TabGemma: Text-Based Tabular ICL via LLM using Continued Pretraining and Retrieval

Günther Schindler **Maximilian Schambach** **Michael Medek** **Sam Thelin**
SAP SE
{firstname.lastname}@sap.com

Abstract

We study LLMs for tabular prediction with mixed text, numeric, and categorical fields. We introduce TabGemma, a schema-agnostic in-context learner that treats rows as sequences and tackles two practical hurdles when adapting pretrained LLMs for tabular predictions: unstable numeric tokenization and limited context size. We propose to canonicalize numbers via signed scientific notation and continue pretraining of a 12B Gemma 3 model with a target imputation objective using a large-scale real world dataset. For inference, we use a compact n-gram–based retrieval to select informative exemplars that fit within a 128k-token window.

On semantically rich benchmarks, TabGemma establishes a new state of the art on classification across low- and high-data regimes and improves monotonically with more context rows. For regression, it is competitive at small sample sizes but trails conventional approaches as data grows. Our results show that LLMs can be effective tabular in-context learners on highly semantic tasks when paired with dedicated numeric handling and context retrieval, while motivating further advances in numeric modeling and long-context scaling.

1 Introduction

Many real-world tabular prediction tasks include rich textual information such as descriptive column headers, semantically meaningful categoricals or free-text columns alongside numeric and date-like features. Classical tabular predictive models such as gradient-boosted trees excel on structured inputs but typically lack fine-grained semantic understanding of such text. In practice, this gap is bridged with hand-crafted features, bag-of-words or TF-IDF vectors, or separate text encoders glued to tabular pipelines, all of which add complexity and reduce portability across schemas and domains [11].

While recent advances in tabular deep learning achieved impressive performance via end-to-end trained in-context learning (ICL), outperforming conventional approaches in some domains, as pioneered by TabPFN [13, 14] and extended in other works [22, 18], most of these methods also do not make explicit use of the semantic content within the data and rely on conventional feature encodings. Only the recent ConTextTab approach integrates semantic embeddings into a table-native ICL architecture [23] but compresses potentially large free-text cells into a single embedding vector, potentially limiting its semantic expressivity at scale.

Large language models (LLMs) offer a compelling alternative: they can consume heterogeneous data types by serializing tables as text and perform classification or regression via in-context learning, bringing strong semantic capabilities to the textual fields while handling mixed data types through a single interface. However, two hurdles hinder practical deployment for tabular prediction: raw decimals tokenize poorly and inconsistently [27], and the finite context window limits how many relevant exemplars can be provided as context, especially in large datasets. In the past, this has severly limited performance of LLM-based tabular predictors.

AI for Tabular Data workshop at EurIPS 2025

Building on prior work that serializes tables for LLMs and using retrieval-augmented generation to scale ICL, we present TabGemma, a schema-agnostic method that treats tabular prediction as sequence modeling, improves numeric tokenization, and selects informative exemplars via efficient retrieval at inference. Rows are serialized with dedicated separators and numerics are canonicalized into signed base-10 scientific notation, yielding stable token patterns. We continue pretraining of a 12B Gemma 3 [24] model on a column-imputation objective that applies loss only to target-column tokens while conditioning on all feature tokens, aligning next-token prediction with classification/regression. To fit task-relevant support within the context window at inference, we perform nearest-neighbour retrieval using compact hashed character n-gram embeddings per cell, concatenated into row embeddings and indexed with FAISS [3]: at inference, we retrieve k similar rows, serialize them as exemplars, and append the query row with an empty target for the model to decode.

2 Related Work

Tabular deep learning: Prediction on tabular data has long been dominated by boosted trees such as XGBoost, LightGBM, and CatBoost [1, 16, 21]. While strong, these models need to be trained per dataset, cannot benefit from cross-task pretraining, and often require manual feature engineering and extensive hyperparameter optimization. Early deep learning architectures like FT-Transformer [8] and XTab [28] explored transformer-based encoders, while only more recent methods, e.g. TabR, RealMLP, CARTE, TabM, or ModernNCA [9, 15, 17, 10, 26], report consistently competitive, sometimes superior results to boosted trees.

In-context learning for tabular data: TabPFNv1 [13] demonstrated that row-level ICL pretrained on synthetic tasks can outperform boosted trees on small classification problems, eliminating per-task training and hyperparameter tuning. Using real data and retrieval to select context examples, TabDPT achieved similarly strong results and extended the setting to regression, building on ideas also investigated in TabR. Moving beyond row-level encodings, cell-based ICL, as used in TabPFNv2 [14], TabICL [22], and ConTextTab [23] scale to larger datasets and report state-of-the-art results.

Semantics and real data: Capturing fine-grained semantics in real-world tables is key for transfer beyond statistical patterns. CARTE [17] pretrains across diverse sources to model table semantics and achieves state of the art on its benchmark, but requires task-specific fine-tuning. Modern LLMs bring stronger semantic understanding and world knowledge but lack native table support. Several works adapt LLMs to tabular ICL, for example TabLLM, LIFT, and TabuLa-8B [12, 2, 7], which also curates the T4 dataset – a collection of 3M tables derived from TabLib [4] – and show excellent performance in the very low-data regimes.

3 Methodology

Similar to TabuLa, we cast tabular classification and regression as sequence modeling: the input table is serialized into tokens, and a long-context LLM is trained to predict a designated target column causally conditioned on feature tokens. The method is schema-agnostic and centers on three components: canonical row serialization, continued pre-training with a target-imputation objective, and similarity-based retrieval to fit task-relevant exemplars within the context window at inference. An overview of our proposed approach is depicted in Figure 1.

Table serialization: We serialize each table row into a linear token sequence. Every cell is first cast to a canonical string, tokenized, and concatenated in column order. Cells are separated by a dedicated cell-separator token. Each row is terminated with an end-of-row token, which conditions the model to stop decoding once the target has been produced. The input to the model is the concatenation of such row sequences. In particular, we do *not* prepend a task-specific natural-language instruction for the LLM. Numeric values are normalized before tokenization using signed, base-10 scientific notation with four significant digits. For example, 3141.592 becomes +3.1416e+03. This canonicalization is locale-independent and reduces variability. In subword tokenizers, scientific notation induces reusable subpatterns (e.g., "+", "e+0"), which improves the learning of token-based numeric embeddings compared to raw decimals, whose lengths and delimiters vary widely.

Continued pretraining: We initialize from the pretrained Gemma 3 12B checkpoint (without instruction-tuning), which supports a 128k-token context window, and continue pretraining the model on a tabular imputation objective tailored to classification and regression over tables using the large-

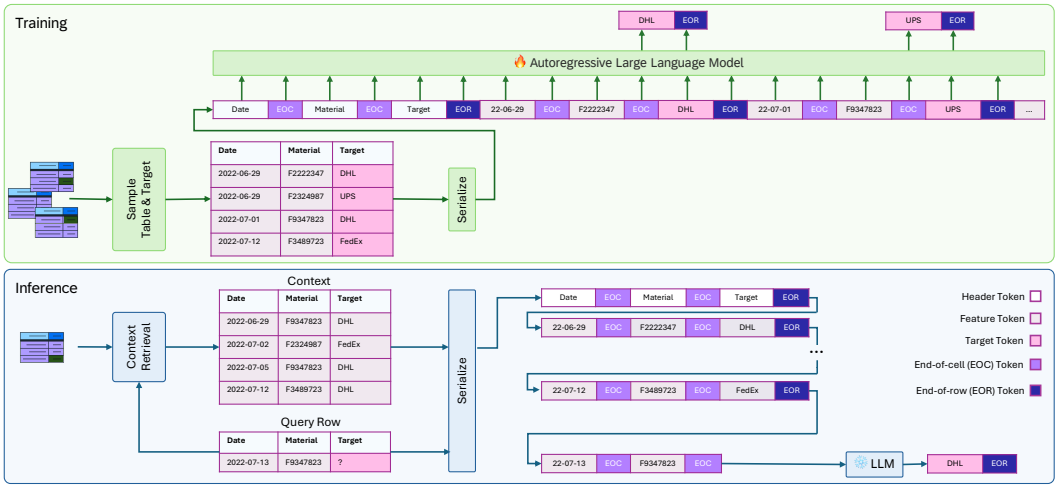


Figure 1: Illustration of our proposed LLM-based tabular prediction architecture with table serialization and target-imputation objective at training, and local context retrieval at inference.

scale T4 table corpus [7]. For each training step, we draw a table from the corpus, uniformly sample 256 rows, and designate one column as the prediction target while using the remaining columns as conditioning features. The input consists of all selected rows with their full feature columns and the target column values present in the text for teacher forcing. We compute token-level cross-entropy only over the tokens belonging to the target column, masking out loss contributions from feature columns. The prompt format scales naturally to different numbers of rows at inference time, allowing the model to condition on variable-sized contexts. Note that, due to the autoregressive nature of LLMs and the use of causal attention masking, *every* target cell is used as a training sample and the approach does not require a fixed context-query split unlike table-native approaches such as TabPFN – practically increasing the effective batch size as well as conditioning the model on different effective context lengths within a single training step.

Inference & Retrieval: Scaling in-context learning to larger tables is constrained by the model's context window. As compared to table-native architectures, this becomes even more pronounced when adapting LLMs for tabular ICL due to the relatively less efficient tokenization scheme. To mitigate this limitation, we employ similarity-based context retrieval to select a small, task-relevant support set for each query at inference. Our approach is similar to that of TabR and TabDPT but note that we use it during inference only while we found it sufficient to randomly sample context during training. We use nearest-neighbour retrieval by constructing compact row embeddings via hashing: Each table row is serialized as a sequence of cells, and each cell is independently vectorized using a bag of character n-grams. The n-grams are hashed into a fixed 256-dimensional vector per cell. The per-cell vectors are concatenated to form the row embedding. This representation is stateless, memory-efficient, and offers parallelization over rows, enabling streaming ingestion and scaling to millions of training rows. For similarity search, we L2-normalize embeddings and index them with FAISS [3]. The retrieved rows are then formatted as ICL exemplars and provided to the LLM.

4 Experiments

Training: We train for 2,500 steps with a batch size of 64, corresponding to roughly 41 million row predictions in total. We uses Adam with a learning rate of 10^{-5}. We do not apply dropout or weight decay. To balance throughput and context utilization, we cap inputs at 16k tokens during training and truncate sequences that exceed this limit. We trained the model for 20 days on 2 H100 GPUs.

Evaluation: We evaluate our approach on several benchmarks: the CARTE benchmark [17], the recently proposed TextTab benchmark [19], as well as the recent TabArena benchmark [6] in its "lite" variant (evaluating a single fold). All benchmarks are mixed classification and regression benchmarks. Whereas CARTE and TextTab are constructed to emphasize semantic features and are thus our focus, TabArena is a more conventional, numerics-heavy benchmark which we include for completeness.

Table 1: Evalation results on the investitated benchmarks sorted by classification performance on CARTE. We report accuracy (Acc) and (soft-clipped) R2 for classification and regression, respectively.

	All		CARTE		TextTab		TabArena	
Model	**Acc**	**R^2**	**Acc**	**R^2**	**Acc**	**R^2**	**Acc**	**R^2**
TabGemma (ours)	83.6	60.7	**79.3**	70.3	84.1	31.6	84.8	57.8
AutoGluon	**85.9**	**70.5**	78.9	**73.4**	83.9	51.8	88.5	78.9
ConTextTab [bagging=8]	85.0	70.3	76.9	72.4	**84.3**	**55.0**	87.6	77.9
ConTextTab [bagging=1]	84.9	69.6	76.4	71.5	84.1	54.4	87.5	77.6
RealMLP [HPO-CV, ens.]	84.6	67.6	73.6	68.2	81.9	52.2	**88.5**	**79.8**
LGBM [HPO-CV]	84.4	66.3	73.4	67.5	81.5	48.6	88.2	78.9
TabPFN	83.2	63.2	72.3	65.0	81.6	41.9	86.7	77.0
Random Forest	83.3	62.5	71.5	63.3	79.8	45.3	87.6	76.3
Naive	70.1	-3.5	53.0	-1.8	70.4	-5.0	75.0	-7.3

We compare against a range of extensively tuned conventional as well as deep learning and ICL baselines, including LGBM, RealMLP, TabPFN, ConTextTab, Random Forest, and a naive predictor. Additionally, we include results of the AutoML framework AutoGluon [5]. The details about the used baselines can be found in Appendix C.2. Unfortunately, despite much effort spent, we were not able to run LLM-based baselines such as TabuLa [7] or GTL [25] due to non-functional reference implementations or problems to evaluate them at scale.

Results: The main results are summarized in Table 1. Here, we report TabGemma results using $k = 128$ retrieved context examples by default. To confirm the efficacy of our approach, we also evaluated an off-the-shelf Gemma 3 model on CARTE. In the absence of prompt engineering, performance is poor, 4.6% accuracy and -98.4 R^2, and was hence excluded. On the semantically rich CARTE and TextTab benchmarks, TabGemma matches or surpasses state-of-the-art baselines in classification performance. To the best of our knowledge, this is the first time an LLM-based approach outperforms extensively tuned baselines, as well as AutoGluon, which stacks a multitude of per-dataset tuned predictors. Note that evaluation is done at the full scale of the available datasets.

However, TabGemma lags behind on regression and on TabArena generally. However, on CARTE, regression performance of TabGemma still surpasses extensively tuned LGBM, as well as tuned and ensembled RealMLP. The poor regression performance on TextTab is surprising and needs to be investigated more closely in the future. As TabArena focuses on conventional, numerically dominated tasks, this underscores current limitations of language models in this regime and highlights further potential in tokenization. While weaker regression performance is expected, our rank-based analysis further indicates that retrieval and long-context handling degrade as the number of rows and columns increases. We investigate this in more detail in the Appendix A.

5 Discussion and Conclusion

We presented TabGemma: an LLM-based in-context learner that combines improved numeric representation, retrieval, and continued pretraining over a large corpus of real-world tables. On semantically rich benchmaks, TabGemma delivers strong classification in both low- and high-data regimes, outperforming extensively tuned baselines and AutoML solutions, while revealing gaps on regression and on wide or very large tables due to context and retrieval constraints. While shining in the few-shot regime, to the best of our knowledge, this is the first time an LLM-based tabular predictor outperforms baselines also at full dataset evaluation.

There are a several limitations of our current approach opening a multitude of future research directions: first, we observe that TabGemma is very effective at classification but underperforms in regression as well as tasks with numeric-heavy features. This motivates further research into numerics-adapted tokenization. Second, performance on very large and wide tables degredates. Further research into more compact tokenization schemes as well as even more efficient retrieval approaches may mitigate this. And last, note that the proposed table serialization and autoregressive modeling breaks the natural column and row order permutation equivariance inherent to many tabular prediction problems. The effect of this on tabular language modeling needs to be investigated more closely openining potential gains when ensembling over permuted inputs.

Acknowledgements

We would like to thank Johannes Hoehne, Johannes Hoffart, and Markus Kohler for their insightful comments and suggestions throughout the development of this work. We thank Thassilo Klein for providing valuable feedback on the draft of this contribution.

References

[1] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, 2016. ISBN 978-1-4503-4232-2.

[2] Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. LIFT: Language-interfaced fine-tuning for non-language machine learning tasks. *Advances in Neural Information Processing Systems*, 35:11763–11784, 2022.

[3] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The FAISS library. *arXiv preprint arXiv:2401.08281*, 2024.

[4] Gus Eggert, Kevin Huo, Mike Biven, and Justin Waugh. TabLib: A dataset of 627 M tables with context. *arXiv preprint arXiv:2310.07875*, 2023.

[5] Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. AutoGluon-Tabular: Robust and accurate AutoML for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

[6] Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, Frank Hutter, et al. TabArena: A living benchmark for machine learning on tabular data. *arXiv preprint arXiv:2506.16791*, 2025.

[7] Joshua P Gardner, Juan Carlos Perdomo, and Ludwig Schmidt. Large scale transfer learning for tabular data via language modeling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[8] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943, 2021.

[9] Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. TabR: Tabular deep learning meets nearest neighbors. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In *International Conference on Learning Representations*, 2025.

[11] Léo Grinsztajn, Edouard Oyallon, Myung Jun Kim, and Gaël Varoquaux. Vectorizing string entries for data processing on tables: when are larger language models better? *arXiv preprint arXiv:2312.09634*, 2023.

[12] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. TabLLM: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR, 2023.

[13] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023.

[14] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

[15] David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned MLPs and boosted trees on tabular data. *Advances in Neural Information Processing Systems*, 37:26577–26658, 2024.

[16] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, 2017.

[17] Myung Jun Kim, Leo Grinsztajn, and Gael Varoquaux. CARTE: Pretraining and transfer for tabular learning. In *Forty-first International Conference on Machine Learning*, 2024.

[18] Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Hamidreza Kamkari, Alex Labach, Jesse C Cresswell, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony L Caterini. TabDPT: Scaling tabular foundation models. *arXiv preprint arXiv:2410.18164*, 2024.

[19] Martin Mráz, Breenda Das, Anshul Gupta, Lennart Purucker, and Frank Hutter. Towards benchmarking foundation models for tabular data with text. In *ICML 2025 Workshop on Foundation Models for Structured Data (FMSD)*, 2025.

[20] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: Unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 2018.

[22] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. *arXiv preprint arXiv:2502.05564*, 2025.

[23] Marco Spinaci, Marek Polewczyk, Maximilian Schambach, and Sam Thelin. ConTextTab: A semantics-aware tabular in-context learner. *arXiv preprint arXiv:2506.10707*, 2025.

[24] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

[25] Xumeng Wen, Han Zhang, Shun Zheng, Wei Xu, and Jiang Bian. From supervised to generative: A novel paradigm for tabular deep learning with large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3323–3333, 2024.

[26] Han-Jia Ye, Huai-Hong Yin, De-Chuan Zhan, and Wei-Lun Chao. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. In *International Conference on Learning Representations*, 2025.

[27] Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, and Songfang Huang. How well do large language models perform in arithmetic tasks? *arXiv preprint arXiv:2304.02015*, 2023.

[28] Bingzhao Zhu, Xingjian Shi, Nick Erickson, Mu Li, George Karypis, and Mahsa Shoaran. XTab: Cross-table pretraining for tabular transformers. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

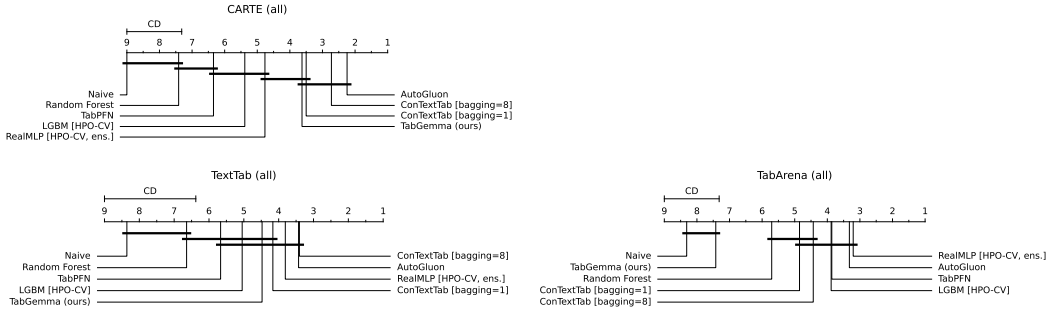
A Additional results

A.1 Critical difference diagrams

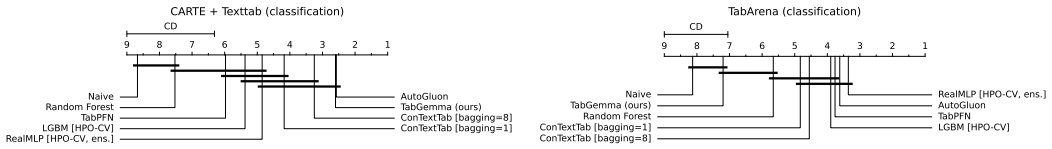
We provide critical difference (CD) diagrams in Figure 2. To this end, we use the `autorank` package[1]. We calculate CD-diagrams on the full benchmark as well as on classification- and regression-only subsets. Note that, since the support set for classification tasks on CARTE and TextTab were too small to calculate the CD, we show a joined evaluation on the two instead.

We observe that TabGemma performs SOTA on semantic classification tasks within CARTE and TextTab, on par with AutoGluon. While its regression performance lags behind here, it is not statistically significantly worse then other state-of-the-art approaches such as ConTextTab or RealMLP.

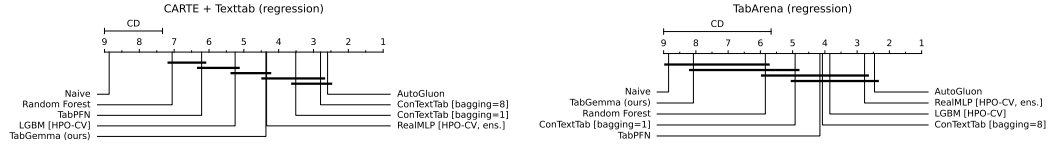
On Tabarena, the performance of TabGemma is noticeably worse, statistically on par with a non-tuned Random Forest predictor. This shortcoming on non-semantic tasks opens future research directions.



(a) All tasks.



(b) Classification tasks only.



(c) Regression tasks only.

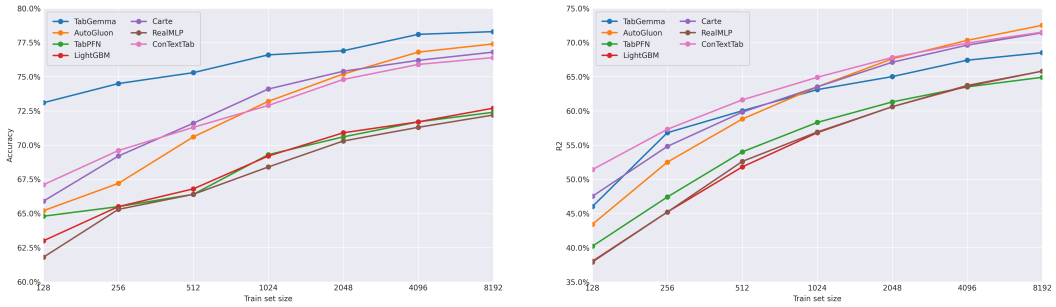
Figure 2: Critical difference diagrams on the investigated benchmarks, including all datasets ("all") as well as classification- and regression-only subsets. Note that, due to the limited support of classification tasks on CARTE and TextTab, evaluation was performed over the union of tasks in this case.

[1] github.com/sherbold/autorank

A.2 Sample efficiency and few-shot domain

Figure 3 examines performance as a function of available training data. To this end, using the CARTE benchmark datasets, we subsample each training set to 128, 256, . . . , 8192 rows and evaluate each model under the same subset: baselines are trained on that subset, whereas TabGemma receives the same subset as its retrieval pool and in-context exemplars. In low-data regimes, TabGemma is highly competitive on classification and surpasses the baselines by a notable margin. This observation is in line with previous LLM-based results such as TabuLa [7] but stretches to much larger shot examples, whereas Tabula reported results only in the very few-shot domain of up to 32 samples. In the few-shot domain and particularly for highly semantic tasks, LLMs likely benefit most from their world knowledge obtained during their extensive pretraining.

For regression, TabGemma keeps pace up to roughly 1024 training examples but lags behind AutoGluon and CARTE once more data are available.



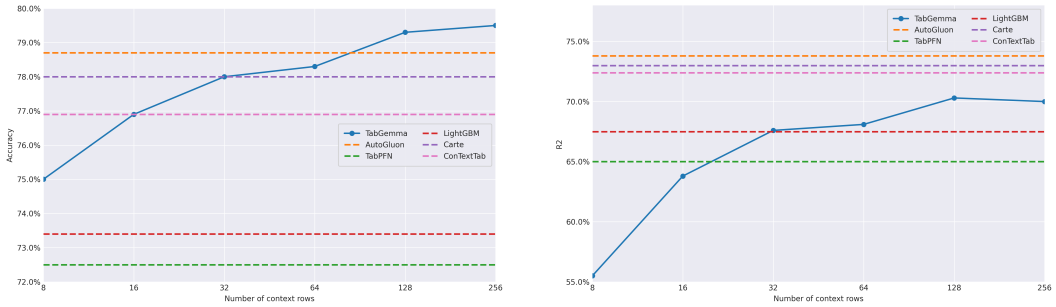
(a) Average Accuracy on classification.

(b) Average R2 on regression.

Figure 3: Results on the CARTE benchmark for varying train subsets.

A.3 Scalability with context size

Next, we studiey how TabGemma scales as we increase the number of context rows, i.e., the number of retrieved training examples included in the prompt. The results are depicted in Figure 4. Dashed lines denote baselines trained on the full training split (or, for ICL-style baselines, using the full training pool as their retrieval source). Solid lines denote TabGemma with varying number of k context rows and no gradient updates. As k grows, TabGemma improves monotonically and establishes a new state-of-the-art on classification within CARTE. On regression, however, AutoGluon, CARTE and ConTextTab remain ahead, highlighting a current limitation of our numeric handling for continuous targets. Nevertheless, TabGemma still outperforms an extensivly tuned and ensembled LGBM predictor.



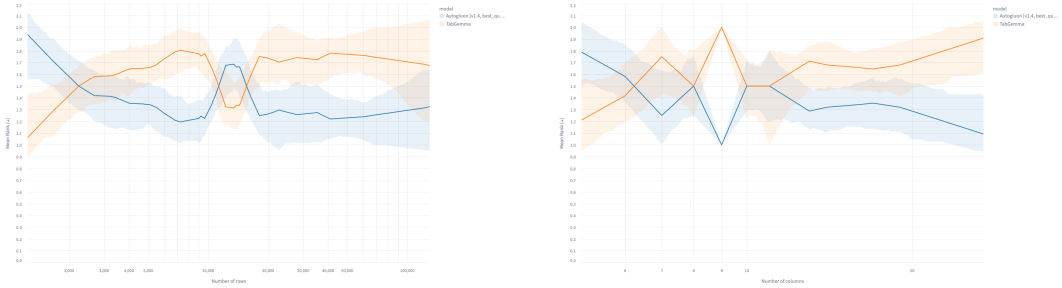
(a) Average Accuracy on classification.

(b) Average R^2 on regression.

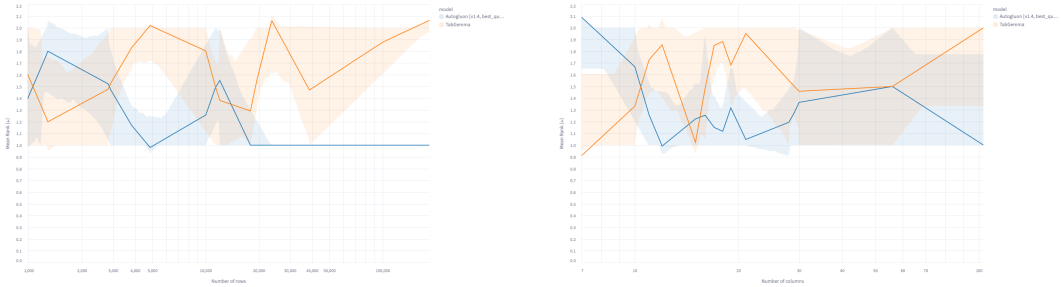
Figure 4: Results on the CARTE benchmark for varying context rows.

B Limitations across task scale and dimensionality

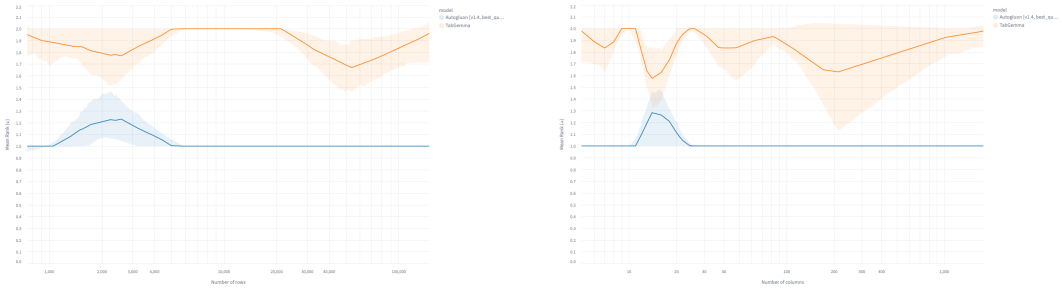
Figure 5 reports mean rank (lower is better) for TabGemma versus AutoGluon across CARTE, TextTab, and TabArena, stratified by training-set size and the number of columns. AutoGluon consistently leads in regimes with more training rows and wider tables, and the gap widens as scale increases. These trends suggest two main bottlenecks for TabGemma: (1) retrieval precision declines as the candidate pool grows, reducing the quality of in-context examples; and (2) the 128k-token context window limits how many rows and columns can be represented without lossy compression, weakening long-context modeling.



(a) CARTE.



(b) TextTab.



(c) TabArena.

Figure 5: Dependence of mean rank (lower is better) of AutoGluon (blue) vs. TabGemma (orange) on the number of rows (left) and number of columns (right) across different evaluated benchmarks.

C Evaluation datasets and baselines

C.1 Datasets

As previously discussed, we evaluate all methods on the CARTE, TextTab, and TabArena-Lite benchmark covering 123 tasks in total. We use the CARTE data from the official reference implementation[2]. CARTE contains 51 tasks in total, 11 exclusively binary classification tasks and 40 regression tasks. We create custom 80:20 train target splits using the target column for stratification. TextTab contains 21 tasks in total, 9 mostly binary classification tasks and 12 regression tasks. We use all tasks from the original publication [19] as well as the "extra datasets" given in the reference implementation[3]. Again, we create custom stratified 80:20 train test splits. TabArena contains 51 tasks in total, 38 mostly binary classification tasks and 13 regression tasks. We use the splits as defined in the official OpenML release[4] of the benchmark's lite variant covering the first fold only.

The task size distribution of all evaluated benchmarks is depicted in Figure 6.

C.2 Baselines

Throughout, we follow the evaluation protocol of [23]. That is, we use a AutoGluon-based standardized feature encoder for all baselines that do not provide a custom one. In particular, the encoder natively handles categorical data, free text (via conventional NLP features), as well as datetime encoding. In particular, we evaluate the following model versions.

Pytabkit models: We use the `pytabkit` [15] for evaluating RealMLP, and LightGBM. We evaluate LightGBM with ensembled hyperparameter optimization across 5-fold inner cross-validation (HPO-CV). For RealMLP, we do the same but combine it with the recently introduced learned ensemble, further pushing its performance (HPO-CV, Ens.). For the HPO variants, we use the recently added `tabarena` search spaces proposed in [6].

TabPFN: We use the model from the official `tabpfn` package at version 2.1.0 with the `tabpfn-extensions` package version 0.1.0. For datasets larger than the native 10 k limit of TabPFN, we sample a random 10 k subset of the training split. For datasets with more than the 500 feature limit, we select a random subsample of 500 features.

ConTextTab: We evaluate ConTextTab v1.0.1 using the reference implementation and checkpoint[5]. We set a context size of 8k samples and evaluate variants without and with 8-fold bagging.

CARTE: We use the model provided in the official `carte-ai` package with version 0.0.26. We use `CARTEClassifier` and `CARTERegressor` with default parameters for classification and regression tasks, respectively.

Sklearn models: We evaluate the Random Forest and Naive baseline models from `scikit-learn` [20], combining them with the default preprocessor as outlined above. Evaluation is performed using `scikit-learn` v1.5.2.

For the naive predictor, we use the `DummyClassifier` and `DummyRegressor` to predict the most frequent, respectively mean value of the train splits as the naive majority baseline.

For the random forest predictor, we use the `RandomForestClassifier` and `RandomForestRegressor` for classification and regression tasks, respectively, using default hyperparameters. The model handles missing values natively.

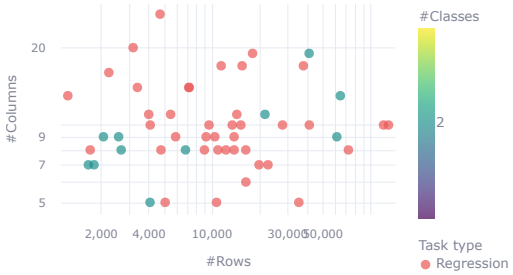
AutoGluon: We evaluate using AutoGluon v1.4 with its native feature encoder. We use the `best_quality` preset with a per-dataset time limit of 4 h. (We have found the "extreme" preset to only yield slightly better results at the expense of much higher compute requirements).

[2] github.com/soda-inria/carte

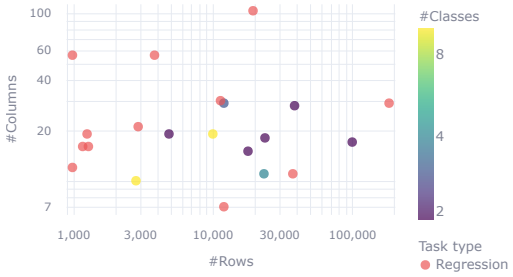
[3] github.com/mrazmartin/TextTabBench

[4] openml.org/search?type=study&study_type=task&id=457

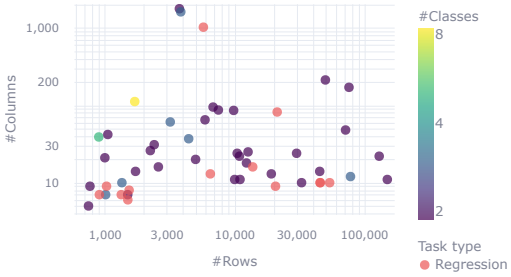
[5] github.com/SAP-samples/contexttab



(a) CARTE.



(b) TextTab.



(c) TabArena.

Figure 6: Dataset statistics of the evaluated benchmarks.