Towards Localization via Data Embedding for TabPFN

Anonymous Author(s)
Affiliation
Address
email

Abstract

Prior-data fitted networks (PFNs), especially TabPFN, have shown significant promise in tabular data prediction. However, their scalability is limited by the quadratic complexity of the transformer architecture's attention across training points. In this work, we propose a method to localize TabPFN, which embeds data points into a learned representation and performs nearest neighbor selection in this space. We evaluate it across six datasets, demonstrating its superior performance over standard TabPFN when scaling to larger datasets. We also explore its design choices and analyze the bias-variance trade-off of this localization method, showing that it reduces bias while maintaining manageable variance. This work opens up a pathway for scaling TabPFN to arbitrarily large tabular datasets.

1 Introduction

Prior-data fitted networks (PFNs; Müller et al., 2022) are a class of neural networks that are trained on synthetic prior data and perform in-context learning for new tasks. TabPFN (Hollmann et al., 2023), a specific implementation of PFNs for tabular data, has shown impressive performance, often rivaling state-of-the-art models such as random forests and gradient boosting (McElfresh et al., 2023). However, a key limitation of TabPFN is its use of a transformer architecture (Vaswani et al., 2017), which scales quadratically with the number of training points due to the self-attention mechanism. TabPFN was trained on up to 1024 training data points, yet, in a scaling experiment, the model demonstrated improved performance up to 4096 data points. Nagler (2023) studied the underlying statistical foundations and conducted a bias-variance analysis of the PFN model and found that improved performance for larger datasets is due to a reduction in variance and demonstrated this using a simple toy experiment. In this work, we extend this preliminary experimental study and 1) propose a method to localize TabPFN that we dub LE-TabPFN, 2) study design decisions of LE-TabPFN, and 3) study its performance on 6 datasets. We show that the localization method that we dub LE-TabPFN leads to improved performance over TabPFN with dataset subsamples for large datasets and is a promising candidate for scaling TabPFN to arbitrary dataset sizes.

2 Background

TabPFN (Hollmann et al., 2023) belongs to the broader class of prior-data fitted networks (PFNs, Müller et al., 2022). It is a foundation model that is pre-trained on synthetic datasets to approximate $p(y|\mathbf{x}_*, \mathcal{D})$, i.e. a training dataset $\mathcal{D}$ and a query point $\mathbf{x}_*$, for which we want to make a prediction y. TabPFN conducts in-context learning, which is in contrast to traditional machine learning, which requires training and hyperparameter tuning of a supervised learning algorithm for every new dataset. For a new dataset, only a forward pass through the PFN is required.

While TabPFN has demonstrated robust predictive performance, its reliance on a transformer-based self-attention mechanism means that the computational cost scales quadratically with the number

Submitted to 3rd Table Representation Learning Workshop at NeurIPS (TRL@NeurIPS 2024). Do not distribute.

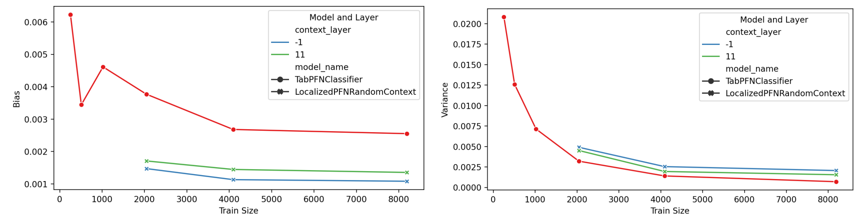

Figure 1: Bias-variance decomposition of the prediction error of TabPFN. Left: bias, Right: variance.

of data points. Although the model was trained with a maximum of 1024 data points, experiments by Hollmann et al. (2023) suggest that its performance continues to improve when presented with up to 4096 points. However, this comes at a significant memory and computational cost, making it impractical for even larger datasets. Nagler (2023) provided a theoretical explanation for this phenomenon through a bias-variance decomposition. In particular, the improved prediction quality can be entirely attributed to a decrease in variance. The predictions remain biased, however, because the TabPFN arichtecture does not adequately localize the predictions around the feature values. To alleviate this, Nagler (2023) proposed a simple *localization* strategy:

1. Construct a reduced training set $\widetilde{\mathcal{D}}(\mathbf{x})$ by keeping only the k nearest neighbors of $\mathbf{x}$ from $\mathcal{D}$.
2. Predict the label corresponding to $\mathbf{x}$ using only $\widetilde{\mathcal{D}}(\mathbf{x})$ as training data.

This strategy leads to a decreasing bias when going beyond what TabPFN was trained for, and additionally, could be a promising strategy for scaling TabPFN to arbitrary dataset sizes.

3 Method

We propose to refine the localization by basing it on a learned representation that we read out at intermediate layers of TabPFN. Concretely, this requires several design decisions: 1. A layer in TabPFN to read out the transformed representation. 2. A separate and fixed context, that is used to embed new data points into the intermediate representation 3. A distance function between two points in the embedded space.

The choices of the readout layer depend on the TabPFN architecture. The current implementation available uses an encoder-only architecture with 12 layers, followed by a 3-layer fully-connected neural network. For this initial study we chose the simplest possibilities for 2 and 3: a random context of size 1024, and the Euclidean distance. For the readout layer we chose the last encoder block, unless specified otherwise. By wrapping this method around the scikit-learn interface (Pedregosa et al., 2011), we localize the context on a per-query basis, scaling the features for each test point independently. Because our method localizes the context using embeddings, we dub it *LE-TabPFN*.

4 Bias-Variance Decomposition of the Localized Embedded TabPFN

To validate our localization approach, we replicate the bias-variance decomposition experiment from Nagler (2023) and compute the bias-variance decomposition of the RMSE. For this, we first simulate 100 datasets $\mathcal{D}_n$ from $p_0(1|X) = \frac{1}{2} + \sin(1^T\mathbf{X})/2$ with $Y \in \{0,1\}, X \sim \mathcal{N}(0, I_5)$, and apply TabPFN and LE-TabPFN. Then, we compute the average squared bias and variance over 500 samples $X_{test} \sim \mathcal{N}(0, I_5)$. In contrast to the original experiment, that only used up to 4000 data points in a single dataset, we use up to 8096 data points. Our results in Figure 1 confirm that the localization method reduces bias compared to the original TabPFN, while the increase in variance remains small.

5 Exploratory Experiments

5.1 Experimental Setup

Because of the exploratory nature of our paper, we restrict ourselves to a small number of datasets. Concretely, we use three datasets that were previously used to demonstrate scaling effects in TabPFN (adult-census, electricity and eeg-eye-state, Thomas et al., 2024). We note that adult-census and electricity are suboptimal to examine TabPFN as they contain missing values and categorical

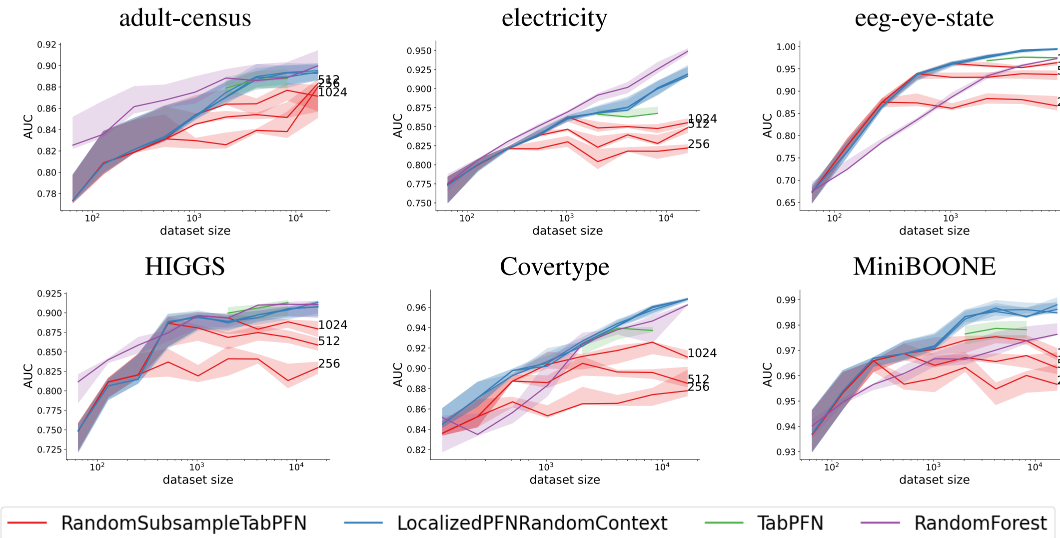

Figure 2: Median AUC over dataset size.

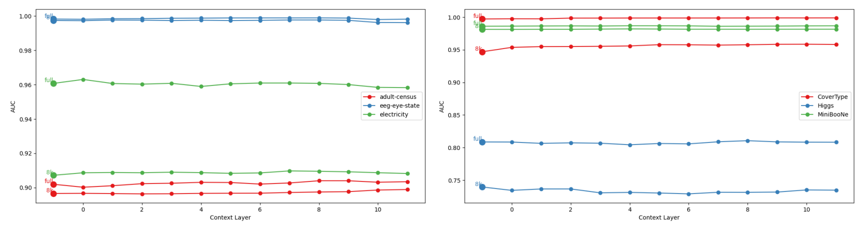

Figure 3: Performance of LE-TabPFN as a function of the readout layer. -1 is the raw data, 0 is the embedding layer of the transformer, and positive numbers are the respective encoder blocks.

features, two dataset characteristics that TabPFN was not trained on, and which are currently handled by preprocessors that wrap the actual model. In addition we use three large datasets that only contain numerical features (Higgs, Covertype and MiniBooNe) and less than 100 features to replicate the training setting of the TabPFN. We obtained the datasets from OpenML (Vanschoren et al., 2014) using OpenML-Python (Feurer et al., 2021). Furthermore, we restricted ourselves to only the 1st fold and only 2000 test data points of the respective OpenML tasks to keep the computational cost low. We impute missing values with the per-feature mean and conduct three repetitions per dataset.

5.2 Does the Localization allow Scaling to Arbitrary Dataset sizes

Next, we investigate whether LE-TabPFN can leverage additional data to improve performance, as hypothesized. We compare learning curves for LE-TabPFN, TabPFN with 4096 data points, TabPFN with random subsamples, and a random forest (Breiman, 2001). As shown in Figure 2, LE-TabPFN continues to improve with larger training sets, while standard TabPFN with random subsamples plateaus. LE-TabPFN also improves over TabPFN with up to 4096 data points, which suggests that the reduction in bias outweighs the reduction in variance due to the increased number of data points.

5.3 Understanding the impact of the readout layer on predictive quality

We hypothesize that later layers better capture the relation between data points, and thereby, lead to embeddings that produce a better context, giving rise to improved performance. However, as seen in Figure 3, the impact of the readout layer is surprisingly small, with only minor improvements for later layers on a subsample of the CoverType dataset. This suggests that the original feature space remains highly informative for the datasets in question.

Surprised by the small impact of the readout layer, we now try to find possible causes for this unexpected outcome. First, we check if using a "remote" context – comprising the most distant data points – rather than a local one, results in degraded performance. As shown in the bottom of

Table 1: Quantitative comparison of LE-TabPFN with TabPFN on all dataset. Top: results using 8000 training data points. Middle: results using all training data points. Bottom: ablation using a remote context, where the performance drops when using the last layer for the embedding (Section 5.3).

train_size	model_name	context_layer	CoverType	Higgs	MiniBooNe	adult-census	eeg-eye-state	electricity
	TabPFN on random subsamples		90.99	69.87	96.81	88.30	94.75	85.59
8000.0	TabPFN		93.40	73.25	97.46	89.73	97.75	87.82
	RandomForest		95.01	**77.02**	97.26	89.56	97.38	**93.21**
		raw features	94.68	73.97	98.12	89.65	**99.73**	90.72
	LE-TabPFN	0	95.35	73.45	98.11	89.66	99.72	90.86
		11	**95.81**	73.48	**98.14**	**89.89**	99.61	90.82
	TabPFN on random subsamples		91.97	67.63	97.51	87.82	94.53	85.34
full	RandomForest		99.79	**81.55**	98.54	89.80	98.38	**97.13**
		raw features	99.71	80.87	98.60	90.19	**99.82**	96.07
	LE-TabPFN	0	99.74	80.85	98.61	90.02	99.80	96.30
		11	**99.87**	80.83	**98.66**	**90.34**	99.81	95.82
		raw features	46.54	56.00	53.42	77.69	44.82	62.68
full	Remote context TabPFN	0	35.49	44.21	10.67	29.20	45.32	30.22
		11	27.85	34.21	20.24	16.40	9.63	25.64

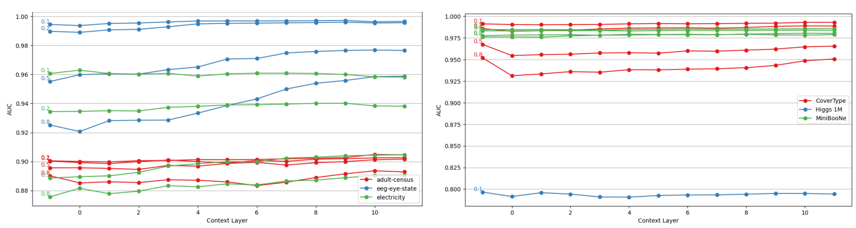

Figure 4: Performance of LE-TabPFN as a function of the readout layer when adding various amounts of random features ($X \in \{0.1, 0.2, 0.5, 0.8\}$ stands for $X * n_{features}$ random features that are added to the original dataset to reduce the meaningfullness of the original representation).

Table 1, the results confirm our hypothesis: performance declines significantly when embeddings from deeper layers are used, with AUC dropping below chance level. This indicates that later layers capture different information compared to earlier ones. We suspect that the raw feature space is "too informative" for our task, meaning that Euclidean distances in the original space are already highly meaningful. To analyze this, we augment the dataset with random features, reducing the relevance of the original features and expecting better performance from embeddings derived from deeper layers. The results in Figure 4 support this hypothesis: when we add random features, the performance improves as we use embeddings from later layers, outperforming both earlier layers and raw features.

5.4 Main Results: TabPFN vs Localized Embedded TabPFN

Lastly, we draw a quantitative comparison between TabPFN on 8000 data points, the random subsample TabPFN (subsampled to 1024 data points), a random forest trained on all data, and the LE-TabPFN, on subsamples of 8000 data points as well on the full datasets. We give all results in Table 1, and can observe that LE-TabPFN improves over TabPFN using a random subsample on all studied datasets. Also, while TabPFN is inferior to the RandomForest on all studied datasets, LE-TabPFN is superior to RandomForest on four out of six datasets, and almost closes the performance gap on the remaining two datasets. We do not find a strong impact of the readout layers on these datasets. Overall, we can see that localization can scale TabPFN to arbitrary training sizes.

6 Conclusion and Future Work

We demonstrated that the localization principle is a powerful paradigm to scale PFNs to large datasets. The localization principle is especially helpful for the TabPFNs model, which can now be applied to machine learning datasets with more than 1024 data points. In the future, we plan to include the localization in the pre-training step, as suggested by Nagler (2023). In addition, we want to compare against a similar idea motivated by RAG (Thomas et al., 2024). Since our work is mostly a proof-of-concept, we have not yet optimized it for inference speed. Finally, we also plan to extend the empirical study, and to compare against methods to learn a single, static context (Feuer et al., 2024; Rundel et al., 2024; Ma et al., 2024), and other standard models (boosting, neural networks) in large-scale settings, such as TabZilla (McElfresh et al., 2023).

References

Müller, S., N. Hollmann, S. Arango, J. Grabocka, and F. Hutter (2022). “Transformers Can Do Bayesian Inference”. In: *Proceedings of the International Conference on Learning Representations (ICLR’22)*. Published online: `iclr.cc`.

Hollmann, N., S. Müller, K. Eggensperger, and F. Hutter (2023). “TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second”. In: *International Conference on Learning Representations (ICLR’23)*. Published online: `iclr.cc`.

McElfresh, D., S. Khandagale, J. Valverde, V. Prasad C, G. Ramakrishnan, M. Goldblum, and C. White (2023). “When Do Neural Nets Outperform Boosted Trees on Tabular Data?” In: *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS’23)*. Curran Associates, pp. 76336–76369.

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

Nagler, T. (2023). “Statistical Foundations of Prior-Data Fitted Networks”. In: *Proceedings of the 40th International Conference on Machine Learning (ICML’23)*. Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett. Vol. 202. Proceedings of Machine Learning Research. PMLR, pp. 25660–25676.

Pedregosa, F. et al. (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Thomas, V., J. Ma, R. Hosseinzadeh, K. Golestan, G. Yu, M. Volkovs, and A. Caterini (2024). “Retrieval & Fine-Tuning for In-Context Tabular Models”. In: *1st ICML Workshop on In-Context Learning*.

Vanschoren, J., J. van Rijn, B. Bischl, and L. Torgo (2014). “OpenML: Networked Science in Machine Learning”. In: *SIGKDD Explorations* 15.2, pp. 49–60.

Feurer, M., J. van Rijn, A. Kadra, P. Gijsbers, N. Mallik, S. Ravi, A. Müller, J. Vanschoren, and F. Hutter (2021). “OpenML-Python: an extensible Python API for OpenML”. In: *Journal of Machine Learning Research* 22.100. Ed. by B. Kegl, pp. 1–5.

Breiman, L. (2001). “Random Forests”. In: *Machine Learning Journal* 45, pp. 5–32.

Feuer, B., R. Schirrmeister, V. Cherepanova, C. Hegde, F. Hutter, M. Goldblum, N. Cohen, and C. White (2024). “TuneTables: Context Optimization for Scalable Prior-Data Fitted Networks”. In: *arXiv:2402.11137 [cs.LG]*.

Rundel, D., J. Kobialka, C. von Crailsheim, M. Feurer, T. Nagler, and D. Rügamer (2024). “Interpretable Machine Learning for TabPFN”. In: *Explainable Artificial Intelligence*. Ed. by L. Longo, S. Lapuschkin, and C. Seifert. Vol. 2154, pp. 465–476.

Ma, J., V. Thomas, G. Yu, and A. Caterini (2024). “In-Context Data Distillation with TabPFN”. In: *arXiv:2402.06971 [cs.LG]*.