TabDPT-Turbo: Efficient In-Context Learning for Tabular Prediction

Anonymous Authors[1]

Abstract

Tabular foundation models, driven by in-context learning, have rapidly grown in quality and popularity. However, recent approaches with either cell-based architectures or retrieval have sacrificed efficiency for raw performance, restricting their utility in situations where compute is limited or inference speed is crucial. We adopt an alternate approach, sticking with row-based attention while incorporating long context pre-training to eliminate the need for retrieval. By combining this with architectural improvements and SSL pre-training on a newly-sourced, larger corpus of real data results, we present TabDPT-Turbo, a model that provides comparable default performance to TabDPT on TabArena-Lite, CC18, and CTR23, at orders of magnitude faster.

1. Introduction

Tabular data remains the primary modality driving predictive AI in industry (van Breugel & van der Schaar, 2024). Tabular Foundation Models (TFMs), backed by in-context learning (ICL) have emerged in response, offering high-quality predictions without the need for extensive model training or hyperparameter tuning (Hollmann et al., 2023; Qu et al., 2025). However, many recent TFMs rely on expensive operations such as retrieval (Ma et al., 2025; Zhang et al., 2025b) or cell-based attention (Hollmann et al., 2025) to push performance at the cost of efficiency. This reduces their scope, limiting effectiveness in resource-constrained environments or low-latency settings.

This work instead revisits row-based TFMs (Hollmann et al., 2023). Efficiency and low resource deployment are typically easier to attain with row-based rather than cell-based architectures, particularly as context length increases. Besides the immediate benefits, enabling a more performant architecture with efficiency as a core design principle also provides the flexibility to perform inference-time adjustments such as fine-tuning. It can also produce a more practically interpretable model under real world settings, since tabular explainability methods often rely on sampling and are 2-4 orders of magnitude slower than inference, making them infeasible for slower models and larger datasets (e.g., Sundararajan et al. (2017); Castro et al. (2009)).

Considering these benefits, we directly extend TabDPT (Ma et al., 2025) as it is the highest-performing row-based architecture on the standard TabArena benchmark (Erickson et al., 2025) and is fully open-sourced. We pre-train on real data with the same Self-Supervised Learning (SSL) procedure, albeit with a corpus sourced from OpenML (Vanschoren et al., 2014) which is an order of magnitude larger than the previous pre-training set used in TabDPT. We also incorporate numerous architecture changes to improve model performance, including optimizations specifically developed for long-context training which enables us to avoid retrieval. While retrieval has been shown to improve TFM performance by providing a tailored local context to each test example (Thomas et al., 2024; Ma et al., 2024), it also greatly increases inference compute and memory requirements. Furthermore, the relevance of retrieval becomes less clear in the presence of long contexts; it has been shown in forthcoming work (Anonymous, 2026) that ICL-based TFMs learn nearest-neighbour-like behaviour internally. Our resulting model, TabDPT-Turbo, is a fast version of TabDPT with matching performance. We summarize our contributions below:

- We present TabDPT-Turbo: a retrieval free, long context, row-based TFM aimed at efficient inference.
- We provide a new pre-training data corpus based on filtered and deduplicated OpenML tables, preserving the TabDPT-style SSL objective while scaling the amount and diversity of training data.
- We introduce numerous architecture and loss changes: attention temperature scaling, per-layer target routing through transformer value projections, learned thinking rows, regression-as-classification with Continuous Ranked Probability Score (CRPS), and an auxiliary context prediction loss.
- Results on TabArena-Lite (Erickson et al., 2025), CC18 (Bischl et al., 2021), and CTR23 (Fischer et al., 2023) show better predictive performance than Tab-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

DPT at **orders of magnitude faster** inference.

2. Related Work

Tabular Foundation Models TFMs are an active area of research, becoming dominant in tabular predictive benchmarks (Erickson et al., 2025), with TabPFN variants (Hollmann et al., 2023; 2025), TabICL and TabICLv2 (Qu et al., 2025; 2026), TabDPT (Ma et al., 2025), and many other strong models being introduced recently (Zhang et al., 2025a;b; Bouadi et al., 2025). But for a given compute budget, traditional or supervised tabular models can still provide competitive trade-offs, with Random Forest (Breiman, 2001), EBM (Lou et al., 2013), CatBoost (Prokhorenkova et al., 2018), and TabM (Gorishniy et al., 2025) all appearing on TabArena Pareto curves at the time of writing. A number of major design decisions affect TFM compute usage, and so we approach them with a focus on efficiency.

Cell- vs. Row-Based Architectures Many recent TFMs opt for cell-based attention mechanisms (Grinsztajn et al., 2025; Zhang et al., 2025b) instead of the classic row-based architecture popularized by TabPFNv1 (Hollmann et al., 2023). Indeed, a cell-based tokenization is expressive for heterogeneous columns and gets closer to encoding column invariance, which is often considered a desirable property for TFMs, but sequence length scales with $n \times m$ for n rows and m columns. We instead opt for a row-based setup as a deliberate efficiency choice: since each row is one token, long contexts become more feasible and inference remains easier to batch. Qu et al. (2026) use a hybrid approach, first using a lower-dimensional cell-based transformer that essentially acts as an encoder before passing through a row-based transformer; this is still slower than fully row-based models when the number of features increases.

Retrieval Several existing models, including TabDPT, use retrieval to select instances to include in the context (Thomas et al., 2024; Ma et al., 2025; Zhang et al., 2025b). This operation allows the model to utilize more training instances, but requires a separate context for each inference instance, meaning instances cannot be batched along with a shared context. Using long contexts instead maintains approximate locality in TFMs via attention (Anonymous, 2026) and so we elect to remove the retrieval for more efficient inference.

3. Method

3.1. Data

We adopt TabDPT's general training approach as a starting point since it is an open source row-based architecture. While TabDPT was originally trained on 112 datasets, the authors argued for the potential of scaling data to improve model performance. We follow this direction while additionally deduplicating with respect to TabArena, which was not done by TabDPT.

To extend the training data, we retrieved all active datasets on OpenML (Bischl et al., 2025) and applied filtering and deduplication stages to derive our final training set. First, we used the deduplication code provided in the original TabDPT training repository to exclude all datasets that were possible duplicates or derivations of datasets in CC18, CTR23, and TabArena. Empirically, we observed that datasets with too few columns could harm performance, and therefore filtered out datasets with fewer than 10 columns. We also removed excessively large datasets to speed up training, as we did not observe any significant change in final performance when doing so. Specifically, we removed datasets with over 200 columns or a 300 MB file size. We also applied column-level filters as part of pre-processing, removing categorical columns with a cardinality over 100, and entirely constant columns. Finally, we conducted another round of duplicate detection among groups of datasets that had equal row and column count.

We reordered each candidate table to make comparisons invariant to row and column order. Rows and columns were permuted so that the largest element appeared in the top-left corner, after which the first row and first column were sorted. If two reordered tables were within a fixed tolerance, we kept only one of them. We compare our pre-training corpus with TabDPT in Table 1.

Table 1. Corpus statistics: TabDPT vs. Ours

Metric	TabDPT	Ours	Ratio
Number of datasets	112	1,445	12.90×
Total rows	32.4M	309.9M	9.56×
Total features	15.3K	43.7K	2.86×
Total cells	0.87B	5.60B	6.41×

Beyond scaling for performance, extending the training data ensures that we have a diverse range of datasets (up to 5M rows), enabling us to train the model with longer context.

Self-Supervised Setup Our SSL procedure follows TabDPT (Ma et al., 2025). There, a table is first randomly sampled, then a task is constructed from this table. A column is selected, and if it satisfies some basic quality checks, it can be used as either a regression or classification target. Some post-processing is used to introduce more diversity, such as randomizing/merging classes or applying random functions and normalizing for regression. A random subset of the remaining columns is used as features. Finally, a random subset of instances is selected to be used as context, and another as queries; the TFM is trained to predict the targets for the queries given the query features, along with the context features and targets. In this work we use context

lengths up to 32k rows.

3.2. Architecture

We use a row-based transformer architecture for efficiency. Each table row is padded with zeros to 128 dimensions and then encoded with a linear layer. This follows the TabDPT design (with 128 instead of 100) and avoids the cost of cell-level modelling, whose sequence length scales with both rows and columns. The trade-off is that the encoder is not intrinsically invariant to column order. However, the row-based design is more efficient, stable during training, and robust against representation collapse. It enables ensembling by permuting feature columns. For datasets with more than 128 features, the model must reduce the feature dimension with methods like PCA or column subsampling.

The transformer backbone is a pre-norm variant using a 512-dimensional embedding, 32 transformer layers, 8 attention heads, and SwiGLU blocks. We also prepend 64 learned thinking rows to the sequence as in Hollmann et al. (2025). These rows are soft tokens and are not passed through the linear encoder; instead, they participate in attention, providing additional latent computation. Following TabDPT's attention design, rows attend only to the context and thinking rows, preventing query-to-query information flow.

Target conditioning is injected inside each transformer layer rather than being simply added to the row embedding as opposed to TabDPT. Each layer has a small target encoder that maps context targets through an MLP to embeddings. These embeddings are concatenated to the corresponding context row representations in the attention value stream, while queries and keys remain functions of the row representations. This makes target information available to the model through values while keeping context and query row representations similar to each other.

Queries and keys are additionally normalized per head before scaled dot-product attention. The attention branch includes a learned sigmoid gate, computed per token and attention head, which multiplicatively gates the attention output before the output projection. We also scale the attention temperature as the context length grows to mitigate dispersion (Veličković et al., 2025).

The prediction head is an MLP with one hidden layer that maps the final row representation to a joint output vector with 16 classification logits and 2048 regression logits. For classification, the first entries are interpreted as class logits and normalized with a softmax over the active classes. For regression, the remaining logits parameterize a categorical distribution over uniform bins spanning 10 standard deviations around the mean (calculated from context samples). The softmax over these bins gives a predictive distribution, and the point prediction is obtained as the expected bin centre. We depict the architecture in Figure 4 in the Appendix.

3.3. Objective

For classification tasks, we train with cross-entropy over up to 16 classes, matching the first 16 logits of the joint prediction head. When a dataset has fewer than 16 classes, only the active class logits are used. For datasets with more than 16 classes during inference, prediction can be extended using the same digit-by-digit strategy as TabDPT.

For regression, targets are standardized, and the model predicts a categorical distribution over 2048 bins spanning the interval $[-10, 10]$ in standardized target space (akin to Balazadeh Meresht et al. (2025) and Hollmann et al. (2025)). We optimize the CRPS loss between the predicted cumulative distribution and the target CDF induced by the observed value. This avoids choosing an arbitrary smoothing bandwidth required by cross-entropy on the same task. The CRPS loss is a proper scoring rule (Landsgesell & Knoll, 2026) which encourages calibrated predictive distributions, offering more information than TabDPT's point estimates. At inference time, the scalar prediction is obtained as the expectation of the bin centres under the predicted distribution.

During training, the regression loss is automatically rescaled by the ratio of the two base losses to put classification and regression on a similar scale without manual tuning. We additionally use a z-loss on the query logits to prevent logit explosion and a cross-entropy loss on context rows. The context prediction loss encourages the representations used for context and query rows to remain similar.

3.4. Inference Without Retrieval

The row-based transformer, long-context training, context-dependent attention scaling, and gated attention allow the model to use large contexts. We ran the model without retrieval on datasets with up to 100k context rows, using a single shared sequence containing all context rows followed by all query rows. This removes the need to build a kNN index and avoids passing a separate context for every query point through the model. This makes inference substantially more efficient than TabDPT.

4. Experiments

We evaluate TabDPT-Turbo on TabArena-Lite, CC18, and CTR23. Since our goal is to provide efficient inference, we report the default setting without using the tuned or ensembled configurations reported by TabArena. Our evaluation employs 8 forward passes per dataset, matching the evaluation protocol used for top performing foundation models such as TabICLv2, the TabPFN family, and TabDPT.

In Figure 5 of the Appendix, we see that, on TabArena-Lite,

TabDPT-Turbo ranks fourth, trailing only the most recent state-of-the-art foundation models while outperforming TabDPT, XGBoost, TabICL, and TabPFNv2. While TabDPT-Turbo does not claim the top spot in raw prediction accuracy, it offers near-peak performance and improves substantially over TabDPT at a low inference cost. TabDPT-Turbo's performance is comparable to models that are computationally costlier or need dataset-specific training.

Even with 8 forward passes per dataset, our total time for fitting and predicting is an average of 0.76s per 1000 instances. The existing TabArena-Lite results only report k-NN, ExtraTrees, and Random Forest methods as being faster, and notably report substantially slower total times for **all** gradient boosted tree ensembles or neural network models. While our hardware configurations differed, all models with comparable performance either require supervised training or use computationally heavier architectures, making it plausible for our method to be faster.

We further compare TabDPT-Turbo against TabDPT on CC18 and CTR23 in Figure 1. For each model, we vary the number of inference passes and plot predictive performance against measured wall-clock inference time. Because TabDPT constructs a query-specific context, its inference cost grows with the number of query points and retrieved examples. In contrast, TabDPT-Turbo uses a single shared context followed by all query rows, allowing the query set to be evaluated in one efficient, batched forward pass.

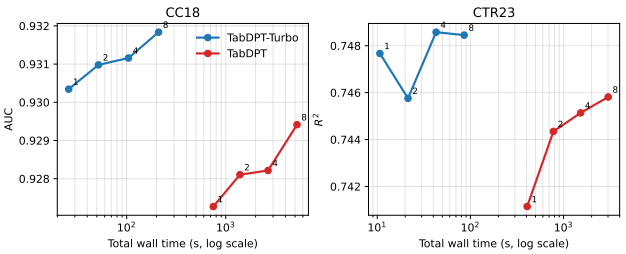

Figure 1. Performance versus total wall time for TabDPT-Turbo and TabDPT on CC18 and CTR23 with $1, 2, 4$ and 8 forward passes. $K = 2{,}048$ neighours were used for TabDPT.

4.1. Ablations and Scaling

We investigate scaling along two dimensions: context size and model capacity. Throughout, d denotes the hidden dimension, and L denotes the number of layers.

Context Scaling We trained two models of identical architecture ($d = 256$, $L = 8$) but with different maximum context sizes: one limited to 1,024 input rows, the other extending to 16,384. As shown in Figure 2, the model trained with the larger context consistently outperforms its short-context counterpart, showing the benefit of increasing context size. Note that the total number of rows per batch was identical in these two settings.

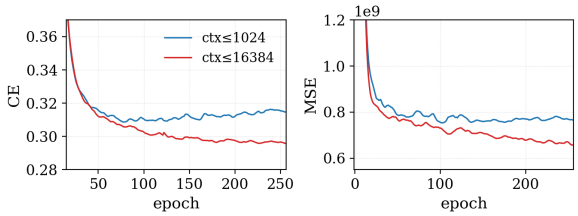

Figure 2. **Smaller vs. larger context.** Validation trajectories for two models with identical architecture but trained with different context sizes. Blue is trained on contexts of up to 1,024 rows and red up to 16,384.

Retrieval Ablation Figure 3 compares retrieval at varying sample sizes K against full-context inference on TabDPT-Turbo. On both classification and regression tasks, full context outperforms retrieval at every K except $K = 8{,}192$, and even there the margin is small (mean AUC of 0.917 vs. 0.915, mean R^2 of 0.83 vs. 0.80). Meanwhile, retrieval is at least $100\times$ slower than full-context inference, and the only configuration that surpasses full context, $K = 8{,}192$, is nearly $1000\times$ slower. This suggests that the performance gap between full-context attention and nearest-neighbour retrieval is negligible, and that retrieval no longer justifies the inference-time cost it introduces.

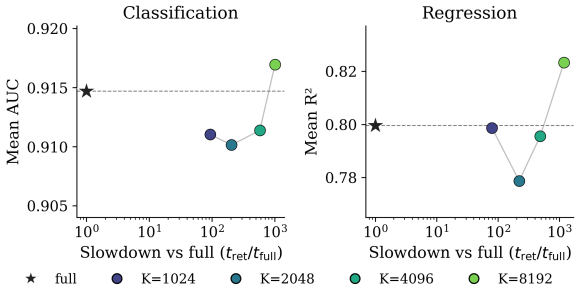

Figure 3. **Inference time vs. accuracy.** TabDPT-Turbo with no retrieval (star, dashed baseline) compared against retrieval with varying numbers of retrieved rows K (circles). The x-axis shows inference slowdown relative to full-context mode (t_{ret}/t_{full}, log scale). Left: Mean AUC on CC18 and TabArena-Lite classification tasks. Right: Mean R^2 on CTR23 and TabArena-Lite regression tasks.

5. Conclusion

While modern tabular ICL models have focused on maximizing predictive performance, this work pushes the boundaries of their efficiency. We present a row-based, retrieval-free, long-context model that enables computational trade-offs previously unavailable to neural network models.

Looking ahead, we believe this model provides a more effective starting point for scaling data and compute than TabDPT, as it is orders of magnitude more cost-effective. We aim to explore methods to further maximize its performance. We are also interested in leveraging it for understudied tabular data regimes, including datasets with large feature counts, large instance counts, or highly imbalanced data.

References

Anonymous. Omitted during review period for anonymity, 2026.

Balazadeh Meresht, V., Kamkari, H., Thomas, V., Ma, J., Li, B., Cresswell, J., and Krishnan, R. CausalPFN: Amortized causal effect estimation via in-context learning. *Advances in Neural Information Processing Systems*, 38: 154945–154984, 2025.

Bischl, B., Casalicchio, G., Feurer, M., Gijsbers, P., Hutter, F., Lang, M., Gomes Mantovani, R., van Rijn, J., and Vanschoren, J. OpenML benchmarking suites. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Bischl, B., Casalicchio, G., Das, T., Feurer, M., Fischer, S., Gijsbers, P., Mukherjee, S., Müller, A. C., Németh, L., Oala, L., Purucker, L., Ravi, S., van Rijn, J. N., Singh, P., Vanschoren, J., van der Velde, J., and Wever, M. Openml: Insights from 10 years and more than a thousand papers. *Patterns*, 6(7): 101317, 2025. doi: 10.1016/j.patter.2025.101317. URL https://www.cell.com/patterns/fulltext/S2666-3899(25)00165-5.

Bouadi, M., Seth, P., Tanna, A., and Sankarapu, V. K. Orion-msp: Multi-scale sparse attention for tabular in-context learning. 2025. URL https://arxiv.org/abs/2511.02818.

Breiman, L. Random forests. *Machine learning*, 45:5–32, 2001.

Castro, J., Gómez, D., and Tejada, J. Polynomial calculation of the shapley value based on sampling. *Computers & operations research*, 36(5):1726–1730, 2009.

Erickson, N., Purucker, L., Tschalzev, A., Holzmüller, D., Desai, P. M., Salinas, D., and Hutter, F. TabArena: A living benchmark for machine learning on tabular data. In *Advances in Neural Information Processing Systems*, 2025.

Fischer, S. F., Feurer, M., and Bischl, B. OpenML-CTR23 – A curated tabular regression benchmarking suite. In *AutoML Conference (Workshop)*, 2023.

Gorishniy, Y., Kotelnikov, A., and Babenko, A. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In *The Thirteenth International Conference on Learning Representations*, 2025.

Grinsztajn, L., Flöge, K., Key, O., Birkel, F., Jund, P., Roof, B., Jäger, B., Safaric, D., Alessi, S., Hayler, A., Manium, M., Yu, R., Jablonski, F., Hoo, S. B., Garg, A., Robertson, J., Bühler, M., Moroshan, V., Purucker, L., Cornu, C., Wehrhahn, L. C., Bonetto, A., Schölkopf, B., Gambhir, S., Hollmann, N., and Hutter, F. TabPFN-2.5: Advancing the state of the art in tabular foundation models. *arXiv: 2511.08667*, 2025.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023.

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.

Landsgesell, J. and Knoll, P. ScoringBench: A benchmark for evaluating tabular foundation models with proper scoring rules. *arXiv preprint arXiv:2603.29928*, 2026.

Lou, Y., Caruana, R., Gehrke, J., and Hooker, G. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 623–631, 2013.

Ma, J., Thomas, V., Yu, G., and Caterini, A. In-context data distillation with TabPFN. In *ICLR Workshop on Understanding of Foundation Models (ME-FoMo)*, 2024.

Ma, J., Thomas, V., Hosseinzadeh, R., Kamkari, H., Labach, A., Cresswell, J. C., Golestan, K., Yu, G., Caterini, A. L., and Volkovs, M. TabDPT: Scaling tabular foundation models on real data. In *Advances in Neural Information Processing Systems*, 2025.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, 2018.

Qu, J., Holzmüller, D., Varoquaux, G., and Morvan, M. L. TabICL: A tabular foundation model for in-context learning on large data. In *International Conference on Machine Learning*, 2025.

Qu, J., Holzmüller, D., Varoquaux, G., and Morvan, M. L. TabICLv2: A better, faster, scalable, and open tabular foundation model. *arXiv preprint arXiv:2602.11139*, 2026.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Thomas, V., Ma, J., Hosseinzadeh, R., Golestan, K., Yu, G., Volkovs, M., and Caterini, A. Retrieval & fine-tuning for in-context tabular models. In *Advances in Neural Information Processing Systems*, 2024.

van Breugel, B. and van der Schaar, M. Why tabular foundation models should be a research priority. In *International Conference on Machine Learning*, 2024.

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

Veličković, P., Perivolaropoulos, C., Barbero, F., and Pascanu, R. Softmax is not enough (for sharp size generalisation). In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=S4JmmpnSPy.

Zhang, X., Maddix Robinson, D., Yin, J., Erickson, N., Ansari, A. F., Han, B., Zhang, S., Akoglu, L., Faloutsos, C., Mahoney, M., Hu, T., Rangwala, H., Karypis, G., and Wang, Y. B. Mitra: Mixed synthetic priors for enhancing tabular foundation models. In *Advances in Neural Information Processing Systems*, volume 38, 2025a.

Zhang, X., Ren, G., Yu, H., Yuan, H., Wang, H., Li, J., Wu, J., Mo, L., Mao, L., Hao, M., et al. LimiX: Unleashing structured-data modeling capability for generalist intelligence. *arXiv preprint arXiv:2509.03505*, 2025b.

A. Appendix

A.1. Architecture Diagram

As described in the main text, the architecture uses transformer blocks as its backbone. Input features are first zero-padded to the required size and passed through a linear encoder. Thinking rows are then appended to the sequence as soft tokens. Targets are routed to the transformer value projections via per-layer MLPs rather than being added to the embedded rows. The output layer is split into classification and regression heads.

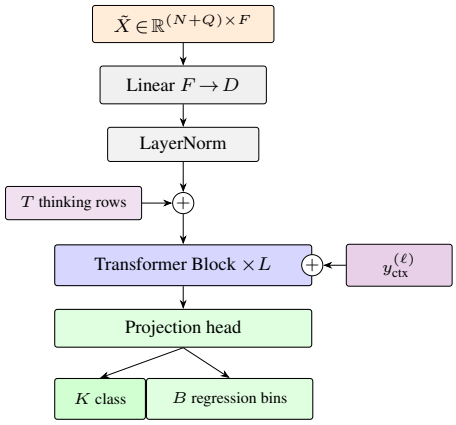

Figure 4. TabDPT architecture.

A.2. TabArena-Lite Elo score

Figure 5 provides the TabArena-Lite results for our models versus scores provided in the TabArena repository.

A.3. Model Scaling

We trained a medium-sized model with half the number of attention blocks as the full TabDPT-Turbo, giving both models a maximum context length of 32,768 for a fair comparison. As shown in Figure 6, TabDPT-Turbo (labelled "large") outperforms the medium model after roughly 500 epochs, indicating that scaling the parameter count yields meaningful gains.

We therefore see that scaling both context length and parameter count leads to significant performance improvements.

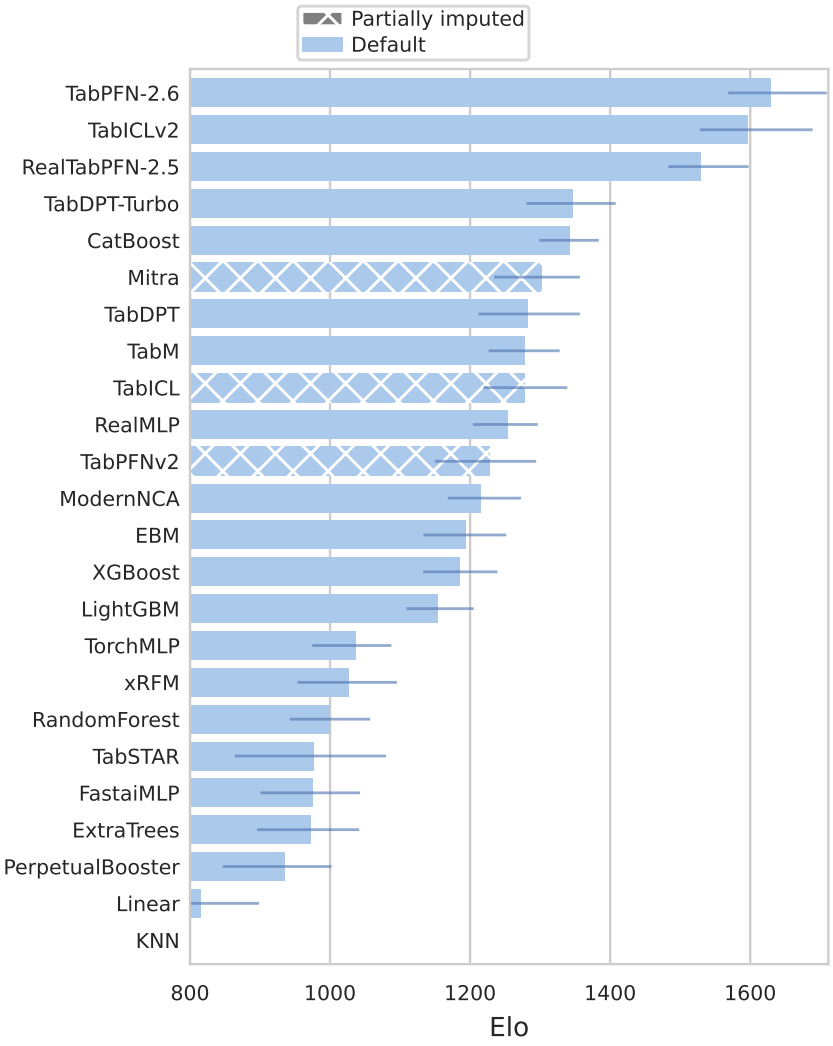

Figure 5. **TabArena-Lite Elo score comparison.** Evaluation is done with the default, non-tuned setting.

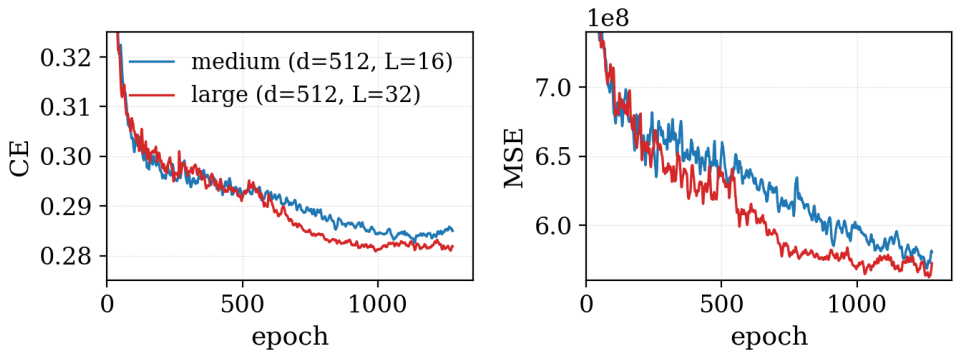

Figure 6. **Smaller vs. larger model.** Validation trajectories for the medium and large models. Left: cross-entropy loss on CC18 and TabArena-Lite classification tasks. Right: mean squared error on CTR23 and TabArena-Lite regression tasks.