Early Stopping Tabular In-Context Learning

Jaris Küken [1 2] **Lennart Purucker** [1] **Frank Hutter** [3 4 1]

Abstract

Tabular foundation models have shown strong performance across various tabular learning tasks via in-context learning, offering robust generalization without any downstream finetuning. However, their inference-time costs remain high, particularly for larger datasets. To address this, we propose early-stopping the in-context learning process. We achieve this by dynamically evaluating whether to stop in-context learning after each Transformer encoder layer. Once stopped, we decode the embedding using a pre-trained layer-wise decoder. Experiments across 34 small classification tasks size show that early stopping in-context learning accelerates inference by up to $\times 1.3$ with negligible degradation in predictive performance. To assess scalability, we further evaluate our method on five larger classification tasks, achieving speedups of up to $\times 2.2$. Our results demonstrate the potential of early exiting as an effective and practical strategy for improving the efficiency of tabular in-context learning.

1. Introduction

Tabular data is widely present across domains such as finance or healthcare (Borisov et al., 2022; van Breugel & van der Schaar, 2024), many of which involve time-critical decision making. Traditionally, tree-based models like XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017) or CatBoost (Prokhorenkova et al., 2018) have demonstrated strong performance on tabular problems, often surpassing deep learning approaches (Grinsztajn et al., 2022). However, these models typically require extensive training and hyperparameter tuning for each downstream task.

Recent research has shifted towards tabular foundation models (TFMs) (van Breugel & van der Schaar, 2024), many of which leverage in-context learning (Brown et al., 2020) to enable fast adaption with minimal or no task-specific training. Notable examples include TabPFN (Hollmann et al., 2025; 2023), TabICL (Qu et al., 2025) or TabDPT (Ma et al., 2024). These models build on top of the Transformer architecture (Vaswani et al., 2017) to condition the context data and perform inference directly on new tasks. However, the self-attention mechanism scales quadratically with context size, making inference costly in TFMs.

We investigate whether the in-context learning in TFMs can be stopped early—without significant loss in performance—through an entropy-based early exiting strategy. Specifically, we pre-train lightweight decoders at each Transformer layer on synthetic data and use them to probe the test sample during inference. Based on the entropy of the prediction, we determine whether to exit early. This enables inference speedups of up to $\times 1.3$ on small datasets and up to $\times 2.2$ on larger tasks, with minimal degradation in performance and without the need for any downstream task-specific finetuning. Thus, we preserve a key advantage of TFMs: strong predictive performance without downstream task-specific finetuning. While early-exit strategies have been explored in natural language processing (Xin et al., 2020; Liu et al., 2020; Hou et al., 2020) and vision (Teerapittayanon et al., 2016), they remain largely underexplored in the context of in-context learning. Our work aims to close this gap and highlights the potential of early exiting as a practical tool for improving TFM efficiency. **Our contributions are: (I)** We introduce a simple yet effective entropy-based early-stopping mechanism for tabular in-context learning. **(II)** We demonstrate that inference of TFMs can be sped up by up to $\times 1.3$ on small tasks and $\times 2.2$ on larger tasks with negligible loss in predictive performance. **(III)** Unlike prior early-exit methods, our approach does not require task-specific finetuning, maintaining the advantages of in-context learning of TFMs.

2. Related Work

Improving Transformer Efficiency. Improving the computational efficiency of Transformer models (Vaswani et al., 2017) has been widely studied, particularly natural language processing. Early exit strategies have been introduced for encoder-based architectures like BERT (Devlin et al., 2019),

[*]Equal contribution [1]Department of Computer Science, University of Freiburg, Freiburg, Germany [2]Zuse School ELIZA, Darmstadt, Germany [3]Prior Labs, Freiburg, Germany [4]ELLIS Institute, Tübingen, Germany. Correspondence to: Jaris Küken <kuekenj@cs.uni-freiburg.de>.

Foundation Models for Structured Data at the 42^{nd} International Conference on Machine Learning, Vancouver, Canada.

allowing inference to terminate at intermediate layers without traversing the full model. Notable examples include DynaBERT (Hou et al., 2020), FastBERT (Liu et al., 2020) or CascadeBERT (Li et al., 2021). Additional efficiency-oriented approaches such as quantization (Kim et al., 2021; Shen et al., 2020; Zafrir et al., 2019) and knowledge distillation (Hinton et al., 2015; Wang et al., 2020; Sanh et al., 2020), have also been proposed, though they typically involve model compression rather then dynamic inference control and are therefore related to our work. Our method is most related to early-exit architectures like DeeBERT (Xin et al., 2020) and BranchyNet (Teerapittayanon et al., 2016), but differs in its application to in-context learning. In contrast to prior work, which often requires task-specific finetuning to calibrate exit decisions, our approach leverages a pretrained, fixed exit mechanism that enables efficient inference.

Tabular In-Context Learning. In-context learning (ICL) (Brown et al., 2020) has recently been adapted to tabular data through several distinct approaches. One prominent line of work interprets ICL as approximate Bayesian inference over tabular tasks (Xie et al., 2021; Müller et al., 2022; Reuter et al., 2025), leading to the development of Transformer-based foundation models such as TabPFN (Hollmann et al., 2023; 2025), TabICL (Qu et al., 2025) or TabDPT (Ma et al., 2024).

3. Method

We propose a straightforward yet effective early-exit mechanism for tabular in-context learning. Our method can be used with any TFM; for this study, we use TabPFNv2 (Hollmann et al., 2025). Our approach introduces pre-trained, layer-specific decoders into the Transformer (Vaswani et al., 2017) architecture, enabling dynamic termination of the forward pass during inference. At each layer, we decide whether to exist early based on prediction entropy, as illustrated in Figure 1. Crucially, our method requires no task-specific finetuning and preserves the zero-shot capabilities of the base model. All exit points are trained on synthetically generated data from a prior.

Method Motivation. We investigate whether intermediate layers of TabPFN can produce useful predictions prior to the final layer. As shown in Figure 2, activations from earlier layers, when passed through the final decoder, already yield non-trivial performance—albeit typically suboptimal compared to the last layer. This indicates that significant predictive information emerges well before the end of the forward pass. To better leverage these early representations, we pre-train dedicated decoders for each Transformer layer, enabling flexible early exits.

Layer-Specific Decoder Pretraining. We modify the original TabPFN (Hollmann et al., 2025) architecture by attaching a decoder to each encoder layer, as illustrated in Figure 1. Each intermediate decoder shares the architecture of the original TabPFN decoder, but is trained independently. To enable generalization across tasks, we pre-train each decoder on synthetic datasets sampled from the prior introduced by Müller et al. (2023)[1]. During training, the Transformer backbone is frozen, and only the decoder corresponding to the current exit layer is optimized. For the final layer, we retain the original TabPFN decoder. In total, each decoder is trained on approximately 820,000 synthetic datasets. Details of the data generation and training procedure are provided in Appendix A.

Early Stopping Based on Prediction Entropy. At inference time, we perform early stopping based on the predictive uncertainty of each layer's decoder, similar to BranchyNet (Teerapittayanon et al., 2016) and DeeBERT (Xin et al., 2020). After each Transformer layer, all test samples are passed through the corresponding decoder, and we compute the entropy of their predictive distributions. We then average the entropies across the test set and compare the result to a predefined threshold τ. If the average entropy falls below τ, the model exits early and outputs the corresponding predictions. Otherwise, the forward pass continues to the next layer. The threshold τ is a dataset specific, tunable hyperparameter. Importantly, as all decoders are pre-trained, no adaption to downstream tasks is needed, thus making our method a cheap and efficient addition to the standard forward pass of TabPFN. A detailed overview of our proposed algorithm is specified Appendix B.

4. Experiments

We evaluate our method on two categories of classification tasks to assess its effectiveness across dataset sizes. First, we benchmark on 34 small-scale binary classification tasks from the PMLB-mini suite (Knauer et al., 2024) containing datasets with up to 500 samples[2]. Second, we assess scalability and inference efficiency on five larger classification datasets containing up to 5,000 samples, where TabPFN's inference latency becomes more critical due to the quadratic complexity of the self-attention mechanism. We specify all datasets in Appendix C.1. For all experiments, we report mean performance over 10-fold cross-validation. We report details on the hyperparameters for the TabPFN evaluation in Appendix C.2.

Performance of Intermediate Layers. We investigate the predictive capacity of intermediate Transformer layers in TabPFN. To do this, we pre-train a decoder for each layer

[1]We adapt the pre-training setup specified in: `https://github.com/microsoft/ticl`

[2]All experiments are run on a single NVIDIA RTX 2080 GPU

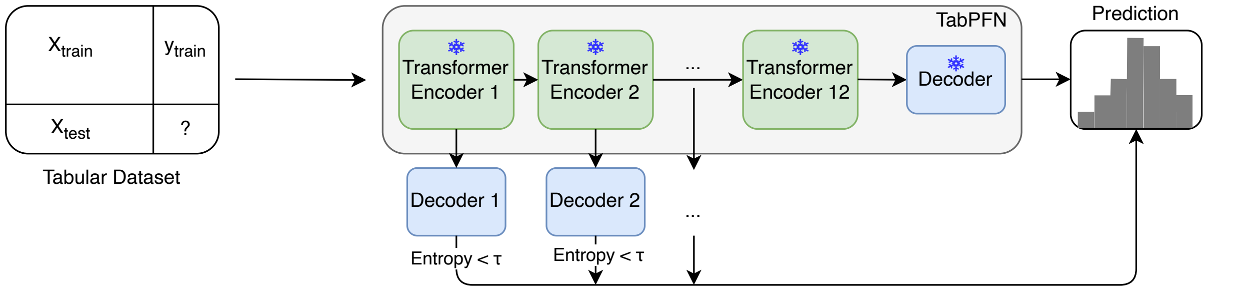

Figure 1. **Early Exit Strategy for TabPFN.** We extend TabPFN with an early-exit mechanism for in-context learning. For each Transformer layer, we pre-train a dedicated decoder on synthetically generated data from the prior (blue) while keeping the Transformer layers (green) and the final decoder frozen. During inference, all test samples are passed through each decoder in sequence. If the average prediction entropy falls below a predefined threshold τ, the forward pass is terminated early and predictions are returned.

as described in Section 3. For each benchmark dataset, we terminate the forward pass at each layer and pass the resulting representation through its corresponding decoder. We then measure the classification accuracy for each exit point.

Early Stopping based on Entropy. We evaluate the proposed entropy-based early-exit method as shown in Figure 1. We test a range of five entropy thresholds τ to control early exit behavior. For each setting, we report the predictive accuracy and the average number of Transformer layers evaluated per inference run. This allows us to quantify the trade-off between accuracy and computational cost.

Scalability to Larger Datasets. To further evaluate the scalability of our approach, we test it on larger datasets containing up to 5.000 samples—settings where TabPFN's inference latency becomes particularly critical. Importantly, since our decoders are pre-trained only on synthetic datasets with up to ~ 1.000 samples, this also serves as a test of the method's ability to generalize to larger-scale tasks.

5. Results

Intermediate Layers Enable Confident Early Exits. Figure 2 shows the classification accuracy obtained when exiting TabPFN at different Transformer layers using their corresponding pre-trained decoders, compared to using only the final decoder. Results are averaged over 34 small-scale datasets from the PMLB-mini (Knauer et al., 2024) benchmark. We observe that by pre-training individual decoders for each layer stabilizes performance across layers, with several intermediate layers reaching accuracy close to that of the final layer. This suggests that many inputs can be confidently predicted before completing the full forward pass, enabling faster inference with negligible loss in accuracy.

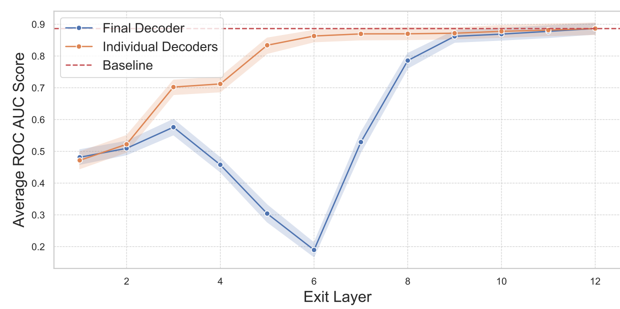

Figure 2. **Performance per Layer over 34 Small Datasets with Individual Decoders and Final Decoder.** We report the average ROC AUC score across 34 small-scale datasets, averaged over 10 folds, along with 95% confidence intervals. We compare two exit strategies: using the original final decoder at intermediate layers (blue) and using individually pre-trained decoders specific to each layer (red). For reference, we also include the full TabPFN baseline, where inference proceeds through the entire Transformer (red dashed). Notably, intermediate layers already achieve strong performance, with layers as early as 5 matching the final layer's accuracy when equipped with individual decoders. Moreover, these layer-wise decoders significantly outperform intermediate exits routed through the original final decoder.

Entropy-Based Early Exit Trades Accuracy for Efficiency. To validate our early exit strategy, we apply the entropy-based criterion across a range of thresholds τ. Figure 3 illustrates the trade-off between runtime savings and classification accuracy for large datasets, where inference cost is especially pressing. On larger tasks, thresholds in the range $\tau \in [0.4, 0.5]$ yield up to $\times 2.2$ faster inference with only small accuracy degradation. On small-scale datasets, thresholds between $\tau \in [0.3, 0.4]$ already achieve up to $\times 1.3$ speedup with negligible loss in predictive performance as shown in Table 1. Appendix C contains further evaluation

Table 1. **Impact of Different Entropy Thresholds on Inference Performance.** We report ROC AUC, difference in wall-clock runtime (in seconds) compared to full TabPFN forward pass, and average exit layer across 34 small and 4 large classification datasets for 5 different entropy thresholds. All values are averaged across datasets.

	Small Datasets					
	Baseline	τ=0.1	τ=0.2	τ=0.3	τ=0.4	τ=0.5
ROC AUC	0.886 ± 0.17	0.888 ± 0.17	0.887 ± 0.16	0.883 ± 0.17	0.873 ± 0.18	0.861 ± 0.19
Runtime Δ in (s)	—	0.004 ± 0.00	-0.013 ± 0.00	-0.044 ± 0.00	-0.069 ± 0.00	-0.084 ± 0.00
Avg. Exit Layer	12.0	11.0	9.8	7.6	6.1	5.3
	Large Datasets					
	Baseline	τ=0.1	τ=0.2	τ=0.3	τ=0.4	τ=0.5
ROC AUC	0.960 ± 0.04	0.957 ± 0.04	0.941 ± 0.04	0.923 ± 0.04	0.920 ± 0.04	0.920 ± 0.04
Runtime Δ in (s)	—	-0.550 ± 0.51	-1.194 ± 0.36	-1.369 ± 0.45	-1.415 ± 0.32	-1.464 ± 0.45
Avg. Exit Layer	12.0	9.48	7.02	5.42	5.0	5.0

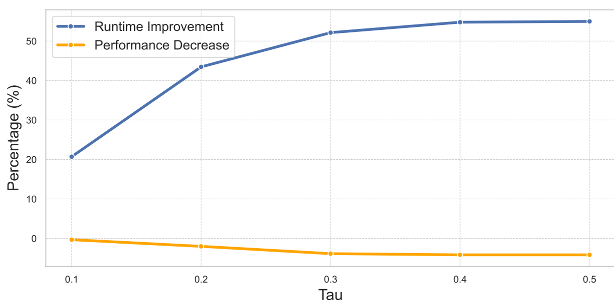

Figure 3. **Tradeoff between Improved Runtime and Decrease in Predictive Performance.** We report the relative improvements in runtime as well as the relative decrease in ROC AUC score averaged over 5 large datasets when early stopping based on entropy for 5 different entropy thresholds. It is trivial to see that a higher threshold τ leads to higher improvements in runtime, but also leads to a stronger decrease in terms of predictive performance. Notably, the gains in terms of relative runtime improvement strongly exceed the relative performance decrease.

results on small datasets. Across tasks, this allows TabPFN to preserve its strong predictive accuracy while significantly improving efficiency. Table 1 provides additional insights: the average exit layer, actual runtime savings in seconds, and corresponding accuracy deltas, across both small and large datasets for a fixed set of threshold values. These results confirm that our early-exit mechanism adapts flexibly to varying data scales, maintaining competitive predictive performance while offering tangible gains in inference efficiency. We report all discussed metrics, per-dataset in Appendix E. These findings demonstrate that entropy-based early exiting offers a practical and cheap mechanism to accelerate inference in tabular in-context learning without sacrificing significant predictive quality.

Generalization to Larger Datasets. Table 1 reports detailed results on larger datasets. Early stopping demonstrates strong potential in this setting, yielding up to $\times 2.2$ faster inference for thresholds in the range $\tau \in [0.4, 0.5]$. While performance degradation becomes more noticeable—up to 4%—compared to smaller datasets, it remains moderate relative to the efficiency gains. These results indicate that early stopping can substantially reduce inference costs even for large-scale tasks, and pave the way for more refined and adaptive strategies to maintain performance. At the same time, these results highlight the ability of the individual decoders generalize effectively to larger datasets, despite being trained only on smaller-scale data. Appendix C provides further insights into the evaluation of all benchmark datasets.

6. Conclusion

We propose early-exit tabular in-context learning, unlocking dynamic inference stopping for the Tabular Foundation Model (TFM) TabPFN based on prediction entropy. By pre-training decoders for each Transformer layer on synthetic data—without the need for any task-specific finetuning—our method introduces minimal overhead while significantly improving inference efficiency. Experiments across both small and large classification benchmarks indicate that many predictions can be confidently made before reaching the final layer, achieving up to $\times 1.3$ faster inference runtime with only marginal loss in accuracy on small datasets. On larger-scale tasks this effect is even more pronounced with inference being up to $\times 2.2$ faster, while still maintaining near-optimal performance. Our approach retains the zero-shot generalization and strong predictive performance of TFMs while mitigating the inference cost associated with larger input contexts. These results highlight the possibility of early exiting, progressing towards real-time inference for tabular in-context learning.

Acknowledgements

L.P. acknowledges funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under SFB 1597 (SmallData), grant number 499552394; and funding by ELSA – European Lighthouse on Secure and Safe AI funded by the European Union under Grant Agreement No. 101070617. Frank Hutter acknowledges the financial support of the Hector Foundation. Jaris Küken is supported by the Konrad Zuse School of Excellence in Learning and Intelligent Systems (ELIZA) through the DAAD program Konrad Zuse Schools of Excellence in Artificial Intelligence, sponsored by the Federal Ministry of Education and Research. Finally, we thank the reviewers for their constructive feedback and contribution to improving the paper.

References

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., and Kasneci, G. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.-F., and Lin, H. (eds.), *Proceedings of the 33rd International Conference on Advances in Neural Information Processing Systems (NeurIPS'20)*, pp. 1877–1901, 2020.

Chen, T. and Guestrin, C. XGBoost: A scalable tree boosting system. In Krishnapuram, B., Shah, M., Smola, A., Aggarwal, C., Shen, D., and Rastogi, R. (eds.), *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16)*, pp. 785–794, 2016.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Grinsztajn, L., Oyallon, E., and Varoquaux, G. Why do tree-based models still outperform deep learning on typical tabular data? In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Proceedings of the 35th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022.

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)*, 2017.

Hinton, G. E., Vinyals, O., and Dean, J. Distilling the Knowledge in a Neural Network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531. arXiv: 1503.02531.

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.

Hou, L., Huang, Z., Shang, L., Jiang, X., Chen, X., and Liu, Q. DynaBERT: Dynamic BERT with Adaptive Width and Depth. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9782–9793. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6f5216f8d89b086c18298e043bfe48ed-Abstract.html.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. In Guyon et al. (2017).

Kim, S., Gholami, A., Yao, Z., Mahoney, M. W., and Keutzer, K. I-BERT: Integer-only BERT Quantization. *International Conference on Machine Learning (Accepted)*, 2021.

Knauer, R., Grimm, M., and Rodner, E. PMLBmini: A Tabular Classification Benchmark Suite for Data-Scarce Applications, September 2024. URL http://arxiv.org/abs/2409.01635. arXiv:2409.01635 [cs].

Li, L., Lin, Y., Chen, D., Ren, S., Li, P., Zhou, J., and Sun, X. CascadeBERT: Accelerating Inference of Pre-trained Language Models via Calibrated Complete Models Cascade. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 475–486, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.43. URL https://aclanthology.org/2021.findings-emnlp.43/.

Liu, W., Zhou, P., Wang, Z., Zhao, Z., Deng, H., and Ju, Q. FastBERT: a Self-distilling BERT with Adaptive Inference Time. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6035–6044, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.537. URL https://aclanthology.org/2020.acl-main.537/.

Ma, J., Thomas, V., Hosseinzadeh, R., Kamkari, H., Labach, A., Cresswell, J. C., Golestan, K., Yu, G., Volkovs, M., and Caterini, A. L. TabDPT: Scaling Tabular Foundation Models, October 2024. URL http://arxiv.org/abs/2410.18164. arXiv:2410.18164 [cs].

Müller, A., Curino, C., and Ramakrishnan, R. Mothernet: A foundational hypernetwork for tabular classification. *arXiv:2312.08598 [cs.LG]*, 2023.

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.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A., and Gulin, A. Catboost: Unbiased boosting with categorical features. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'18)*, pp. 6639–6649, 2018.

Qu, J., Holzmüller, D., Varoquaux, G., and Morvan, M. L. TabICL: A Tabular Foundation Model for In-Context Learning on Large Data, February 2025. URL http://arxiv.org/abs/2502.05564. arXiv:2502.05564 [cs].

Reuter, A., Rudner, T. G. J., Fortuin, V., and Rügamer, D. Can Transformers Learn Full Bayesian Inference in Context?, January 2025. URL http://arxiv.org/abs/2501.16825. arXiv:2501.16825 [cs].

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, March 2020. URL http://arxiv.org/abs/1910.01108. arXiv:1910.01108 [cs].

Shen, S., Zhen, D., Ye, J., Ma, L., Yao, Z., Gholami, A., Mahoney, M., and Keutzer, K. Q-BERT: Hessian Based Ultra Low Precision Quantization of BERT. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34: 8815–8821, April 2020. doi: 10.1609/aaai.v34i05.6409.

Teerapittayanon, S., McDanel, B., and Kung, H. BranchyNet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469, December 2016. doi: 10.1109/ICPR.2016.7900006. URL https://ieeexplore.ieee.org/document/7900006.

van Breugel, B. and van der Schaar, M. Why tabular foundation models should be a research priority. 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.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. Attention is all you need. In Guyon et al. (2017).

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An Explanation of In-context Learning as Implicit Bayesian Inference. October 2021. URL https://openreview.net/forum?id=RdJVFCHjUMI.

Xin, J., Tang, R., Lee, J., Yu, Y., and Lin, J. DeeBERT: Dynamic Early Exiting for Accelerating BERT Inference. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2246–2251, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.204. URL https://aclanthology.org/2020.acl-main.204/.

Zafrir, O., Boudoukh, G., Izsak, P., and Wasserblat, M. Q8BERT: Quantized 8Bit BERT. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, pp. 36–39, December 2019. doi: 10.1109/EMC2-NIPS53020.2019.00016. URL https://ieeexplore.ieee.org/abstract/document/9463531.

A. Decoder Pre-Training.

We pre-train each decoder independently on synthetic datasets sampled from the prior, following the prior setup from (Hollmann et al., 2023). During training, all Transformer components are frozen; only the target decoder is updated. Each decoder is attached at a specific Transformer layer, receiving its output as input, and trained for a total of 100 epochs. Detailed training settings are provided in Table 2. Settings for the synthetic datasets from the prior are stated in Table 3. We train all decoders on a single NVIDIA RTX 2080 GPU.

Table 2. **Training Parameters.** We report the parameters for the pre-training setup. We train each decoder for a total of 100 epochs. #Steps/Epoch states the number of synthetically drawn datasets per epoch from the prior.

Training Parameters	Value
Epochs	100
Batch Size	8
#Steps/Epoch	1024
Learning Rate	3e-5

Table 3. **Parameters for the Prior.** We report the parameters for the synthetic datasets drawn from the Prior.

Prior Parameters	Value
#Samples per Dataset	1152
#Features per Dataset	100
#Max Classes per Dataset	10

B. Algorithm Details.

Algorithm 1 specifies the implementation of dynamic early exiting in tabular in-context learning.

Algorithm 1 Early-Exit Inference for TabPFN

Input: Table embedding $x = (x_{\text{train}}, y_{\text{train}}, x_{\text{test}})$, threshold τ
for $i = 1$ to $N_{TransformerLayers}$ **do**
  $x \leftarrow$ TransformerLayer$_i(x)$
  $\hat{y} \leftarrow$ Decoder$_i(x_{\text{test}})$
  $p \leftarrow$ softmax$(\hat{y})$
  $H \leftarrow -\sum p \log p$
  if $H < \tau$ **then**
    return p
  end if
end for
return p

C. Evaluation Details.

C.1. Datasets

We report details about the used small datasets in Table 4 and large datasets in Table 5.

Table 4. **Small Datasets.** The datasets used from the pmlb-mini suite (Knauer et al., 2024)

Dataset	OpenML ID	#Features	#Samples	#Targets
parity5	40714	5	32	2
analcatdata_fraud	40660	11	42	2
aids	346	4	50	2
analcatdata_bankruptcy	476	6	50	2
analcatdata_japansolvent	467	9	52	2
analcatdata_asbestos	459	3	83	2
lupus	472	3	87	2
postoperative-patient-data	40683	8	88	2
analcatdata_cyyoung9302	479	10	92	2
analcatdata_cyyoung8092	465	10	97	2
analcatdata_creditscore	461	6	100	2
molecular-biology_promoters	956	57	106	2
analcatdata_boxing1	448	3	120	2
mux6	40681	6	128	2
analcatdata_boxing2	444	3	132	2
corral	40669	6	160	2
backache	463	32	180	2
prnn_crabs	446	7	200	2
sonar	40	60	208	2
biomed	481	8	209	2
prnn_synth	464	2	250	2
analcatdata_lawsuit	450	4	264	2
SPECT	336	22	267	2
heart-statlog	53	13	270	2
hepatitis	55	19	155	2
breast-cancer	13	9	286	2
hungarian	231	13	294	2
cleve	40710	13	303	2
haberman	43	3	306	2
SPECTF	337	44	349	2
ionosphere	59	34	351	2
colic	27	22	368	2
vote	56	16	435	2
irish	451	5	500	2

C.2. Hyperparameters

We use TabPFNv2 (Hollmann et al., 2025) for all experiments. We use default classification hyperparameters from the paper evaluation.

Table 5. **Large Datasets.** The datasets used from the original TabPFNv2 evaluation. (Hollmann et al., 2025)

Dataset	OpenML ID	#Features	#Samples	#Targets
ada	41156	48	4147	2
churn	49791	20	5000	2
phoneme	1489	5	5404	2
Satellite	40900	36	5100	2
sylvine	41146	20	5124	2

D. Additional Results for Large Datasets and Small Datasets.

We present evaluation results—similar to the evaluation of the small datasets—for large datasets. Figure **??** illustrates the tradeoff between relative runtime improvement and performance decrease for small datasets. Figure 5 shows the performance per layer over large datasets.

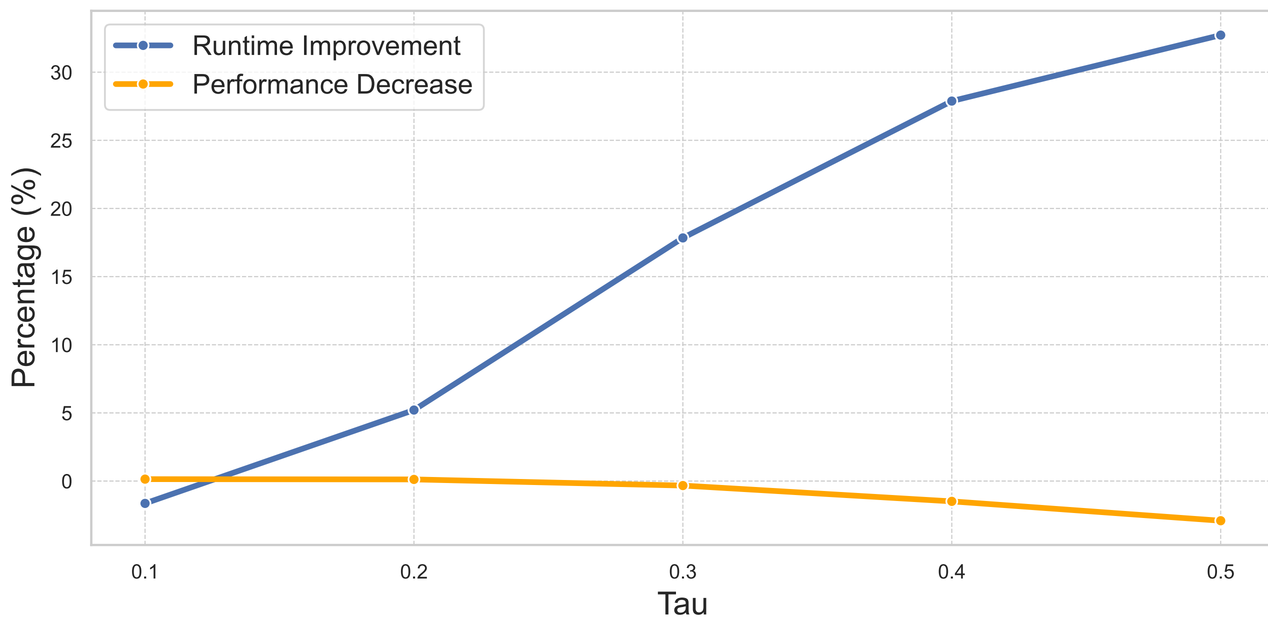

Figure 4. **Tradeoff between Improved Runtime and Decrease in Predictive Performance.** We report the relative improvements in runtime as well as the relative decrease in ROC AUC score averaged over 34 small datasets when early stopping based on entropy for 5 different entropy thresholds.

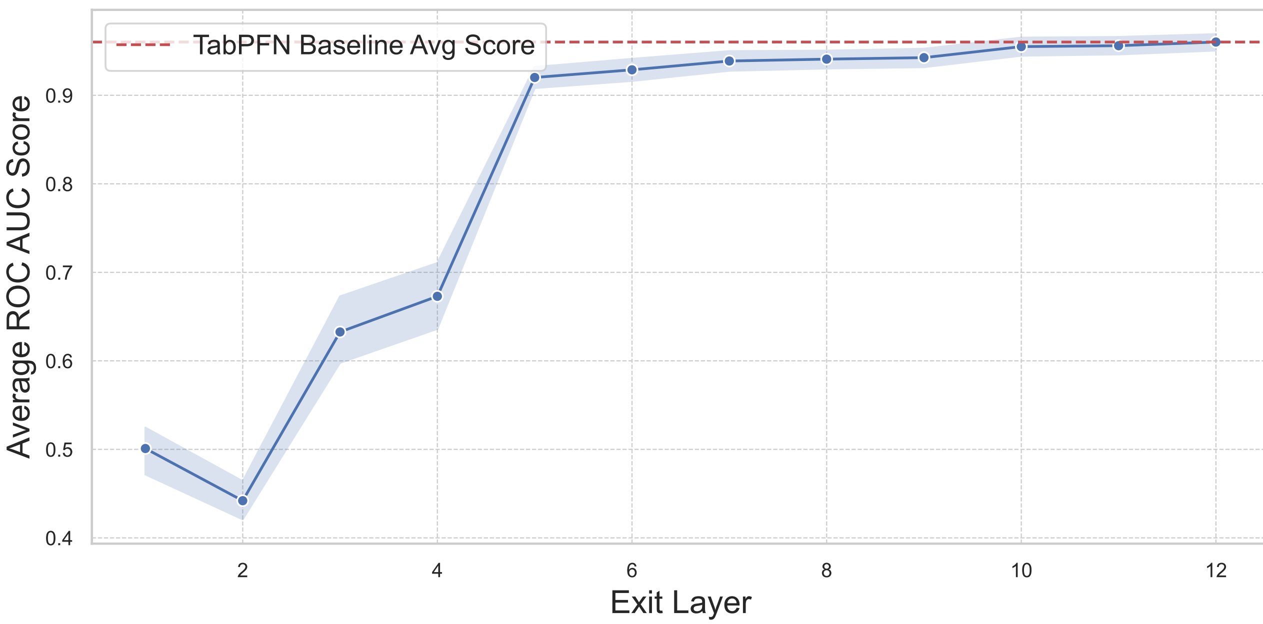

Figure 5. **Performance per Layer over 5 Large Datasets with Individual Decoders.** We present the performance average ROC AUC score over 5 large datasets with up to 5k samples averaged over 10 folds as well as the 95% confidence interval. Additionally we present the baseline of TabPFN when passing through the entire transformer in red. Similar to small datasets, early layers already exhibit strong performance.

E. Detailed Results.

We present the per-dataset results of our evaluation of small datasets in Table 6 and for large datasets in Table 8. We further report the exit layer per dataset, for small datasets in Table 7 and for large datasets in Table 9.

Table 6. **ROC AUC Scores per Dataset over Different Thresholds for Small Datasets.** We report ROC AUC scores per dataset for 5 different thresholds. All scores are averages over 10 fold cross validation.

Dataset	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
parity5	1.000 ± 0.00	1.000 ± 0.00	1.000 ± 0.00	0.950 ± 0.16	0.725 ± 0.38
analcatdata_fraud	0.833 ± 0.24	0.867 ± 0.23	0.800 ± 0.28	0.733 ± 0.31	0.767 ± 0.27
aids	1.000 ± 0.00	1.000 ± 0.00	1.000 ± 0.00	0.850 ± 0.32	0.783 ± 0.39
analcatdata_bankruptcy	0.967 ± 0.11	0.967 ± 0.11	0.967 ± 0.11	0.967 ± 0.11	0.950 ± 0.16
analcatdata_japansolvent	0.933 ± 0.14	0.950 ± 0.11	0.900 ± 0.22	0.900 ± 0.22	0.900 ± 0.22
analcatdata_asbestos	0.871 ± 0.15	0.871 ± 0.15	0.872 ± 0.15	0.866 ± 0.15	0.866 ± 0.15
lupus	0.820 ± 0.18	0.820 ± 0.18	0.826 ± 0.17	0.809 ± 0.18	0.728 ± 0.24
postoperative-patient-data	0.356 ± 0.14	0.356 ± 0.14	0.356 ± 0.14	0.384 ± 0.15	0.448 ± 0.19
analcatdata_cyyoung9302	0.917 ± 0.16	0.903 ± 0.16	0.896 ± 0.16	0.915 ± 0.16	0.915 ± 0.16
analcatdata_cyyoung8092	0.880 ± 0.13	0.896 ± 0.12	0.883 ± 0.14	0.864 ± 0.14	0.864 ± 0.14
analcatdata_creditscore	1.000 ± 0.00	1.000 ± 0.00	1.000 ± 0.00	0.995 ± 0.02	0.995 ± 0.02
molecular-biology_promoters	0.579 ± 0.25	0.585 ± 0.25	0.589 ± 0.27	0.610 ± 0.24	0.601 ± 0.24
analcatdata_boxing1	0.879 ± 0.10	0.879 ± 0.10	0.882 ± 0.10	0.892 ± 0.10	0.882 ± 0.10
mux6	1.000 ± 0.00	1.000 ± 0.00	0.990 ± 0.03	0.943 ± 0.10	0.904 ± 0.11
analcatdata_boxing2	0.853 ± 0.09	0.855 ± 0.09	0.853 ± 0.09	0.863 ± 0.08	0.822 ± 0.10
corral	1.000 ± 0.00	0.987 ± 0.02	0.979 ± 0.04	0.976 ± 0.04	0.976 ± 0.04
backache	0.701 ± 0.13	0.711 ± 0.15	0.703 ± 0.11	0.700 ± 0.11	0.697 ± 0.11
prnn_crabs	1.000 ± 0.00	1.000 ± 0.00	0.999 ± 0.00	0.999 ± 0.00	0.999 ± 0.00
sonar	0.944 ± 0.05	0.944 ± 0.05	0.944 ± 0.05	0.920 ± 0.09	0.902 ± 0.09
biomed	1.000 ± 0.00	0.978 ± 0.02	0.971 ± 0.03	0.973 ± 0.03	0.973 ± 0.03
prnn_synth	0.948 ± 0.03	0.945 ± 0.03	0.946 ± 0.03	0.946 ± 0.03	0.946 ± 0.03
analcatdata_lawsuit	0.990 ± 0.02	0.990 ± 0.02	0.990 ± 0.02	0.990 ± 0.02	0.990 ± 0.02
SPECT	0.833 ± 0.07	0.834 ± 0.07	0.844 ± 0.08	0.848 ± 0.08	0.848 ± 0.08
heart-statlog	0.908 ± 0.06	0.908 ± 0.06	0.912 ± 0.06	0.911 ± 0.06	0.910 ± 0.06
hepatitis	0.856 ± 0.09	0.864 ± 0.09	0.879 ± 0.08	0.876 ± 0.10	0.879 ± 0.10
breast-cancer	0.728 ± 0.06	0.730 ± 0.06	0.729 ± 0.06	0.738 ± 0.07	0.730 ± 0.07
hungarian	0.908 ± 0.06	0.909 ± 0.06	0.909 ± 0.07	0.912 ± 0.07	0.912 ± 0.07
cleve	0.910 ± 0.05	0.911 ± 0.05	0.912 ± 0.06	0.909 ± 0.06	0.909 ± 0.06
haberman	0.714 ± 0.17	0.714 ± 0.17	0.719 ± 0.16	0.700 ± 0.13	0.697 ± 0.12
SPECTF	0.964 ± 0.04	0.964 ± 0.04	0.951 ± 0.05	0.949 ± 0.04	0.949 ± 0.04
ionosphere	0.983 ± 0.02	0.974 ± 0.02	0.957 ± 0.04	0.949 ± 0.04	0.949 ± 0.04
colic	0.901 ± 0.04	0.900 ± 0.04	0.892 ± 0.04	0.868 ± 0.05	0.869 ± 0.05
vote	0.992 ± 0.01	0.983 ± 0.02	0.979 ± 0.03	0.979 ± 0.03	0.979 ± 0.03
irish	1.000 ± 0.00	1.000 ± 0.00	1.000 ± 0.00	1.000 ± 0.00	1.000 ± 0.00

Table 7. **Exit Layer per Dataset over Different Thresholds for Small Datasets.** We report the exit layer per dataset for 5 different thresholds when dynamically early stopping. All exit layers are averages over 10 fold cross validation.

Dataset	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
parity5	10.3	10.0	9.4	7.1	5.3
analcatdata_fraud	12.0	10.6	7.1	5.7	5.0
aids	12.0	12.0	12.0	11.2	9.1
analcatdata_bankruptcy	11.9	9.8	6.7	5.7	5.0
analcatdata_japansolvent	12.0	12.0	6.7	5.6	5.0
analcatdata_asbestos	12.0	11.6	7.2	5.0	5.0
lupus	12.0	12.0	11.3	7.9	5.0
postoperative-patient-data	12.0	12.0	12.0	10.6	5.7
analcatdata_cyyoung9302	11.3	7.5	5.2	5.0	5.0
analcatdata_cyyoung8092	12.0	10.8	5.9	5.0	5.0
analcatdata_creditscore	8.0	6.0	5.0	5.0	5.0
molecular-biology_promoters	12.0	12.0	11.3	7.1	5.0
analcatdata_boxing1	12.0	12.0	11.6	7.7	5.0
mux6	11.9	11.4	10.6	8.5	5.9
analcatdata_boxing2	12.0	12.0	12.0	11.0	7.8
corral	9.5	5.9	5.0	5.0	5.0
backache	12.0	8.1	5.0	5.0	5.0
prnn_crabs	7.5	6.1	5.0	5.0	5.0
sonar	12.0	12.0	10.9	7.0	5.2
biomed	9.6	6.5	5.1	5.0	5.0
prnn_synth	12.0	10.8	5.0	5.0	5.0
analcatdata_lawsuit	6.1	5.0	5.0	5.0	5.0
SPECT	12.0	10.3	5.7	5.0	5.0
heart-statlog	12.0	11.4	7.0	5.0	5.0
hepatitis	12.0	6.8	5.0	5.0	5.0
breast-cancer	12.0	12.0	12.0	8.2	5.1
hungarian	12.0	10.0	5.1	5.0	5.0
cleve	12.0	12.0	6.1	5.0	5.0
haberman	12.0	12.0	10.7	5.4	5.0
SPECTF	12.0	12.0	9.2	5.0	5.0
ionosphere	11.3	6.7	5.3	5.0	5.0
colic	12.0	12.0	8.4	5.0	5.0
vote	6.8	5.1	5.0	5.0	5.0
irish	6.4	5.3	5.0	5.0	5.0

Table 8. **ROC AUC Scores per Dataset over Different Thresholds for Small Datasets.** We report ROC AUC scores per dataset for 5 different thresholds. All scores are averages over 10 fold cross validation.

Dataset	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
ada	0.919 ± 0.02	0.908 ± 0.02	0.9 ± 0.02	0.899 ± 0.02	0.899 ± 0.02
churn	0.919 ± 0.03	0.885 ± 0.03	0.886 ± 0.03	0.888 ± 0.03	0.888 ± 0.03
phoneme	0.969 ± 0.01	0.969 ± 0.01	0.89 ± 0.02	0.877 ± 0.02	0.877 ± 0.02
Satellite	0.987 ± 0.02	0.972 ± 0.02	0.972 ± 0.02	0.972 ± 0.02	0.972 ± 0.02
sylvine	0.992 ± 0.0	0.971 ± 0.0	0.968 ± 0.01	0.965 ± 0.01	0.965 ± 0.01

Table 9. **Exit Layer per Dataset over Different Thresholds for Large Datasets.** We report the exit layer per dataset for 5 different thresholds when dynamically early stopping. All exit layers are averages over 10 fold cross validation.

Dataset	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
ada	12.0	6.0	5.1	5.0	5.0
churn	9.5	6.0	5.5	5.0	5.0
phoneme	12.0	12.0	6.0	5.0	5.0
Satellite	5.9	5.0	5.0	5.0	5.0
sylvine	8.0	6.1	5.5	5.0	5.0