed Transformer performance in long-sequence modeling tasks while significantly reducing inference costs (Dao & Gu, 2024). This makes it a promising candidate for scaling TabPFN beyond its limitations on the size of the dataset. However, substituting the Transformer in TabPFN with Mamba introduces a fundamental issue: Mamba operates as a causal model, meaning its representations are inherently order-sensitive. The authors of (Zeng et al., 2024) and (Thielmann et al., 2024) have already noticed this problem. They replaced the Transformer in tabular foundation models with Mamba and observed that Mamba's sensitivity to input order limits its scalability in tabular prediction tasks. This raises the research question of how to mitigate order sensitivity for SSMs to make them suitable for tabular data.

To address this issue, we propose to use Hydra (Hwang et al., 2024), a bidirectional extension for SSMs within TabPFN. Hydra uses *quasiseparable matrix mixers* to enable bidirectional sequence modeling. Therewith, we can retain the efficiency of Mamba while allowing symmetric context aggregation across the input and thereby reducing the role of the input order. Moreover, we propose repeated context permutations (RCP), invoking Hydra with random input permutations to reduce its order-sensitivity further. In an empirical study, we evaluate substituting the Transformer in TabPFN with Mamba and Hydra. Our experimental results show that the Hydra-based TabPFN significantly reduces computational and memory complexity, allowing larger inputs while retaining predictive performance similar to the Transformer-based version. We further show that RCPs improve accuracy and align predicted distributions across permutations. This paper showcases that bidirectional SSMs can also be seen as an alternative to reduce the quadratic complexity alongside approaches with FlashAttention (Dao, 2024) in (Hollmann et al., 2025) or Linear Attention (Katharopoulos et al., 2020) in (Zeng et al., 2024).

2. Prior-Data Fitted Networks

Prior-Data Fitted Networks (PFNs, Müller et al. (2022)) are a class of models trained on synthetic tasks sampled from a predefined prior distribution over learning problems called the prior. By learning to predict over such a distribution, these networks approximate Bayesian inference without explicit posterior computation at inference time. Transformers (Vaswani et al., 2017) have several benefits to model structured data, making them a commonly used architecture for PFNs. Their core component is the self-attention mechanism, which enables the model to compute interactions between all pairs of inputs in a sequence.

The Tabular Prior-Data Fitted Network (TabPFN) is a Transformer-based instance of PFNs that is especially designed for tabular classification tasks. TabPFN receives an entire dataset as input and classifies it based on the pretrained Transformer model. The model was trained on millions of synthetic classification tasks generated from simple structural causal models and Bayesian neural networks. The offline meta-training enables predictions on a new dataset in a single forward pass, given both the training and test data as input. This produces calibrated outputs within milliseconds, without requiring gradient-based adaptation. However, this performance can only be guaranteed for small datasets due to the limitations of the Transformer architecture. The self-attention operation leads to a quadratic complexity with respect to the input length. The input length corresponds to the number of rows in a tabular setting.

3. SSMs for Tabular Prior-Fitted Networks

SSMs have emerged as efficient alternatives to Transformers for sequence modeling, especially in tasks involving long-range dependencies. These models are inspired by classical state-space models from control theory and signal processing, where sequences are represented via recurrent updates to (latent) state variables (Gu et al., 2021).

3.1. Mamba

One prominent SSM is Mamba (Gu & Dao, 2023), which introduces a selective mechanism to update state variables. Thereby, it achieves linear-time inference (cf. Figure 1) while matching or exceeding Transformer performance in several domains (Gu & Dao, 2023). For this reason, we are testing Mamba as an alternative for the Transformer within TabPFN. For the implementation of Mamba, we interpret the rows of the table as a sequence of classification examples.

Like other SSMs, Mamba can be expressed as *semiseparable matrix multipliers* (Dao & Gu, 2024), enabling a hardware-efficient implementation. However, most SSMs, including Mamba, are causal by design: they process sequences in a forward (autoregressive) manner.

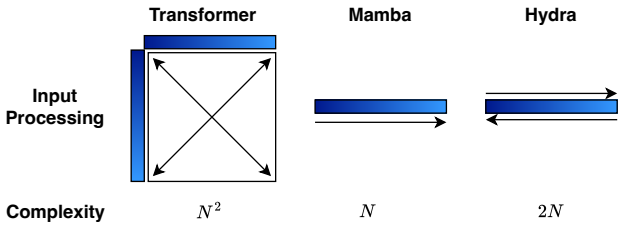


Figure 1. Comparison of different backbone architectures for TabPFN based on their input processing. Transformers have quadratic complexity, whereas Mamba offers linear-time processing, and Hydra adds bidirectional capabilities.

This may present a problem in tabular data, where the order of rows has no meaning and thus there exists no natural order. Also, Mamba scans in an unidirectional way, making it possible to draw causal connections from only one direction. This may lower in-context learning capabilities for tabular tasks, which was also concluded by (Zeng et al., 2024).

3.2. Hydra

To address the issues of Mamba, we additionally consider Hydra (Hwang et al., 2024) as another alternative for the Transformer. Hydra is a bidirectional extension of Mamba that enables efficient processing of sequences using *quasiseparable matrix mixers*. Hydra retains the linear-time benefits of Mamba while allowing the model to be more agnostic of the order, which is essential for principled inference on tabular datasets. An overview of the different backbone architectures is provided in Figure 1.

To replace the Transformer in TabPFN with Hydra, we make the following modifications:

Backbone Replacement The Transformer encoder is replaced with a stack of Hydra layers. Each layer consists of a bidirectional state-space mixing. This is followed by feed-forward transformations, closely mirroring the Transformer block structure but with linear-time complexity.

Embedding Format We retain the original data embedding strategy, where each input is represented as a concatenation of feature values and a class label. As in TabPFN, inference involves marginalizing over all possible label assignments for unlabeled data.

Parameter Compatibility Due to architectural differences, Hydra-based TabPFN requires retraining on the synthetic task distribution used for the original TabPFN. However, the training pipeline remains unchanged aside from the swap of the backbone.

Algorithm 1 RCP for Tabular PFN

Input: Number of permutations r, context D, x_{test}
Output: Predicted class values
Initialize an empty list: $outputs \leftarrow [\,]$
for $i = 1$ **to** r **do**
  Shuffle rows of D: $D_p \leftarrow \text{shuffle}(D)$
  Concatenate x_{test} to D_p: $D_{\text{in}} \leftarrow D_p \cup x_{\text{test}}$
  Predict: $outputs[i] \leftarrow \text{PFN.predict}(D_{\text{in}})$
end for
return average of $outputs$

3.3. Repeated Context Permutations

SSMs are inherently dependent on the input sequence order, distinguishing them from Transformers, which are intrinsically positionally invariant. This limits the application of SSMs to tabular data, mainly because all training examples in one inference are not considered equally important. To decrease the sensitivity of SSM-based PFNs on the sequence order of the rows of a given dataset, we integrate row-wise *repeated context permutations* (RCP) into the inference. This approach reduces the dependency of the results on row ordering by predicting r times with a shuffled context and averaging the predicted probability distributions. RCPs linearly increase the inference time by a factor of r (see Algorithm 1). To assess the benefit of RCPs for SSM-based TabPFN, we include an ablation study in Section 4.

4. Evaluation

In this section, we compare our implementation of Mamba and Hydra with the state-of-the-art Transformer model.

4.1. Experiment Setup

For evaluation, we employ publicly available datasets from OpenML (Vanschoren et al., 2014). All models are evaluated on the multiclass classification datasets from the OpenML CC-18 benchmarking suite (Bischl et al., 2019). More specifically, these datasets were filtered to contain ≤ 2000 rows, ≤ 100 features, and ≤ 10 classes beforehand to meet the constraints of TabPFN. This results in a total of 30 analyzed datasets. Each dataset is randomly split into training and test sets 16 times to account for variance in observations. We evaluated each model in the test set and reported the mean and standard error across all splits. For more details on our experiments, we refer to Appendix A. The code is publicly available on GitHub.

As stated in Section 3.3, SSMs are inherently dependent on the order of the input sequence. To measure the influence of this order on the predictions, we employ the KL divergence:

$$D_{KL}(P||Q) = \sum_i P_i log(\frac{P_i}{Q_i}) \ .$$

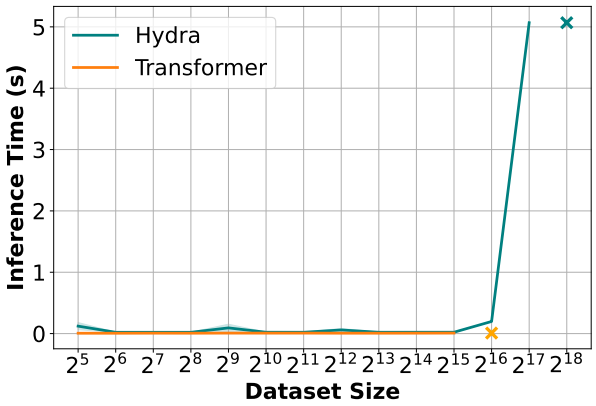


Figure 2. Mean inference time (standard error as shaded areas) for increasing input dataset sizes, ranging from 2^5 to 2^{17}, comparing Hydra- and Transformer-based TabPFN, on an H100 with 80GB VRAM. The Transformer allows for input sizes of up to 2^{15}, and Hydra up to 2^{17}

We compare two predictions with shuffled contexts and quantify the impact of the row ordering on the resulting output. The KL divergence then provides an indication of the dependency. A higher value for the KL divergence indicates a higher entropy and thus a greater dependence on the order of the rows in the table for the model.

4.2. Results

In the following, we will present three experiments that compare Mamba and Hydra as replacements for the Transformer within TabPFN. First, we will asses the inference times, as these are the main motivation for using SSMs versus the Transformer model. Second, we will evaluate the performance of the models on the datasets. This provides a raw understanding of the performance loss when using SSMs compared to the Transformer. Lastly, we will evaluate the dependency on the order of the input sequence from reasons stated in Sections 3.3 and 4.1.

First, we compare inference times of Transformer-based TabPFN and its Hydra-backed variant across varying input sizes. Mamba exhibited similar behavior to Hydra; therefore, its results are omitted for clarity. As shown in Figure 2, the mean inference time grows with dataset size. For the Transformer, inference fails at 2^{16} rows, as the space requirements for the quadratic self-attention matrix exceed the available 80GB VRAM. Hydra only fails at 2^{18} rows, i.e., a two orders of magnitude larger dataset. However, it *does not* exceed the available memory yet, but PyTorch's 32-bit indexing limit, which is a software limitation but not a hardware one. Note that the scalability of Transformer can be improved through, e.g., FlashAttention, reducing space requirements by a constant factor of 20 (Dao et al., 2022).

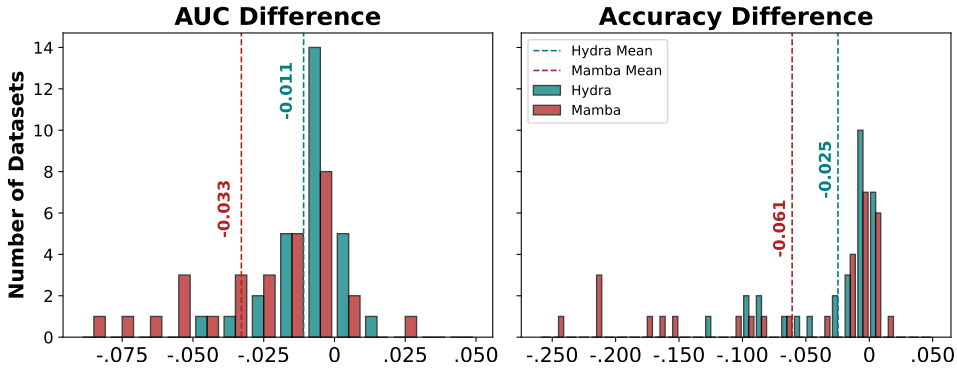


Figure 3. Comparison of the performance of Mamba and Hydra models with the Transformer as a baseline

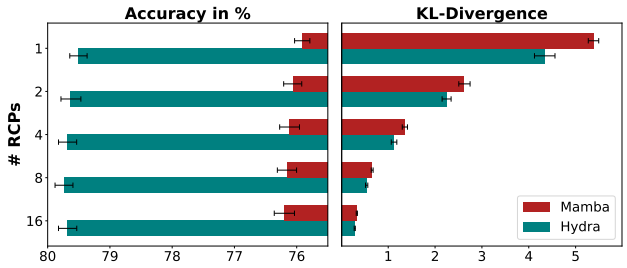


Figure 4. Effect of Repeated Context Permutations on the KL divergence and the accuracy in TabPFN with Hydra and Mamba as a backbone model.

Second, we compare the performance of Mamba and Hydra in terms of accuracy and AUC OvO in Figure 3 (detailed results in Appendix B.1). Here, we calculate the performance difference for each dataset between the corresponding SSM and the Transformer. The results show that Mamba exhibits higher variance than Hydra in both AUC and accuracy. On average, Hydra achieves 3.6% higher accuracy than Mamba, showing a clear advantage of the bidirectional approach over the unidirectional approach. Furthermore, Hydra achieves an average difference of 1.1%, indicating that its performance is quite close to that of the Transformer. Notably, Hydra also attains the best performance on some of the datasets.

Third, we investigate the influence of the number of RCPs on our results. Figure 4 presents the effects of the RCPs on the KL divergence and the quality of the prediction in terms of accuracy. In addition to the average accuracy, 95% confidence intervals were added to show the variance over the splits (cf. 4.1). As can be seen, order sensitivity is decreased as the number of RCPs increases. Also, with an increasing number of RCPs, the accuracy increases to a certain extent, which can be attributed to outliers based on a disadvantageous ordering of table rows being averaged out. This effect on the accuracy is comparably small compared to the relatively large variance. We skip AUC here because no significant improvements can be observed for this metric.

5. Conclusion

This paper introduced the use of State-Space Models for tabular foundation models. Mamba and Hydra are analyzed as Tabular Prior-Data Fitted Networks and compared to the state of the art, i.e., the Transformer architecture. The results show that our approach can deal with larger input sizes due to its linear complexity as opposed to the quadratic complexity of Transformers while maintaining competitive performance. In contrast to other approaches that aimed to enable large table analysis for tabular PFNs, our method directly addresses the quadratic complexity of the underlying Transformer architecture, thereby targeting the root cause of scalability limitations rather than just fixing symptoms.

Comparative analyses revealed that the bidirectional approach Hydra performed better on average than Mamba, which parsed inputs unidirectionally. Additionally, we found the use of multiple Repeated Context Permutations to be useful. These findings suggest that Hydra-based PFNs, combined with Repeated Context Permutations, hold promise for analyzing large tabular datasets, and we hope they inspire further work in this direction.

For the number of rows in the context, we limited ourselves to 1000, conforming to Hollmann et al. (2023). The most promising use case and future direction is to test SSMs as tabular PFNs for longer contexts (for example, $>$ 10k rows, the current limit for TabPFNv2 in (Hollmann et al., 2025)). In addition, future work may further investigate how to mitigate the impact of the row order on the prediction of an SSM-based tabular PFN. We support the assumption proposed by (Thielmann et al., 2024) that certain row orderings of the context may enhance the performance of SSMs. Moreover, such an optimal ordering may differ for unidirectional and bidirectional SSMs.

References

Bischl, B., Casalicchio, G., Feurer, M., Hutter, F., Lang, M., Mantovani, R., van Rijn, J. N., and Vanschoren, J. Openml benchmarking suites and the openml100. *arXiv:1708.03731v1 [stat.ML]*, 2019.

Dao, T. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations (ICLR'24)*. ICLR, 2024. Published online: iclr.cc.

Dao, T. and Gu, A. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning (ICML'24)*, volume 251 of *Proceedings of Machine Learning Research*. PMLR, 2024.

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. FlashAttention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

Gu, A. and Dao, T. Mamba: Linear time sequence modeling with selective state spaces. *arXiv:2312.00752 [cs.LG]*, 2023.

Gu, A., Johnson, I., Goel, K., Saab, K., Dao, T., Rudra, A., and Ré, C. Combining recurrent, convolutional, and continuous-time models with linear state space layers. In *Advances in Neural Information Processing Systems 34, NeurIPS*, 2021.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations (ICLR'23)*. ICLR, 2023. Published online: iclr.cc.

Hollmann, N., Müller, S., Purucker, L., Krishnakumar, A., Körfer, M., Hoo, S. B., Schirrmeister, R. T., and Hutter, F. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

Hwang, S., Lahoti, A. S., Puduppully, R., Dao, T., and Gu, A. Hydra: Bidirectional state space models through generalized matrix mixers. In *Advances in Neural Information Processing Systems 38, NeurIPS*, 2024.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In Daume III, H. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*, volume 98, pp. 5156–5165. Proceedings of Machine Learning Research, 2020.

Müller, S., Hollmann, N., Arango, S., Grabocka, J., and Hutter, F. Transformers can do Bayesian inference. In *The Tenth International Conference on Learning Representations (ICLR'22)*. ICLR, 2022. Published online: iclr.cc.

Thielmann, A. F., Kumar, M., Weisser, C., Reuter, A., Säfken, B., and Samiee, S. Mambular: A sequential model for tabular deep learning. *Arxiv*, 2024. doi: 10.48550/ARXIV.2408.06291.

Vanschoren, J., van Rijn, J., Bischl, B., and Torgo, L. OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2014.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'17)*. Curran Associates, Inc., 2017.

Zeng, Y., Kang, W., and Mueller, A. C. Tabflex: Scaling tabular learning to millions with linear attention. In *NeurIPS 2024 Third Table Representation Learning Workshop*, 2024.

A. Experimental Setups

This section further describes how the above-mentioned experiments were conducted and which hardware was used.

A.1. Training

Training was done on one Nvidia A40 GPU with 48GB of memory. We trained the models based on the Transformer for 48 hours, Mamba for 52 hours, and Hydra for 134 hours. Each training run independently sampled datasets from the same Prior. Thus, it was assumed that the training environment for all models is similar due to the same underlying distribution.

A.2. Validation

The best model was selected based on the best score on the validation set. For this, we used the same datasets as (Hollmann et al., 2023) listed in Table A.2. We excluded two datasets due to their overlap with the OpenML CC-18 benchmarks used for testing.

Table 1. Validation Set from OpenML

Dataset-ID	Dataset Name	# Instances	# Features	# Classes
13	breast-cancer	286	9	2
43	haberman	306	3	2
59	ionosphere	351	34	2
1498	sa-heart	400	9	2
40710	cleave	303	13	2

A.3. Testing Input Sequence Lengths

We used a node equipped with an NVIDIA H100 80GB, 64 CPU Cores, and 64GB RAM to test increasing input sequence lengths. For testing, we repeated the executions of Hydra- and Transformer-based TabPFN 10 times. Furthermore, instead of specific datasets, we generated random tensors with 99 feature columns and the number of rows ranging from 2^5 to 2^{18}.

A.4. Hyperparameter

A thorough hyperparameter optimization was not feasible due to long training times. We found that the number of steps per epoch and the batch size proposed in (Hollmann et al., 2023) lead to catastrophic forgetting effects in both Mamba and Hydra. Therefore, we changed those values to avoid such issues. By default, we used double the number of Transformer encoder layers as SSM blocks as suggested by (Gu & Dao, 2023). Table A.4 lists all hyperparameters used for training.

Table 2. Hyperparameter Table for Training

Hyperparameter	Mamba	Transformer	Hydra
Learning Rate	0.0001	0.0001	0.0001
Dropout	0.0	0.0	0.0
Batch Size	512	64	128
Steps per Epoch	16	1024	64
Aggregate k Gradients	8	8	8
Embedding Size	1024	512	1024
Hidden Size	1024	1024	1024
Number of Layers	24	12	24
Number of Heads	-	4	-
Recompute Attention	-	True	-
Use AMP	True	True	True
Optimizer	AdamW	AdamW	AdamW

A.5. OpenML CC-18 Datasets

We used the OpenML CC-18 benchmark datasets, filtered for TabPFN limitations (Hollmann et al., 2023). Table A.5 lists each dataset with its respective characteristics.

Table 3. OpenML CC-18 Datasets Table, filtered on TabPFN limitations

Dataset-ID	Dataset Name	# Instances	# Features	# Classes	# NaNs	Size Min. Class
11	balance-scale	625	5	3	0	49
14	mfeat-fourier	2000	77	10	0	200
15	breast-w	699	10	2	16	241
16	mfeat-karhunen	2000	65	10	0	200
18	mfeat-morphological	2000	7	10	0	200
22	mfeat-zernike	2000	48	10	0	200
23	cmc	1473	10	3	0	333
29	credit-approval	690	16	2	67	307
31	credit-g	1000	21	2	0	300
37	diabetes	768	9	2	0	268
50	tic-tac-toe	958	10	2	0	332
54	vehicle	846	19	4	0	199
188	eucalyptus	736	20	5	448	105
458	analcatdata_authorship	841	71	4	0	55
469	analcatdata_dmft	797	5	6	0	123
1049	pc4	1458	38	2	0	178
1050	pc3	1563	38	2	0	160
1063	kc2	522	22	2	0	107
1068	pc1	1109	22	2	0	77
1462	banknote-authentication	1372	5	2	0	610
1464	blood-transfusion-service-center	748	5	2	0	178
1480	ilpd	583	11	2	0	167
1494	qsar-biodeg	1055	42	2	0	356
1510	wdbc	569	31	2	0	212
6332	cylinder-bands	540	40	2	999	228
23381	dresses-sales	500	13	2	835	210
40966	MiceProtein	1080	82	8	1396	105
40975	car	1728	7	4	0	65
40982	steel-plates-fault	1941	28	7	0	55
40994	climate-model-simulation-crashes	540	21	2	0	46

B. Additional Results

Here, we provide additional results that put the performance of our approaches more into context. The environment for those results was the same as in the other ones in this paper.

B.1. Detailed Performance of SSMs and TabPFN over OpenML Datasets

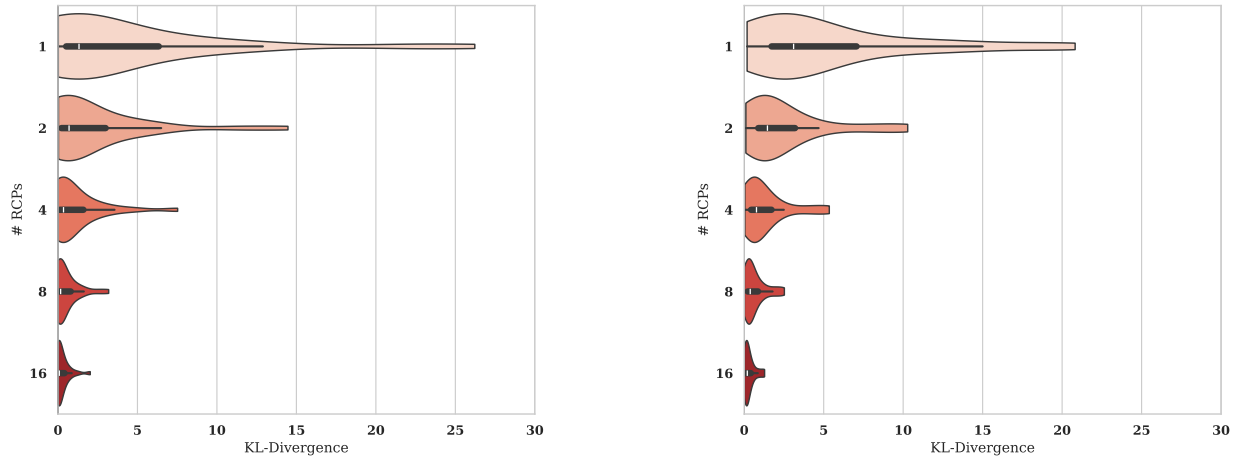
As in Section 4, we compare both SSM architectures with the Transformer-based TabPFN on the OpenML CC-18 benchmark. Model evaluation was performed on each resulting test set, where the average and the standard deviation of all 16 splits were reported. In Table 4, bold values denote the best performance regarding dataset ID and the specific metric. The second best value is underlined if the result is not significantly improved (meaning $p \not< 0.05$), determined by a two-sided Wilcoxon signed rank test.

B.2. KL-Divergence over RPC

In Section 4, we thematized how RCP impacted the KL divergence of the Mamba- and Hydra-based PFNs by averaging the values for the 30 datasets. Figure 5 highlights the distribution of the KL-divergence for the datasets.

Table 4. Results on the OpenML CC-18 dataset with constraints

DID	AUC OVO			Accuracy		
	Transformer	Mamba	Hydra	Transformer	Mamba	Hydra
11	**0.9977** ± .003	0.9648 ± .008	0.9951 ± .004	**0.9824** ± .010	0.8968 ± .013	0.9702 ± .015
14	**0.9753** ± .003	0.9274 ± .007	0.9557 ± .005	**0.7925** ± .017	0.5751 ± .020	0.7002 ± .018
15	0.9942 ± .002	0.9934 ± .002	**0.9946** ± .001	0.9698 ± .004	**0.9732** ± .003	0.9679 ± .004
16	**0.9969** ± .001	0.9657 ± .006	0.9857 ± .003	**0.9418** ± .009	0.7708 ± .028	0.8604 ± .019
18	**0.9635** ± .003	0.9461 ± .003	0.9574 ± .003	**0.7361** ± .014	0.6422 ± .018	0.6949 ± .017
22	**0.9819** ± .001	0.9270 ± .010	0.9697 ± .003	**0.8218** ± .015	0.6029 ± .031	0.7352 ± .021
23	0.7187 ± .015	0.7027 ± .016	**0.7213** ± .016	0.5414 ± .016	0.5243 ± .019	**0.5422** ± .015
29	**0.9308** ± .016	0.9287 ± .016	0.9288 ± .017	**0.8677** ± .019	0.8664 ± .019	0.8673 ± .018
31	**0.7971** ± .014	0.7846 ± .015	0.7911 ± .015	**0.7631** ± .013	0.7511 ± .009	0.7625 ± .012
37	0.8356 ± .011	0.8342 ± .010	**0.8369** ± .011	0.7676 ± .011	0.7703 ± .011	**0.7712** ± .008
50	**0.9728** ± .010	0.6739 ± .022	0.8764 ± .022	**0.9087** ± .013	0.6901 ± .019	0.8184 ± .019
54	**0.9580** ± .003	0.8939 ± .007	0.9293 ± .006	**0.8156** ± .014	0.6622 ± .018	0.7459 ± .015
188	**0.9143** ± .006	0.8776 ± .008	0.9049 ± .005	**0.6539** ± .020	0.5906 ± .021	0.6361 ± .015
458	**1.0000** ± .0	0.9997 ± .0	0.9999 ± .0	**0.9979** ± .002	0.9905 ± .004	0.9954 ± .003
469	**0.5651** ± .015	0.5500 ± .012	0.5649 ± .017	0.1891 ± .014	**0.1916** ± .015	0.1908 ± .015
1049	**0.9326** ± .011	0.9044 ± .014	0.9248 ± .010	**0.9021** ± .013	0.8911 ± .015	0.9008 ± .011
1050	**0.8298** ± .023	0.8080 ± .022	0.8194 ± .021	0.8982 ± .010	0.8987 ± .009	**0.8992** ± .009
1063	**0.8442** ± .025	0.8420 ± .026	0.8412 ± .025	0.8379 ± .018	0.8372 ± .015	**0.8408** ± .015
1068	**0.8851** ± .027	0.8266 ± .021	0.8447 ± .027	0.9299 ± .011	0.9301 ± .011	**0.9301** ± .011
1462	**1.0000** ± .0	**1.0000** ± .0	**1.0000** ± .0	**0.9999** ± .0	0.9982 ± .002	0.9993 ± .001
1464	0.7552 ± .021	0.7550 ± .024	**0.7562** ± .024	**0.7883** ± .013	0.7871 ± .015	0.7843 ± .010
1480	0.7412 ± .027	**0.7429** ± .022	0.7374 ± .023	0.7075 ± .028	**0.7135** ± .019	0.7047 ± .024
1494	**0.9324** ± .007	0.9165 ± .007	0.9229 ± .007	**0.8784** ± .012	0.8600 ± .011	0.8656 ± .014
1510	**0.9963** ± .002	0.9962 ± .003	0.9960 ± .003	**0.9769** ± .009	0.9727 ± .007	0.9736 ± .009
6332	**0.7755** ± .035	0.7200 ± .034	0.7430 ± .034	**0.7314** ± .037	0.6916 ± .024	0.7097 ± .043
23381	0.4637 ± .069	**0.4885** ± .071	0.4813 ± .061	0.5587 ± .041	**0.5778** ± .030	0.5625 ± .030
40966	**1.0000** ± .0	0.9716 ± .008	0.9908 ± .003	**0.9950** ± .007	0.7534 ± .046	0.8700 ± .025
40975	**0.9880** ± .006	0.9057 ± .015	0.9775 ± .010	**0.9597** ± .009	0.8523 ± .012	0.9336 ± .015
40982	**0.9547** ± .007	0.8758 ± .010	0.9280 ± .008	**0.7354** ± .017	0.5685 ± .025	0.6757 ± .023
40994	**0.9444** ± .014	0.9352 ± .015	0.9428 ± .014	**0.9475** ± .014	0.9412 ± .015	0.9472 ± .014



(a) Hydra (b) Mamba

Figure 5. Impact of RPC on KL-divergence of SSM-based PFN class values