Enhancing Tabular Learners with Context-Aware Semantic Embeddings

Günther Schindler **Maximilian Schambach** **Johannes Höhne**
SAP SE
{firstname.lastname}@sap.com

Abstract

While modern tabular learners excel at capturing statistical patterns, they frequently operate in a semantic vacuum, treating textual features as discrete symbols, ignoring the rich semantics inherent in feature names or cell entries. We propose **CASE** (**C**ontext-**A**ware **S**emantic **E**mbeddings), a novel framework that bridges the gap between the semantic understanding of Large Language Models (LLMs) and the statistical capabilities of tabular learners. Unlike existing methods that embed rows in isolation, CASE utilizes a contextualization strategy: we pre-fill the KV cache of a custom-trained Gemma 3-based Tabular Language Model with a representative sample of rows to establish a persistent anchor of the dataset's semantics. This ensures that generated row embeddings are dynamically contextualized, resolving semantic ambiguities and anchoring representations in domain-specific context. Our experiments across several benchmarks (CARTE, TextTab, and TabArena) demonstrate that CASE substantially improves the performance of tabular learners on semantically rich datasets, particularly in low-data regimes. Inference code is available at: `https://github.com/SAP-samples/case` .

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. The dominance of tabular learners such as Gradient Boosted Decision Trees (GBDTs) and recent in-context learners like TabPFN [16] stem from their ability to find statistical patterns in the feature space. However, these models operate in a semantic vacuum: they rely entirely on the provided features to infer relationships, often treating categorical values as discrete symbols and lacking native support for free-text columns. In contrast, Large Language Models (LLMs) possess vast world knowledge acquired during pre-training, allowing them to recognize concepts, hierarchical relationships, and domain-specific nuances that may not be explicitly found in a single table. However, one key drawback hinders using LLMs as tabular predictors natively: table serialization and tokenization is comparably inefficient, resulting in very long context-sequences even for relatively small table sizes, limiting how many relevant rows can be provided as context, especially for large datasets. In the past, this has severely limited the performance of LLM-based tabular predictors.

The challenge lies in effectively combining the semantic power of LLMs with the statistical reasoning of tabular learners. Conventional approaches either use heuristic featurization of string or text columns, such as TF-IDF or other n-gram features as used in AutoGluon or the Skrub library, or generate cell- or row-level embeddings in isolation [28, 20]. While useful, these methods suffer from a lack of distributional semantic awareness: A single row processed in a vacuum lacks the anchor points necessary to resolve ambiguity or interpret values relative to the rest of the dataset.

We introduce **CASE** (**C**ontext-**A**ware **S**emantic **E**mbeddings) to bridge this gap. First, we train a custom Tabular Language Model (TLM) via continued pretraining of a series of Gemma 3 mod-

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

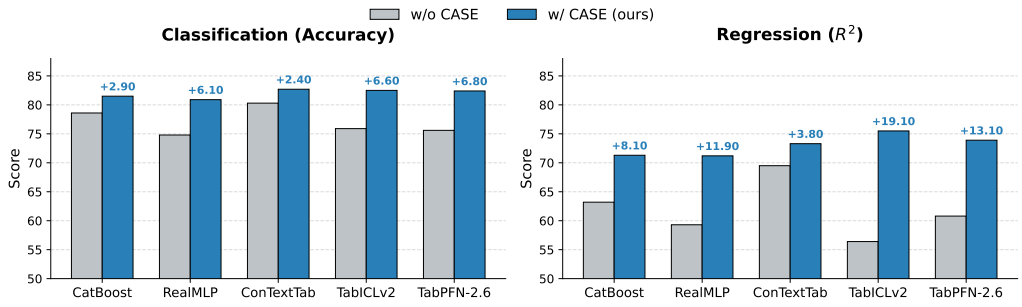


Figure 1: Our proposed **Context-Aware Semantic Embeddings (CASE)** boost prediction quality of tabular learners on semantically rich benchmarks (CARTE [19] and TextTab [23]).

els [29] using a column-imputation objective. We apply the loss only to target-column tokens while conditioning on all feature tokens, aligning next-token prediction with tabular prediction. This yields task-specific representations that capture both the semantic structure of the table and the underlying target distribution. Next, utilizing this semantic embedder, we employ a "Context Priming" strategy: By pre-filling the model's KV cache with a representative sample of rows, including feature as well as target columns from the table's train split, we establish a persistent semantic manifold tailored to the specific dataset. This primed state ensures that when an individual row is embedded, its representation is not merely a translation of its strings, but a contextualized vector positioned relative to the table's global semantics and predictive task.

We then combine the obtained embeddings with tabular learners to achieve both statistical and semantic grounding: thus, our approach offers a best-of-both-worlds paradigm: the table-tuned LLM acts as a sophisticated feature engineer with semantic grounding, while the downstream statistical learner performs the heavy lifting of statistical pattern detection and prediction. By projecting the high-dimensional CASE into a lower dimensional space via Principal Component Analysis (PCA), we provide a computationally efficient way to infuse world reasoning into any tabular pipeline.

2 Related Work

Tabular Learning Baselines: Tabular prediction has historically been dominated by GBDTs such as XGBoost, LGBM, or CatBoost [3, 18, 24]. While robust, these models lack cross-task transferability and require significant per-dataset hyperparameter optimization and retraining. Early deep learning architectures like FT-Transformer [10] and XTab [32] explored transformer-based encoders, but only recent deep learning architectures, such as CARTE, RealMLP, or TabM [19, 17, 11], have achieved parity with GBDTs, however, also require re-training and hyperparameter tuning on each table.

In-Context Learning (ICL): Tabular ICL was pioneered by TabPFN [15], which showed transformers could perform in-context classification on small datasets. This paradigm has evolved from row-level encodings to more scalable cell-based methods like TabPFNv2, TabICL, Mitra, and ConTextTab [16, 25, 26, 31, 28]. While these models excel at structural pattern matching, they often treat values as abstract tokens, overlooking the latent semantic richness inherent in the data. While ConTextTab uses semantic embeddings natively, these embeddings are obtained cell-wise using a small sentence embedder and are subsequently contextualized by the ICL model. The model itself, however, is not trained on semantic prediction tasks like LLMs but rather on in-context learning target prediction which may limit its ability for advanced semantic reasoning .

LLMs and Table Semantics: Models like TabLLM, LIFT, TabuLa, and TabGemma [13, 5, 9, 27] leverage the world knowledge of LLMs to add semantics to tabular predictions. These methods excel in low-data regimes where semantic priors compensate for sparse signals, though they often struggle with numeric-heavy datasets and longer context sizes. Fundamentally, due to the comparably inefficient tokenization, even recent large-context window LLMs may not fit a full table into their context, while table-native ICLs now scale to 10's or 100's of thousand context rows. Other works have also explored LLM- or knowledge graph-based enrichment of conventional tabular learners [19, 20] or experimented with creating target-aware embeddings using per-table tuned text encoders [1].

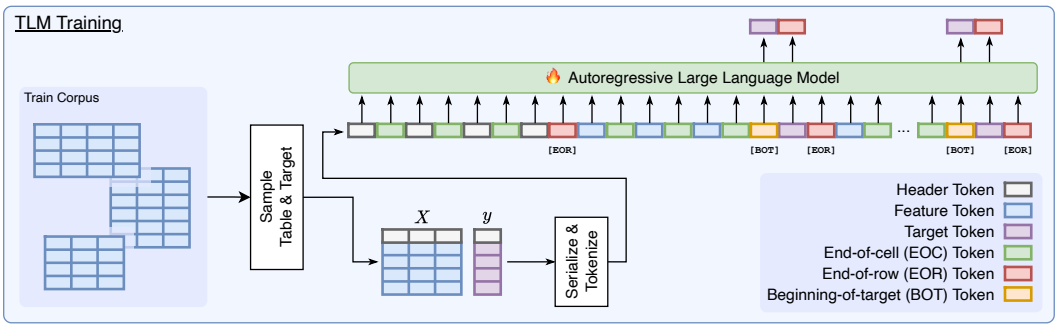


Figure 2: Illustration of our proposed Tabular Language Model (TLM) architecture with table serialization and target-imputation objective during training.

3 Methodology

We introduce CASE, a framework that enhances tabular learners with context-aware semantic embeddings. At its core, we develop a Tabular Language Model (TLM) derived from the Gemma 3 decoder-only family, pretrained via a custom table serialization and an in-context target prediction objective. By leveraging this TLM, we extract row representations that are contextualized, i.e. conditioned on the table's specific semantic distribution and its predictive task. These semantic embeddings are subsequently compressed and fused with the native features to be used together with any downstream tabular learner, combining the rich world knowledge of language models with the statistical precision and predictive capabilities of specialized tabular architectures.

3.1 Tabular Language Model Pretraining

To extract context-aware row representations, we perform continued pretraining on a diverse suite of tabular prediction tasks synthesized from a large-scale collection of real-world datasets. Following the paradigm of TabuLa [9], we frame tabular data as a conditional sequence modeling problem. Tables are serialized into token sequences where a long-context LLM is trained to predict a designated target column, causally conditioned on the preceding feature tokens and the broader dataset context. By casting tabular prediction as an autoregressive generative process, the model learns to synthesize its internal world knowledge with observed statistical patterns. This results in embeddings that are inherently task-aligned and grounded in the table's specific semantic. An overview of this pretraining architecture is illustrated in Figure 2.

Table Serialization and Tokenization: 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, and each row is terminated with an end-of-row token.

That is, for a table consisting of F features X and a target y with N rows, we define a serialization function $\mathcal{S}(\cdot)$ that transforms a table into a continuous token sequence, beginning with the header sequence $\mathbf{H}$, which establishes the semantic identity of the columns. The header sequence is obtained by concatenating the tokenized column headers c_k, adding end-of-cell (EOC) tokens in between and finalizing the sequence with an end-of-row (EOR) token:

$$\mathbf{H} = c_1 \parallel \texttt{[EOC]} \parallel c_2 \parallel \texttt{[EOC]} \parallel \cdots \parallel c_{F+1} \parallel \texttt{[EOC]} \parallel \texttt{[EOR]}\,, \tag{1}$$

where $\parallel$ denotes concatenation along the sequence axis. Following the header, each row r_i of the table is serialized by concatenating the tokenized feature values $v_{i,k}$ and the final target value y_i, separated by the Beginning-of-Target (BOT) marker:

$$\mathcal{S}(r_i) = v_{i,1} \parallel \texttt{[EOC]} \parallel \cdots \parallel v_{i,F} \parallel \texttt{[EOC]} \parallel \texttt{[BOT]} \parallel y_i \parallel \texttt{[EOR]}\,. \tag{2}$$

The total input sequence $\mathbf{S}$ obtained from the table is the concatenation of the header sequence and all row sequences:

$$\mathbf{S} = \mathbf{H} \parallel \mathcal{S}(r_1) \parallel \cdots \parallel \mathcal{S}(r_N)\,. \tag{3}$$

Throughout, numerical targets are normalized into a scientific notation (for example, 3141.592 becomes +3.1416e+03) and other data types are natively handled by the LLM's tokenizer.

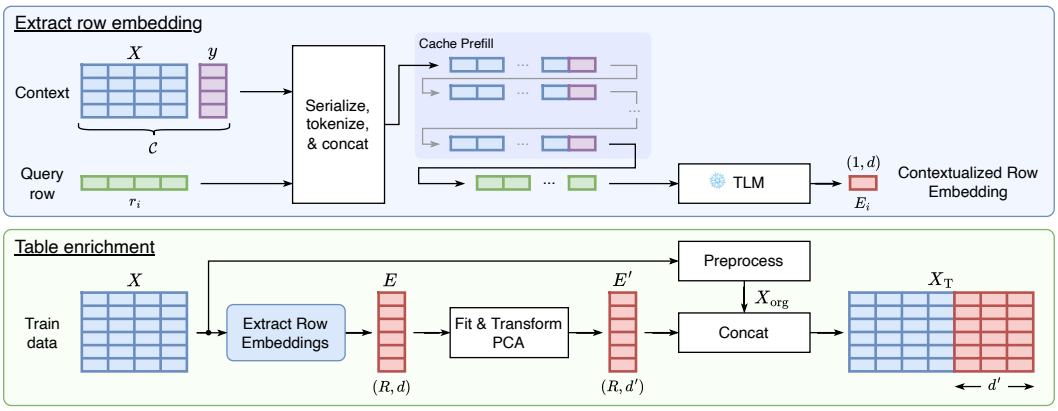


Figure 3: Overview of our proposed context-aware semantic embedding-based table enrichment pipeline. We pre-fill the KV cache of a table-native pretrained LLM using a context subsample of the training data. Rows are embedded using the contextualized [BOT] token of the sequence. For enrichment of the feature matrix X, individual rows are embedded, PCA-transformed and appended to the (potentially preprocessed) feature matrix.

Training Objective: We initialize our model from the pretrained Gemma 3 checkpoint (without instruction-tuning), which supports a 128k-token context window, and continue pretraining the model on a tabular imputation objective. Given an input table with feature-target split, our approach is trained using a supervised objective where the next-token prediction loss is computed exclusively over target tokens. Let $\mathcal{I}_{\text{target}}$ be the set of indices in $\mathbf{S}$ corresponding to the tokens of the target values $\{y_1, \ldots, y_N\}$. The training objective minimizes the target-masked negative log-likelihood,

$$\mathcal{L}(\theta) = -\sum_{t \in \mathcal{I}_{\text{target}}} \log P(s_t \mid s_{<t}; \theta)\,, \tag{4}$$

with respect to the model weights θ. This objective forces the model to predict target tokens from the [BOT] embedding conditioned on its preceding context. Note that, due to the autoregressive nature of the used LLM 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.

Training Data: We train our approach on the large-scale real-world T4 table corpus [9]. Similar to previous works [28, 22, 21], we generate tabular prediction tasks from these unlabeled datasets: 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.

Training Setting: We train different base model sizes (270M, 1B, 4B, and 12B parameters) for 8500 steps with a batch size of 128, corresponding to roughly 280 million rows in total. We uses Adam with a learning rate of 10^{-6}. We do not apply dropout or weight decay. To balance throughput and context utilization, we cap inputs at 16k tokens during training and truncate longer sequences.

3.2 Table Enrichment via Context-Aware Semantic Embeddings

Given a table corresponding to a predictive task with features X and target y, our goal is to perform semantic enrichment of X as part of a tabular data featurization pipeline usable with any tabular predictor. To this end, we leverage the contextualized row-wise embeddings obtained from our pretrained TLM. An overview of the proposed approach is given in Figure 3.

Context Priming and Contextualized Encoding: As opposed to creating isolated, row-wise embeddings for each row of the table, as investigated in previous works [12, 20], our goal is to establish a global semantic context for the row embeddings, grounding each embedding within the table's context and predictive task. To this end we employ a context priming strategy:

First, a random subset of rows is sampled as context $\mathcal{C}$, containing both features X and targets y from the task's train split, serialized and passed through the previously trained TLM. Second, the

Table 1: Evaluation results on the investigated benchmarks, reporting rank (Rk), accuracy (Acc) and (soft-clipped) R^2 in percent for classification and regression.

	All			CARTE			TextTab			TabArena		
Model	**Rk**	**Acc**	**R^2**	**Rk**	**Acc**	**R^2**	**Rk**	**Acc**	**R^2**	**Rk**	**Acc**	**R^2**
TabICLv2 w/ CASE	**2.7**	**86.5**	**76.1**	**1.3**	**81.0**	**77.5**	**2.9**	**85.7**	68.2	4.0	88.3	78.5
AutoGluon	3.1	85.6	73.9	2.6	78.8	73.7	3.7	83.5	67.5	3.4	88.1	79.9
ConTextTab	3.8	85.1	71.2	3.2	77.1	72.4	3.9	84.4	58.8	4.4	87.6	77.8
RealMLP	4.0	84.8	70.8	4.5	74.6	68.6	4.0	82.0	**68.3**	3.6	88.4	79.7
CatBoost	4.2	85.2	69.9	4.7	76.3	68.3	3.6	83.7	65.4	3.8	88.2	79.0
TabPFN-2.6	4.3	84.2	64.8	6.7	70.7	59.9	4.6	81.5	63.8	**1.8**	**88.7**	**80.5**
TabICLv2	4.5	84.3	61.1	6.9	70.5	55.3	4.5	82.5	60.2	2.1	**88.7**	79.9
XGBoost	5.3	84.2	69.1	5.7	73.5	66.9	5.1	81.4	65.4	5.1	87.9	78.8
Naive	8.5	70.1	-3.5	9.0	53.0	-1.7	8.6	70.4	-5.5	8.1	75.0	-7.3

resulting KV cache is retained as a permanent prefix, distilling the table's semantic distribution for all subsequent row encodings. Finally, each target row r_i is passed through the model, reusing the pre-filled KV cache. We extract the row embedding $\mathbf{e}_i$ from the final hidden layer at the final `[BOT]` token position. This embedding captures the integrated semantic summary of the record before the prediction would be made by the LLM, conditioning the embedding both on the global semantic context as well as the underlying predictive task. This is crucially different to existing approaches, which perform either cell- or row-wise, uncontextualized and target-agnostic embeddings.

Table Enrichment: In order to enrich the input feature matrix X with the obtained embeddings, the goal is to append the row embeddings along the feature axis as additional numerical features. However, the obtained row embeddings are high-dimensional, with embedding dimension up to $d = 4096$ for a Gemma 3 12B-based TLM. To reduce the dimensionality, preventing memory or performance issues when appended to the feature matrix for the tabular learners, we apply a Principal Component Analysis (PCA) to the set of embeddings $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$, reducing them to d' components. The final enriched feature matrix is: $X_{\text{T}} = [X_{\text{org}} \parallel \text{PCA}(\mathbf{E})]$, where X_{org} are the original, potentially preprocessed features, depending on the tabular learner used. To prevent data leakage, PCA components are fitted solely on the training distribution. These frozen components are subsequently applied to test-row embeddings during the inference phase.

4 Experiments and Results

Evaluation: We evaluate our approach across three distinct benchmark suites: CARTE [19], TextTab [23], and the single-fold variant of TabArena [8]. Our focus lies on CARTE and TextTab as they are specifically designed to emphasize semantic relationships in tabular data, aligning with the core motivation of our work. Conversely, TabArena serves as a numerics-heavy baseline, included to test the robustness of our approach in environments where statistical signals dominate over semantic ones. All benchmarks cover both classification and regression tasks. Throughout, we report mean accuracy for classification and mean (soft-clipped) R^2 for regression tasks, respectively, following previous works [28].

Baselines: We compare against a diverse set of state-of-the-art baselines, covering conventional per-dataset trained and HPO-tuned ones (XGBoost, CatBoost, and RealMLP), tabular in-context learners (TabPFN-2.6, TabICLv2, and ConTextTab), and the AutoGluon framework, using the "best" preset with a 4 h time limit [7]. In particular, note that we combine the table-tuned baselines (GBDTs and RealMLP) with the AutoGluon feature preprocessor which includes some text feature handling via n-gram features for free text or high-cardinal features. We perform ablations with other forms of text featurization in the subsequent sections. Additional details on the used baselines are provided in Appendix A.

Default setup: For our primary evaluation of CASE, we utilize the custom Gemma 3 12B-based TLM with a context priming window of $k = 128$ rows (with a maximum of 32k tokens) and a PCA-reduced latent space of $d' = 32$ for the feature fusion.

4.1 Tabular Learners benefit from semantic enrichment via CASE

We evaluate the efficacy of the CASE framework by enriching the input features for five prominent tabular learning architectures, spanning conventional as well as recent deep learning baselines, both per-table tuned as well as in-context learning approaches. Namely, we evaluate CatBoost, RealMLP, TabPFN-2.6, TabICLv2, and ConTextTab, representing best-in-class models in each category.

The results on the semantics-heavy CARTE and TextTab benchmarks, covering a total of 20 classification and 51 regression tasks, are shown in Figure 1.

Quantitative Gains: We observe that CASE yields a large accuracy uplift ranging from **2.9% to 6.80%** in classification tasks, and a remarkable R^2 gain between **3.8% and 19.1%** in regression tasks. These results suggest that our novel approach towards semantic embeddings successfully capture latent dependencies that are inaccessible to all investigated models, regardless of their architecture.

Synergy with Semantic Models: Notably, even ConTextTab – which natively incorporates cell-wise text embeddings using a pretrained Sentence Transformer – benefits significantly from CASE. This indicates that our target-aware, contextualized embeddings provide a more nuanced representation of tabular relationships that complement general-purpose sentence embedding used in current models.

Revitalizing ICL Models: A key finding is that CASE successfully boost performance of recent tabular ICL models without native text handling like TabICLv2 and TabPFN-2.6. While these models excel at numerically dominated benchmarks, as we will observe later, they exhibit inconsistent performance out-of-the-box on semantically rich ones. The addition of CASE-enriched features establishes a new performance ceiling. By bridging the "semantic gap" for these models, our approach successfully combines the predictive power of tabular ICLs with semantic reasoning of LLMs.

4.2 In-Depth Analysis of CASE

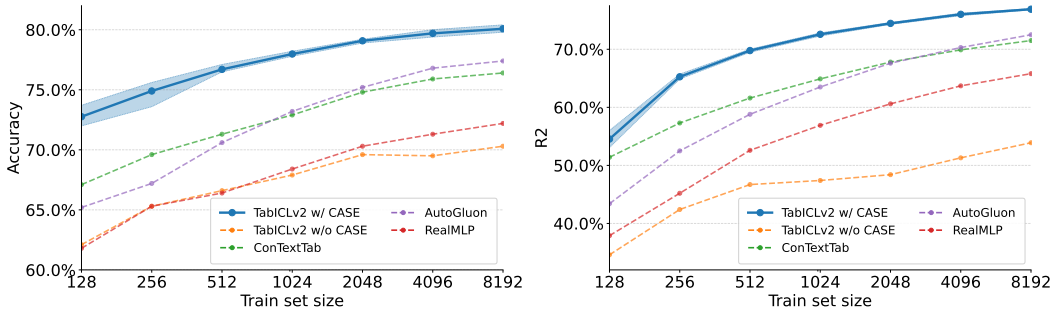
Next, we study the overall performance of CASE on CARTE, TextTab, but also TabArena, and evaluate the synergy of CASE in combination with TabICLv2 [26] as the overall best-performing model from our previous evaluation. For preprocessing, we adhere to TabICL's recommended configurations of Skrub's `TableVectorizer` to ensure a fair comparison with established statistical pipelines. The main results are summarized in Table 1. We provide additional details and investigate the statistical significance of our findings in Appendix B.

Performance on Semantic Benchmarks: As previously indicated, on the semantically rich CARTE and TextTab benchmarks, our approach largely outperforms all existing baselines. On CARTE, we observe a leading gap of over 2 pp in accuracy and almost 4 pp in R^2 over AutoGluon, which ensembles and stacks a multitude of model architectures and involves significant training and tuning for each dataset. Similarly, on TextTab, TabICLv2 with CASE has a more than 2 pp lead in terms of accuracy while being competitive with SOTA on regression tasks. This substantial margin of improvement over TabICLv2, AutoGluon, and ConTexTab suggests that, for datasets where column headers and textual features carry deep external meaning, the infusion of LLM-based world knowledge combined with a table-native target-aware training of the TLM is more effective than pure statistical feature engineering or isolated LLM embeddings.

Boundaries of Semantic Priming: While showcasing significant improvements on semantically rich benchmarks, we observe a small performance trade-off on the TabArena benchmark, where the raw statistical learners (TabICLv2 and AutoGluon) maintain a lead. This is expected, as TabArena is a numerics-heavy suite where the primary predictive signals are numerical rather than semantic. While the drop in performance of TabICLv2 with CASE over the default TabICLv2 is small, it is statistically significant, as we investigate in more detail in Appendix B. However, this also highlights the specialized nature of CASE: it acts as a modular semantic enhancer that can be used selectively.

Sample Efficiency and Few-Shot Performance: To investigate the effectiveness of CASE in the few-shot and low-data regime, we provide a comparison using subsampled training splits of the CARTE benchmark. That is, we subsampled the CARTE training splits ranging from 128 to 8k rows while keeping the test split untouched. To rigorously account for the inherent variance of random context selection, each evaluation is repeated over 5 independent runs using different random seeds.

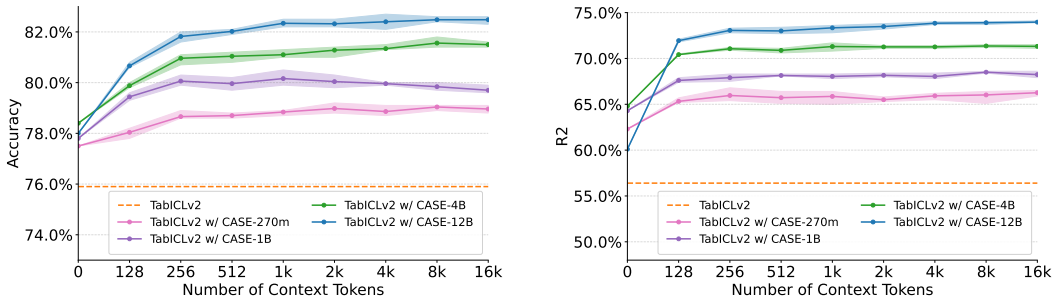
Our results, shown in Figure 4, indicate that in low-data regimes, CASE is highly dominant, outperforming state-of-the-art baselines by a large margin. Furthermore, the performance loss for smaller



(a) Average accuracy across classification tasks.

(b) Average R^2 across regression tasks.

Figure 4: Impact of Context-Aware Semantic Embeddings (CASE) on predictive performance of varying subsamples of the train splits from the CARTE benchmark using a fixed test set. Results for CASE correspond to mean (solid) and min/max (shaded) over five different seeds for context priming.



(a) Average accuracy across classification tasks.

(b) Average R^2 across regression tasks.

Figure 5: Performance across CARTE and TextTab for TabICLv2 enriched with CASE embeddings across varying parameter counts and context window utilization. Results correspond to mean (solid) and min/max (shaded) over five different seeds for context priming.

train sizes seems to be less strong as for conventional predictors. This suggests that the semantic priors embedded in the KV-primed representations provide a critical advantage when statistical signals are sparse, likely benefiting from the pretrained LLM's world knowledge. Furthermore, we observe a powerful synergy between the constituent models: the TLM generates rich, high-dimensional latent representations that can be effectively leveraged by the tabular learning model.

4.3 Ablations

We conduct a series of ablation experiments on the CARTE and TextTab benchmarks to validate our core hypotheses regarding the model architecture, parameter and context scaling, as well as feature representations. Throughout, we use TabICLv2 as the final predictor, as previously discussed.

To understand the scaling laws governing CASE, we investigate how both parameter count and context length influence the quality of the resulting embeddings. In Figure 5, we evaluate variants across four model scales – 270M, 1B, 4B, and 12B parameters – and nine context depths, ranging from 128 to 16k tokens (including a zero-context baseline). Again, we account for the variance of random context selection by evaluating over 5 independent runs using different random seeds.

Model Scaling: Our results reveal that all CASE configurations – regardless of base model scale or utilized context – consistently outperform the vanilla TabICLv2 baseline. We observe a robust, monotonic relationship between parameter count of the TLM base model and predictive performance: as the model size increases from 270M to 12B, the quality of the semantic embeddings improves across both classification and regression tasks as reflected by the increased performance. This aligns with established scaling laws for LLMs, suggesting that larger models possess a richer latent "world model" that allows for more precise mapping of tabular values to their underlying concepts.

Context Scaling: Across all base TLM variants, we observe a significant leap in performance when transitioning from 0 context tokens (corresponding to uncontextualized row embeddings) to 1k tokens. This "contextual leap" validates our hypothesis that a minimal semantic anchor is necessary to resolve the inherent ambiguity of isolated rows or cell values. Interestingly, even uncontextualized embeddings (0 context tokens) provide a substantial boost over the TabICLv2 baseline, indicating that the zero-shot semantic recognition of individual features is, on its own, already valuable.

Robustness to Random Selection: Crucially, each of our 5 independent evaluation runs draws a completely distinct subset of context rows with entirely randomized permutations in row ordering. Despite this shuffling, the resulting min/max variance bands visualized in Figure 5 are notably tight. The marginal fluctuations in downstream performance demonstrate that CASE is remarkably stable and resilient to the stochasticity of context row selection. This indicates that the model successfully distills the global data distributions and semantic structure from the anchor data regardless of the specific layout or samples encountered during inference.

Saturation and Complementary Reasoning: While performance continues to slightly improve as context scales up to 32k tokens, we observe a diminishing marginal return beyond the initial 1k boost. This is somewhat unexpected. As larger models should be able to leverage much larger context windows, one would have expected this saturation point to shift towards larger number of context tokens for larger base model sizes. Maybe, this saturation suggests a synergy between the embedding model and the tabular learner: Our tuned TLM acts as a high-level "reasoner" that requires only a small representative sample of rows to ground the tables' semantics. Once the model has resolved the individual row's context, the statistical heavy lifting is effectively offloaded to the downstream tabular learner, which processes the full training set. This synergy allows CASE to remain computationally efficient, as it only requires a comparably small context KV cache. However, these findings may also suggest further room for improvements, e.g. in pretraining, to more effectively leverage the larger context windows of the larger TLM base models.

Standalone Generative Prediction and Architectural Analysis: While CASE is primarily designed to augment downstream tabular learners, the generative pre-training of the underlying TLM makes the model also usable as a standalone few-shot predictor. In Figure 6, we evaluate this capability by comparing the vanilla TabICLv2, the CASE-enriched TabICLv2, and the standalone CASE-TLM (decoder-only 12B) on CARTE classification tasks.

As a standalone model, the decoder-only TLM demonstrates strong predictive performance that scales monotonically with the number of context rows. Notably, given only 8 context rows, our TLM outperforms TabICLv2 (given the full train split) across the CARTE classification tasks. However, note that CARTE contains exclusively binary classification targets. This confirms that the model has successfully learned to predict target tokens from its context. However, the TLM stays behind TabICLv2 with CASE, confirming the complementary nature of the two approaches. Note that standalone regression was not evaluated, as standard decoder-only models do not natively support continuous value regression without specialized head fine-tuning.

Moreover, we compare different LLM architectures: In the literature, encoder-decoder (e.g., T5) or encoder-only (e.g., BERT) architectures are more conventional choices for embedding generation. To validate our choice of a decoder-only backbone, we trained encoder-decoder variants using the *T5Gemma2* architecture [30] under an identical training budget and training objective. As illustrated in Figure 6, while the encoder-decoder variants also scale with context, they consistently and largely under perform relative to the decoder-only models.

Impact of Contextualization: To isolate the benefit of the proposed KV-cache priming, we compare CASE ($k = 128$) against a non-contextualized row-embedding baseline (`TabICL[Row-Embedding]`, $k = 0$) as well as cell-level embeddings (`TabICL [Cell-Embedding]`) obtained via a standard Sentence Transformer, `all-MiniLM-L6-v2`, similar to related works [28]. We compare this against the default TabICLv2 model as well as combining TabICLv2 with Skrub's TF-IDF-based featurizer. The results are depicted in Table 2.

The significantly degraded performance of the non-contextualized variant as compared to the full CASE approach confirms that row-level semantics are inherently distributional. Without the provided semantic context, the model fails to resolve the semantic ambiguities potentially present in an isolated row. Similarly, using cell-level embeddings shows marginal improvements over TF-IDF-based embeddings, but consistently lags behind our contextualized row embedding approach.

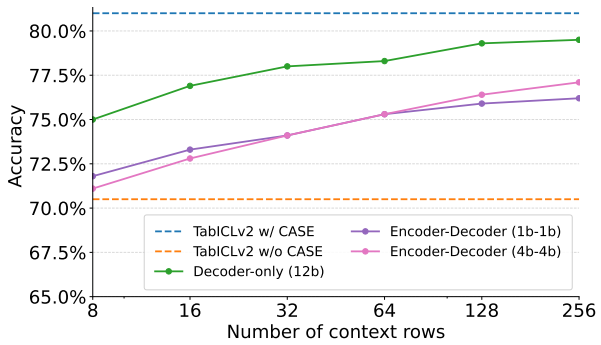


Figure 6: Architectural Comparison on CARTE classification tasks, comparing the predictive capabilities of table-tuned LLMs with different architectures.

Table 2: Feature encoding ablations results on the CARTE and TextTab benchmarks, depicting mean accuracy for classification and R^2 for regression tasks, as well as mean rank across all tasks.

Model	Rank	Acc	R^2
TabICL [TF-IDF + CASE]	**1.97**	**83.1**	**75.5**
TabICL [Default + CASE]	3.38	82.5	73.9
TabICL [Row-Embedding]	3.52	81.7	73.7
TabICL [Cell-Embedding]	4.28	81.2	72.0
TabICL [CASE w/o X_{org}]	4.41	82.1	71.7
TabICL [Gemma3]	4.48	81.8	72.5
TabICL [TF-IDF]	4.76	80.9	69.7
TabICL [Default]	7.06	75.9	56.4

Importance of Tabular-Specific LLMs: Further, we replaced the specialized table-tuned TLM backbone with an off-the-shelf Gemma-3-12B model [29], validating our pretraining objective and table-native tuning. The results are again depicted in Table 2. While again showing slight improvements as compared to conventional TF-IDF features, the significant performance drop compared to CASE underscores that generic LLMs, while possessing vast world knowledge, require table-native adaptations to correctly interpret the structural and relational nuances of tabular data.

Role of Statistical Preprocessing: Finally, we evaluate the synergy between traditional preprocessing and semantic enhancement by comparing TabICLv2 with and without TF-IDF features using Skrub's `TableVectorizer`. Our results demonstrate that while classical statistical techniques remain a robust baseline, the addition of CASE consistently yields performance gains regardless of the preprocessing used. To isolate the contribution of our embeddings, we further evaluate a variant using only CASE features (`TabICL [CASE w/o` X_{org}`]`). We find that standalone embeddings serve as high-utility features; however, the inclusion of the original X_{org} features yields the strongest results.

5 Conclusions

Limitations and Future Work: While CASE demonstrate significant potential, our current implementation has some limitations to be addressed in future works. First, as our framework is built upon a table-tuned Gemma model, it inherits constraints applicable to all table-serialization based approaches combined with LLMs: a lack of permutation equivariance with respect to both row and column ordering (which we, however, find empirically not to be statistically significant in Appendix C.2), and the limited numerical inductive bias inherent in token-based language models. Also, our current context priming strategy relies on uniform random sampling to populate the KV cache, which is inherently stochastic and may omit critical anchor rows that define the table's distribution. However, we empirically find the random context sampling not to be statistically significant (see Appendix C.3). Nevertheless, this motivates further research into developing LLM architectures specifically optimized for tabular structures, for example adapting recent approaches for permutation-invariant attention patterns and positional encodings [6]. Furthermore, using PCA to project the high-dimensional embeddings into a compact latent space may be suboptimal. Future works could investigate non-linear manifold learning techniques which may better preserve the subtle semantic variances in the data. While the use of a language model increases the computational footprint during feature extraction, our KV-cache priming strategy ensures that this overhead remains significantly lower than traditional AutoML pipelines or exhaustive hyperparameter searches (see Appendix D). Finally, while utilizing open-source benchmarks carries a theoretical risk of data contamination from the LLM backbone or continued pretraining, our empirical analyses suggest that performance gains are driven by genuine semantic reasoning rather than memorization (see Appendix C.1). We discuss potential data contamination in detail in Appendix E and further validate our findings in Appendix F on the recently introduced STRABLE benchmark [2], which corroborates the strong performance improvements observed on CARTE and TextTab.

Summary: In this work, we introduced CASE, a framework designed to bridge the historical divide between the vast semantic world knowledge of LLMs and the rigorous statistical inductive biases of specialized tabular learners. At the heart of our contribution is a pretrained, table-tuned LLM which functions as a few-shot semantic engine for structured data. By employing a novel KV-cache priming strategy, CASE enables the generation of target-aware row representations that are no longer processed in a semantic vacuum, but are instead anchored within the specific statistical and domain distribution of the parent table. Our empirical evaluation across the CARTE and TextTab benchmarks demonstrates that CASE substantially advances performance baselines for semantically rich tabular data. The framework is particularly potent in the low-data regime, where the LLM's learned priors compensate for the lack of sufficient statistical signals. By demonstrating that decoder-only architectures can serve as efficient, context-aware embedding engines, we provide a scalable blueprint for building foundation models for tabular data.

Acknowledgements

We would like to thank Johannes Hoffart, Markus Kohler, and Sam Thelin for their insightful comments and suggestions throughout the development of this work. We thank Myung Jun Kim for providing valuable feedback on experimental setups and drafts of this contribution. We thank Michael Medek for implementation support in initial versions of our Tabular Language Model.

References

[1] Alan Arazi, Eilam Shapira, and Roi Reichart. TabSTAR: A tabular foundation model for tabular data with text fields. *Advances in Neural Information Processing Systems*, 38, 2026.

[2] Gioia Blayer, Myung Jun Kim, Félix Lefebvre, Lennart Purucker, Alan Arazi, Eilam Shapira, Roi Reichart, Frank Hutter, Marine Le Morvan, David Holzmüller, and Gaël Varoquaux. STRABLE: Benchmarking tabular machine learning with strings. *arXiv preprint arXiv:2605.12292*, 2026.

[3] 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, 2016.

[4] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot Arena: An open platform for evaluating LLMs by human preference. *arXiv preprint arXiv:2403.04132*, 2024.

[5] 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, 2022.

[6] Beni Egressy and Jan Stühmer. Set-LLM: A permutation-invariant LLM. *Advances in Neural Information Processing Systems*, 38:62798–62834, 2026.

[7] 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.

[8] Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, David Salinas, and Frank Hutter. TabArena: A living benchmark for machine learning on tabular data. *Advances in Neural Information Processing Systems*, 38, 2026.

[9] Joshua P Gardner, Juan Carlos Perdomo, and Ludwig Schmidt. Large scale transfer learning for tabular data via language modeling. *Advances in Neural Information Processing Systems*, 37:45155–45205, 2024.

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

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

[12] 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.

[13] 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*, 2023.

[14] Steffen Herbold. Autorank: A Python package for automated ranking of classifiers. *Journal of Open Source Software*, 5(48), 2020.

[15] 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.

[16] 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), 2025.

[17] 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, 2024.

[18] 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.

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

[20] Félix Lefebvre, Myung Jun Kim, and Gaël Varoquaux. Knowledge-rich embeddings for tabular learning. In *EurIPS 2025 Workshop: AI for Tabular Data*, 2025.

[21] Junwei Ma, Nour Shaheen, Alex Labach, Amine Mhedhbi, Frank Hutter, Anthony L Caterini, and Valentin Thomas. Generalization can emerge in tabular foundation models from a single table. In *EurIPS Workshop on AI for Tabular Data*, 2025.

[22] 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. *Advances in Neural Information Processing Systems*, 2025.

[23] 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.

[24] 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.

[25] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

[26] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

[27] Günther Schindler, Maximilian Schambach, Michael Medek, and Sam Thelin. TabGemma: Text-based tabular ICL via LLM using continued pretraining and retrieval. In *EurIPS 2025 Workshop: AI for Tabular Data*, 2025.

[28] Marco Spinaci, Marek Polewczyk, Maximilian Schambach, and Sam Thelin. ConTextTab: A semantics-aware tabular in-context learner. *Advances in Neural Information Processing Systems*, 38, 2026.

[29] 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.

[30] Biao Zhang, Paul Suganthan, Gaël Liu, Ilya Philippov, Sahil Dua, Ben Hora, Kat Black, Gus Martins, Omar Sanseviero, Shreya Pathak, Cassidy Hardin, Francesco Visin, Jiageng Zhang, Kathleen Kenealy, Qin Yin, Xiaodan Song, Olivier Lacombe, Armand Joulin, Tris Warkentin, and Adam Roberts. T5gemma 2: Seeing, reading, and understanding longer. *arXiv preprint arXiv:2512.14856*, 2025.

[31] Xiyuan Zhang, Danielle C Maddix, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael W Mahoney, et al. Mitra: Mixed synthetic priors for enhancing tabular foundation models. In *Advances in Neural Information Processing Systems*, 2025.

[32] 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 Baseline Details

TabICLv2: We use the model from the official `tabicl` package at version 2.0.3 using its default feature preprocessing if not marked otherwise specifically.

TabPFN: We use the model from the official `tabpfn` package at version 7.1.1, corresponding to the TabPFN-2.6 checkpoint, using its default feature preprocessing if not marked otherwise specifically.

AutoGluon: We evaluate AutoGluon v1.5 with its native feature encoder. We use the `best_quality` preset with a per-dataset time limit of 4 h running on a 32 cores machine with 256 GB of RAM.

ConTextTab: We evaluate ConTextTab v1.1.2 using the reference implementation and checkpoint[1]. We set a context size of 8k samples and evaluate with 8-fold bagging.

Pytabkit models: We use the `pytabkit` [17] for evaluating XGBoost, CatBoost, and RealMLP. We evaluate these models with ensembled hyperparameter optimization across 5-fold inner cross-validation (HPO-CV). For the HPO variants, we use the recently added `tabarena` search spaces proposed in [8].

Naive: We use the Naive predictor from sklearn v1.5.2, estimating the median for regression tasks or the majority class in the case of classification tasks.

A.1 Feature preprocessing

If available, we use the default feature preprocessing of the individual baselines investigated. For Naive, XGBoost, CatBoost, and RealMLP, we use the preprocessor from AutoGluon. For TF-IDF features, we use Skrub's `TableVectorizer`, which uses a pass-through for low-cardinal and numerical features to be handled natively by TabICL, and the TF-IDF-based `StringEncoder` for high-cardinal features with more than 40 classes, using a fixed seed. Note that the TableVectorizer also applies PCA to the obtained features, downcasting them to 30 dimensions. Finally, note that these features are calculate per column. That is, if the table contains 10 high-cardinal or free text features, the approach appends 300 additional features after the transform.

B Additional Results

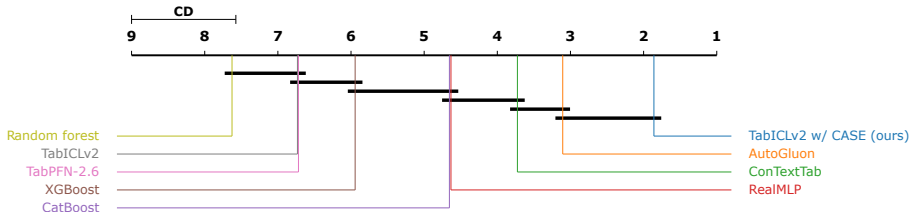
In the following, we provide additional results on both the semantically rich benchmarks (CARTE and TextTab) as well as the numerics-heavy TabArena. We report critical difference (CD) diagrams, as well as win-ratios and ELO scores.

For the calculation of the Elo scores, we utilize the implementation of TabArena [8], which itself is based on ChatBot Arena's implementation [4].

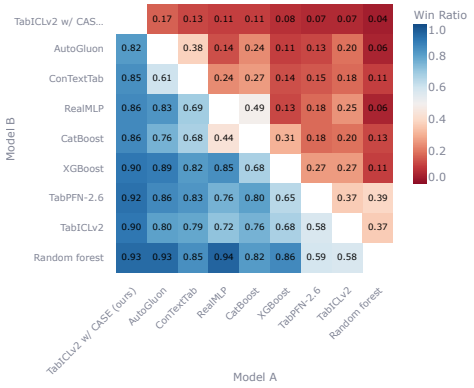
For all ranking results (aside from those used in the critical difference diagrams), we use a robust ranking algorithm. That is, models that lie within 0.1 percentage points (in terms of either accuracy or R^2) are considered ties. The same robustification also applies to our Elo score and win-ratio calculations.

For the critical difference (CD) diagrams, we use the `autorank` library [14]. So as not to influence the statistical guarantees of the significance test used, we do not alter or robustify its internal rank calculation. Hence, the ranks shown in the CD diagrams can slightly deviate from those shown in the result tables or figures.

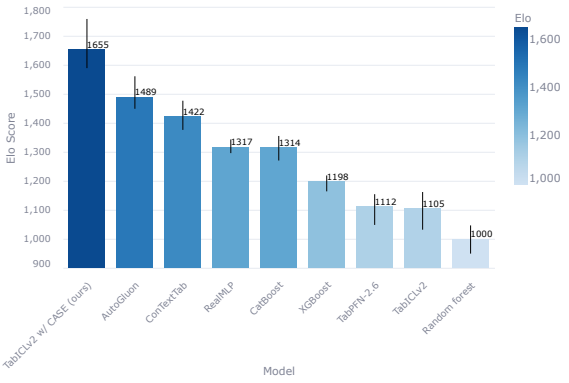
[1] github.com/SAP-samples/contexttab



(a) Critical difference diagram.



(b) Win ratios.



(c) ELO scores.

Figure 7: Critical difference diagrams, win ratios and ELO scores for the semantic-heavy CARTE and TextTab benchmarks. ELO scores are normalized to Random Forest at 1000 ELO.

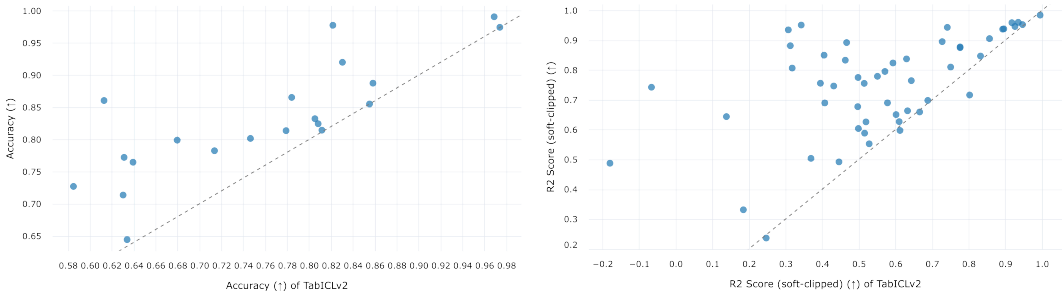
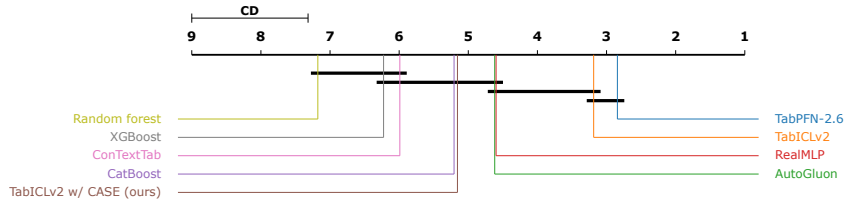


Figure 8: Per-task accuracy (left) and R^2 scores (right) on CARTE and TextTab comparing TabICLv2 with (y-axis) and without CASE (x-axis). On-diagonal scores indicate identical performance while above-diagonal scores indicate that performance is better when using CASE instead of the default.

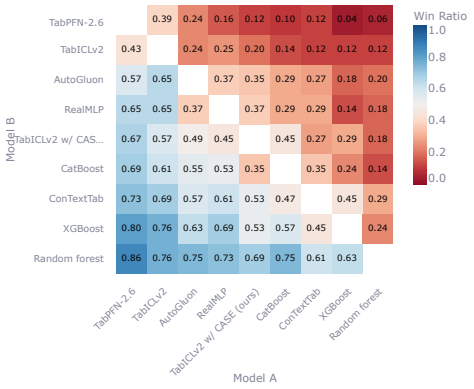
B.1 Semantically Rich Benchmarks

For the CARTE and TextTab benchmarks, the CD diagram, win ratios and ELO score plots are depicted in Figure 7. As before, we observe that TabICL with our proposed CASE enrichment significantly outperforms all baselines. For all single-model baselines, this difference is statistically significant. While our approach also outperforms AutoGluon, as previously denoted in terms of absolute performance and mean rank, now also shown in terms of win ratio and Elo scores, the difference is not quite statistically significant, as denoted in the CD diagram. However, note that AutoGluon ensembles and stacks a multitude of models, requiring extensive tuning for each dataset, while TabICL with CASE performs on-the-fly predictions following the in-context paradigm. Finally, a one-vs-one comparison of per-task results using TabICLv2 with and without CASE, shown in Figure 8, show that the gains in absolute performance are substantial in many cases.

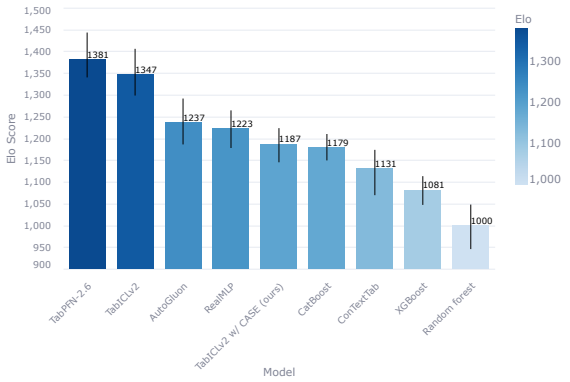
Overall, also these extended results underline the superior performance of our approach, setting a new state of the art for semantically rich datasets.



(a) Critical difference diagram.



(b) Win ratios.



(c) ELO scores.

Figure 9: Critical difference diagrams, win ratios and ELO scores for the numerics-heavy TabArena benchmarks. ELO scores are normalized to Random Forest at 1000 ELO.

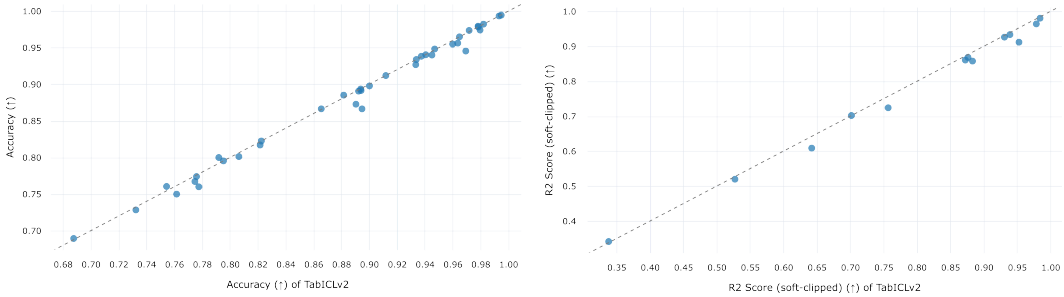
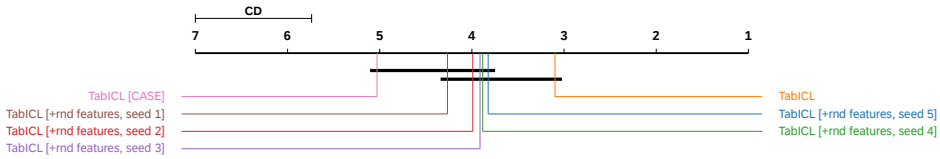


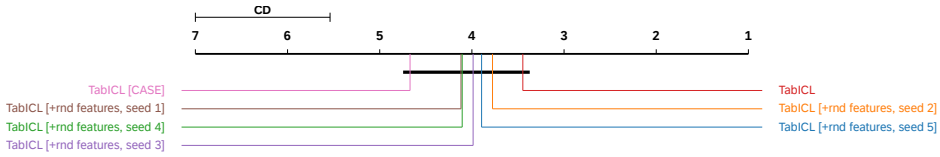
Figure 10: Per-task accuracy (left) and R^2 scores (right) on TabArena comparing TabICLv2 with (y-axis) and without CASE (x-axis). On-diagonal scores indicate identical performance while above-diagonal scores indicate that performance is better when using CASE instead of the default.

B.2 TabArena

As a robustness check, we also show extended results for the numerics-heavy TabArena benchmark in Figure 9. As previously discussed, we observe a performance drop when using CASE with numerics-heavy semantic-scarce dataset. This drop is statistically significant, as shown in the CD diagram. While we observe this drop throughout, in terms of mean rank as well as win ratios and Elo scores, these ranking metrics to some degree hide the differences in the absolute values of the performance metrics: while we have seen substantial gains on semantically rich datasets also in terms of absolute scores, the absolute *drop* in performance on TabArena is relatively small (albeit statistically significant), as shown in Figure 10. This hints at the problem of reporting or relying on single metrics for model comparisons, in particular ranking- or win-based ones: a tiny, but consistent, improvement may result in much better rank or Elo scores while hiding that the absolute improvement may be incremental.



(a) Across all tasks.



(b) Across classification tasks only.

Figure 11: Critical difference diagram comparing vanilla TabICL, TabICL with CASE and TabICL with additional random features across the TabArena benchmark.

Additional investigation of performance drop: Nevertheless, to investigate the potential mechanism underlying this drop in performance, we compare our approach with the performance of TabICL under addition of uninformative features. Doing so, we can investigate to what extent the drop in performance is due to a noisy or uninformative nature of CASE embeddings under purely-numeric datasets. To this end, we append 32 randomly sampled features to the context and queries, corresponding to the same amount of additional features when using CASE, and evaluate over 5 different random seeds. The results are depicted in Figure 11b. Indeed, we can observe that addition of random features does degrade TabICL's performance, albeit this degradation is not statistically significant. While overall TabICL with CASE is statistically significantly worse than bare TabICL across TabArena, as previously discussed, this does not stretch across all noisy feature ablations. Looking more closely, we can make out that the significant part of the performance degradation is driven by the regression tasks on TabArena, while performance remains statistically indistinguishable across the classification parts, where also absolute performance remains virtually identical, dropping from 88.7% to 88.3% accuracy, whereas on regression we see a drop from 79.9% to 78.5% R^2.

To summarize, uninformative, noisy feature do result in some degradation of TabICL's performance and explain part of the drop in performance when using TabICL with CASE on TabArena, in particular on regression tasks. However, some additional degradation can be observed with CASE embeddings on numerics-heavy datasets, in particular on regression tasks, that go beyond noisy embeddings. This warrants further research to robustify CASE embeddings on regression tasks and under the presence of dominantly numerical features. While these drops are statistically significant, they are small in terms of absolute scores.

Table 3: Semantic corruption performance degradation: Disruption of textual semantics drops TabICL [CASE] back to pure non-semantic baseline performance (TabICL), confirming that CASE gains its predictive power via genuine semantic reasoning.

	All	CARTE			TextTab		
Model	Rank	Rank	Acc (%)	R^2 (%)	Rank	Acc (%)	R^2 (%)
TabICL [CASE]	1.3	1.3	80.8	77.5	1.6	86.2	68.9
TabICL	2.3	2.2	78.3	73.9	2.5	84.1	54.5
TabICL [corrupted]	2.7	2.8	77.7	73.4	2.4	84.0	63.8
TabICL [CASE, corrupted]	3.0	3.2	78.2	72.2	2.5	84.2	63.6

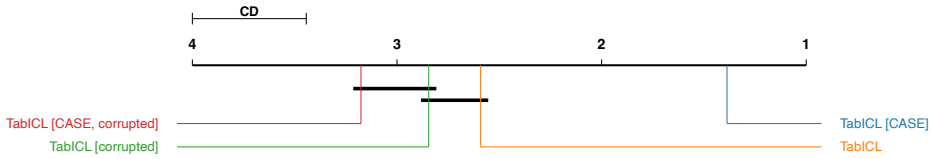


Figure 12: Critical difference diagram under semantic corruptions: Across CARTE and TextTab benchmarks ($\alpha = 0.05$), standard TabICL [CASE] achieves a significantly superior performance rank compared to all non-semantic or corrupted configurations.

C Additional Ablations

C.1 Importance of Semantic Signals

C.1.1 Impact of Semantic Corruption

To further evaluate whether CASE leverages true semantic understanding or merely exploits statistical patterns, we conduct a *semantic corruption test*.

Experimental Setup: We designed a deterministic character-substitution preprocessor that applies a consistent substitution cipher across all column headers and string/textual/categorical cell values. Numerical values and overall row/column structure remain identical. This completely strips away real-world domain semantics (e.g., column headers like `"Age"` or cell values like `"Doctor"`) while preserving identical categorical frequencies. We evaluate two baseline pipelines on CARTE and TextTab benchmarks under both original and corrupted regimes:

1. **TabICL**: Pure non-semantic baseline utilizing TF-IDF encoding.
2. **TabICL [CASE]**: Combined pipeline integrating CASE-12B semantic embeddings with TF-IDF.

Statistical Impact and Analysis: Results are depicted in Table 3. Three key insights emerge regarding when and why CASE provides performance gains:

- **Performance Degradation upon Corruption:** When semantics are corrupted, the mean rank of TabICL [CASE, corrupted] drops sharply from 1.3 to 3.0, returning to a level statistically indistinguishable from the baseline TabICL model (2.3).
- **Statistical Superiority of Uncorrupted CASE:** The CD-diagram in Fig. 12 confirms that the superiority of CASE leveraging semantic context over scrabled text is statistically significant, placing it in its own distinct rank group.
- **Robustness of Non-Semantic Baselines:** Standard TabICL exhibits virtually identical performance between uncorrupted (78.3% Acc, 73.9% R^2) and corrupted (77.7% Acc, 73.4% R^2) datasets on CARTE, as expected for TF-IDF representations.

Conclusion – when CASE Helps: These results confirm that CASE's performance gains are driven by genuine semantic content. When semantic information is present, CASE lifts performance substantially. When semantic information is absent or destroyed, CASE gracefully degrades to a standard non-semantic tabular modeling baseline performance.

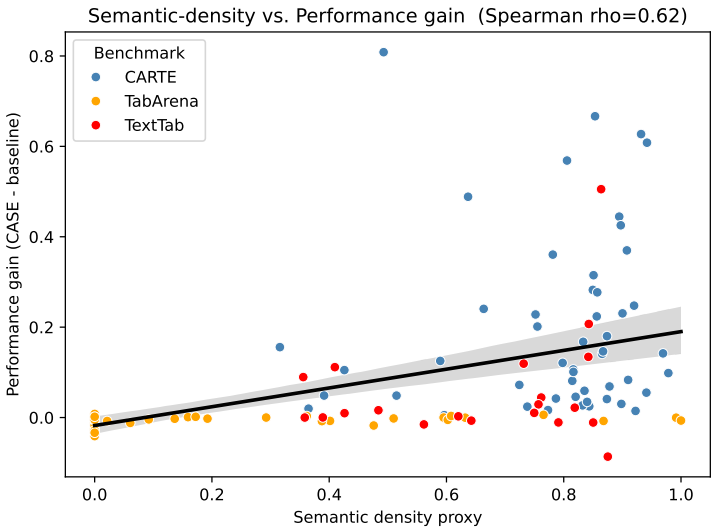


Figure 13: Performance gains of using CASE over a vanilla TabICL baseline depending on the semantic density proxy of the underlying task across the investigated benchmarks.

C.1.2 Performance Correlation with a Semantic Proxy

Finally, to investigate in detail how predictive performance depends on the semantic density of a dataset, we investigate performance improvements of TabICL with CASE over bare TabICL over a semantic density proxy across the TabArena as well as text-heavy benchmarks (CARTE and TextTab).

Semantic Density Proxy: To quantify *how much* of a dataset's predictive information is carried by genuine natural language – as opposed to numerical or categoricals – we define a scalar *semantic density* proxy $\rho \in [0, 1]$ for each table. Intuitively, ρ measures the share of the table's total column-wise information that resides in natural-language text columns. We first partition the (target-excluded) feature columns into string and numeric columns. For each string column c, we tokenize all cell values and compute two quantities: its token-frequency Shannon entropy $H_c = -\sum_t p_t \log_2 p_t$ (in bits), which captures the raw information content of the column, and a *textiness* weight $w_c \in [0, 1]$, the fraction of tokens that are alphabetic and of length ≥ 3. The latter is a lightweight, language-agnostic gate that down-weights columns dominated by identifiers, single-symbol codes, or short abbreviations, which are high-entropy yet carry no exploitable linguistic semantics. For each numeric column c we compute a binned Shannon entropy H_c over 32 equal-width histogram bins. The semantic density is then the textiness-weighted string entropy normalized by the total column entropy,

$$\rho \;=\; \frac{\sum_{c\in\mathcal{S}} w_c\, H_c}{\sum_{c\in\mathcal{S}} H_c \;+\; \sum_{c\in\mathcal{N}} H_c}, \tag{5}$$

where $\mathcal{S}$ and $\mathcal{N}$ denote the sets of string and numeric columns, respectively. By construction, numeric columns only enter the denominator and can therefore only *dilute* the score, while non-linguistic string columns are suppressed through w_c; both known confounds are thus excluded by design. A value of $\rho \to 1$ indicates a table whose information is almost entirely natural-language text, whereas $\rho \to 0$ corresponds to a purely numeric or code-based table. To bound computation on large tables, we subsample at most $5{,}000$ rows per dataset before computing the statistic.

The results are depicted in Figure 13. First, we observe that the proxy nicely distinguishes TabArena versus text-heavy benchmarks (CARTE and TextTab), which contain tasks with much higher semantic density. Next, we confirm that performance gains of using CASE vs vanilla TabICL strongly correlates with the semantic density (Spearman correlation of 62%) confirming that CASE successfully leverages semantic content in those predictive tasks.

Table 4: Robustness to column order permutation: Performance across 5 random column ordering seeds using CASE-12B with TabICLv2. Ranks and metrics show negligible variance.

	All	CARTE			TextTab		
Column Seed	Rank	Rank	Acc	R2	Rank	Acc	R2
Seed 1	2.3	2.4	81.3	77.4	2.2	86.1	68.9
Seed 2	2.5	2.7	80.8	77.5	2.0	86.2	68.9
Seed 3	2.7	2.6	81.1	77.5	3.0	85.8	67.8
Seed 4	2.5	2.5	81.0	77.6	2.6	85.9	67.8
Seed 5	2.6	2.6	80.7	77.5	2.6	85.9	66.6
Mean ± Std	$\mathbf{2.5 \pm 0.1}$	$\mathbf{2.6 \pm 0.1}$	$\mathbf{81.0 \pm 0.2}$	$\mathbf{77.5 \pm 0.1}$	$\mathbf{2.5 \pm 0.4}$	$\mathbf{86.0 \pm 0.2}$	$\mathbf{68.0 \pm 1.0}$

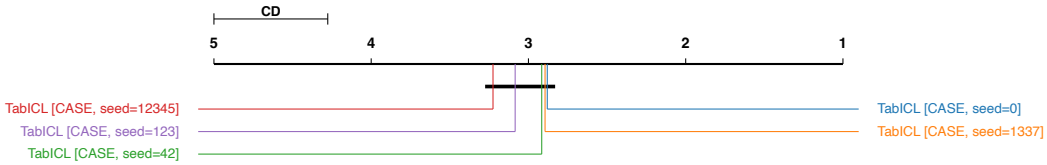


Figure 14: Critical difference diagrams for CARTE and TextTab benchmarks using different seeds for column permutations.

C.2 Impact of Column Order Permutation

CASE is neither row nor column permutation equivariant. While we have previously investigated the robustness with respect to context sampling and row order, we now want to evaluate whether the order of input columns impacts the downstream embedding quality. To this end. we conduct an ablation study over 5 distinct column ordering seeds using the CASE-12B model.

Statistical Equivalence via CD Diagrams: We performed a Friedman test followed by a post-hoc Nemenyi test ($\alpha = 0.05$) across CARTE and TextTab benchmarks to construct Critical Difference (CD) diagrams for the different column ordering seeds. In the resulting CD diagram (Fig 14), all 5 seed variants are connected by a single, continuous critical difference bar. While some fluctuations exist, this confirms that there is *no statistically significant difference* in overall performance ranks across any of the column permutations.

Performance Stability: As shown in Table 4, performance across both CARTE and TextTab benchmarks remains exceptionally tight across all seeds. For instance, the average classification accuracy on CARTE fluctuates within a minimal margin of $81.0\% \pm 0.2\%$, while the overall average rank varies by only ± 0.1.

These results demonstrate that while CASE is architecturally non-equivariant to column permutations, its learned semantic representations are robust with respect to column ordering.

Table 5: Robustness to random context selection: Performance across 5 random context sampling seeds using CASE-12B with the Skrub preprocessor on TabICL. Ranks and metrics show negligible variance.

	All	CARTE			TextTab		
Context Seed	Rank	Rank	Acc	R2	Rank	Acc	R2
Seed 1	2.7	2.8	80.8	77.4	2.4	85.9	68.3
Seed 2	2.3	2.5	80.8	77.5	1.9	86.2	68.9
Seed 3	2.5	2.4	80.7	77.6	3.0	85.6	67.9
Seed 4	2.9	2.8	80.5	77.3	3.0	85.9	67.7
Seed 5	2.4	2.4	80.9	77.6	2.2	85.9	68.6
Mean ± Std	$\mathbf{2.5 \pm 0.2}$	$\mathbf{2.6 \pm 0.2}$	$\mathbf{80.7 \pm 0.2}$	$\mathbf{77.5 \pm 0.1}$	$\mathbf{2.5 \pm 0.5}$	$\mathbf{85.9 \pm 0.2}$	$\mathbf{68.3 \pm 0.5}$

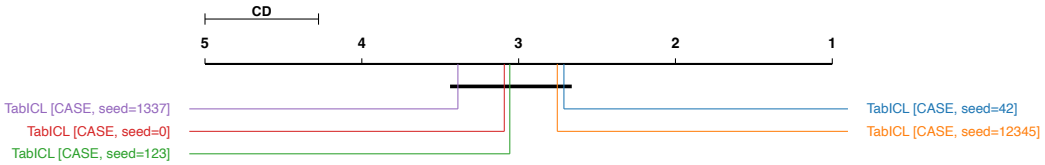


Figure 15: Critical difference diagrams for CARTE and TextTab benchmarks using different seeds for random context selection.

C.3 Impact of Random Context Selection

CASE relies on a randomly selected context to prefill the KV cache. To evaluate whether CASE exhibits instability due to stochastic context selection, we evaluated the CASE-12B model across 5 independent evaluation runs. Each run draws a completely distinct random sample of context rows with a randomized permutation of row order.

Statistical Equivalence via CD Diagrams: To test whether random context selection leads to statistically significant performance variations, we performed a Friedman test followed by a post-hoc Nemenyi test ($\alpha = 0.05$) across all benchmark datasets. The resulting Critical Difference (CD) diagram connects all 5 context selection seeds under a single continuous critical difference bar. This statistically confirms that varying the random seed for context row selection produces no significant difference in overall model performance.

Performance Stability: As detailed in Table 5, performance across all context selection seeds remains remarkably stable. The overall average rank stays tightly bounded around 2.5 ± 0.2, with average accuracy on CARTE showing a tight standard deviation of only 0.2% ($80.7\% \pm 0.2\%$). Similarly, regression performance (R^2) varies by a maximum of 0.3% across all runs on CARTE and TextTab.

These results indicate that CASE effectively distills the broader underlying semantic distribution of the tabular dataset rather than over-indexing on specific, high-leverage context examples or arbitrary row arrangements.

Table 6: Sensitivity to PCA embedding dimensions: Increasing PCA components monotonically improves representation expressiveness while $k = 32$ serves as a strong default.

	All	CARTE			TextTab		
Configuration	Rank	Rank	Acc (%)	R^2 (%)	Rank	Acc (%)	R^2 (%)
TabICL [CASE, $k = 256$]	1.9	1.6	81.3	77.8	2.4	86.3	69.0
TabICL [CASE, $k = 128$]	1.9	1.8	81.0	77.7	1.9	86.2	69.6
TabICL [CASE, $k = 64$]	2.2	2.1	80.7	77.7	2.5	86.1	69.6
TabICL [CASE, $k = 32$]	2.8	2.8	80.8	77.5	2.6	86.2	68.9
TabICL [CASE, $k = 16$]	3.4	3.4	80.7	77.2	3.4	85.8	68.2

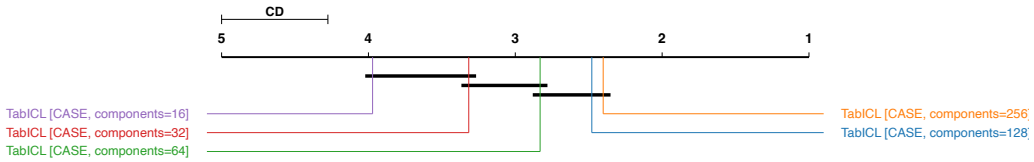


Figure 16: Critical difference diagram across PCA dimensions. Higher PCA dimensions ($k = 128, 256$) yield statistically significant rank improvements across CARTE and TextTab benchmarks ($\alpha = 0.05$).

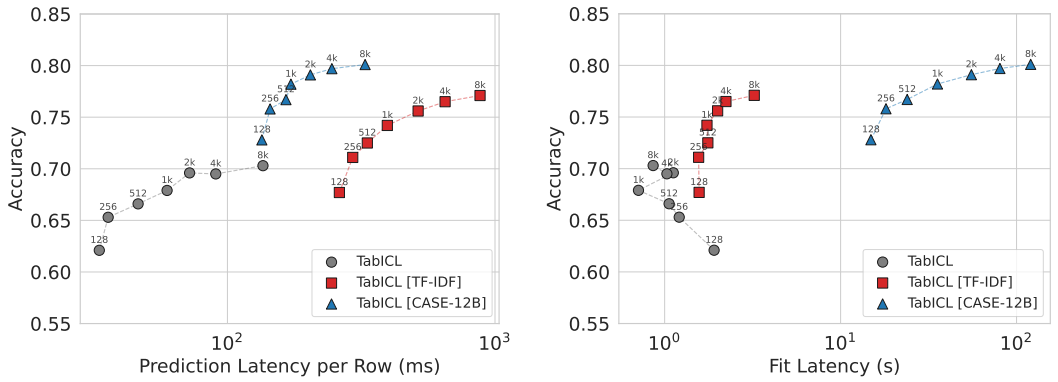
C.4 Sensitivity to PCA Dimension Reduction

To evaluate how embedding dimensionality affects downstream prediction quality, we conducted an ablation study over the number of Principal Component Analysis (PCA) components ($k \in \{16, 32, 64, 128, 256\}$) used to compress CASE representations.

Statistical Ranking via CD Diagrams: We evaluated ranking performance across CARTE and TextTab benchmarks using a Friedman test followed by a post-hoc Nemenyi test ($\alpha = 0.05$). As illustrated in the Critical Difference (CD) diagram (Fig. 16), increasing the target dimension yields progressive performance gains:

- **High-Dimensional Regimes ($k \in \{128, 256\}$):** Models using 128 and 256 components achieve the best overall ranks (1.9), significantly outperforming lower-dimensional settings ($k \in \{16, 32\}$).
- **Standard Default ($k = 32$):** Our default setting of 32 components forms a statistically equivalent clique with $k = 16$ and $k = 64$.

Performance Trade-offs & Analysis: As detailed in Table 6, performance scales predictably with dimensionality: Classification accuracy remains stable across settings (varying by only $\sim 0.6\%$ across $k = 16$ to 256), whereas regression metrics (R^2) benefit noticeably from higher dimensions ($+0.6\%$ on CARTE, $+1.4\%$ on TextTab moving from $k = 16$ to $k = 128$).



(a) Online Prediction Latency per Row (ms): CASE achieves superior predictive quality while maintaining faster online inference than TF-IDF due to a constant downstream feature dimension budget.

(b) Offline Fit Latency (s): Heavy backbone compute is isolated to training time, scaling predictably with sample size.

Figure 17: Predictive Quality vs. Computational Cost Trade-offs on CARTE across subsampled training set sizes $N \in \{128, 256, 512, 1024, 2048, 4096, 8192\}$. Pareto frontiers demonstrate that CASE offers highly competitive inference latency while improving predictive performance over non-semantic baselines.

D Computational Efficiency and Runtime Analysis

While integrating a Tabular Language Model (TLM) into tabular learning pipelines introduces additional computational overhead compared to shallow feature extraction, CASE is explicitly designed to maintain a practical trade-off between rich semantic context and runtime throughput.

Architectural Optimizations: The efficiency of CASE relies on two key architectural principles:

1. **KV-Cache Context Priming:** Background row representations and table-level semantics are processed and cached during `fit()`. At prediction time, context rows do not need to be re-encoded, avoiding redundant forward passes.
2. **Generation-Free Inference:** Unlike standard LLM applications that rely on autoregressive token sampling – the dominant computational bottleneck in LLM inference – CASE extracts target representations from the final hidden states in a single, fast forward pass without invoking the language modeling head.

Online Prediction vs. Offline Fitting Trade-off: A key advantage of CASE is the isolation of heavy neural compute to the offline fitting phase, `fit()`. At inference time, `predict()`, CASE projects row embeddings into a fixed $k = 32$ PCA feature budget regardless of the number of text columns (C) in the input table. In contrast, standard TF-IDF pipelines typically extract up to 32 n-gram components *per text column*, expanding the downstream feature space to $32 \times C$. For tables with numerous textual attributes, this dramatic expansion increases the attention complexity of downstream tabular architectures (such as TabICL).

Empirical Latency Benchmarks: To quantitatively evaluate throughput and predictive quality, we benchmarked TabICL, TabICL [TF-IDF], and TabICL [CASE-12B] across subsampled training set sizes $N \in \{128, 256, 512, 1024, 2048, 4096, 8192\}$ on 11 classification tasks from the CARTE benchmark while keeping the test evaluation sets fixed. All experiments were conducted on a single NVIDIA H100 GPU. As illustrated in Figure 17, CASE consistently defines the Pareto-optimal frontier for the prediction latency.

E Data Contamination and Leakage Analysis

As CASE leverages the pretrained Gemma 3 backbone and utilizes the T4 dataset [9] for continued pretraining, we proactively address the potential for data contamination:. Given that our evaluation benchmarks – CARTE, TextTab, and TabArena – are open-source, there is a risk that samples from

Table 7: Performance comparison on the STRABLE benchmark. Models are ordered by overall mean rank across all classification and regression tasks. Best performance is highlighted in **bold**.

Model Name	Overall Rank (↓)	Accuracy (%) (↑)	R^2 (%) (↑)
TabICLv2 [CASE-12B]	**2.0**	**76.3**	**72.8**
AutoGluon	2.3	75.7	69.9
ConTextTab	2.8	74.6	69.2
RealMLP	3.3	71.9	68.0
TabPFN-2.6	5.0	65.3	54.8
TabICLv2	5.5	65.1	49.1
Naive	6.7	56.0	–4.7

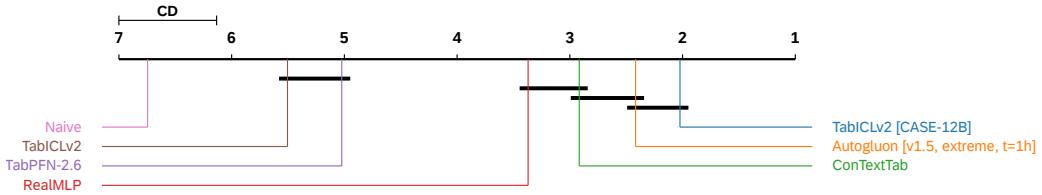


Figure 18: Critical difference diagram across the STRABLE benchmark.

these datasets were present in the LLM's original pretraining corpus or the T4 collection which we used for continued pretraining of the TLM.

T4 Contamination Study: Related work [28] conducted a systematic search for overlap between the T4 training set and the CARTE benchmark. By performing string similarity matching on unique combinations of column names and cell values, and found no evidence of CARTE samples within the T4 dataset. This suggests that the performance gains observed via CASE are not the result of direct leakage during our continued pretraining phase.

Backbone Memorization: While the exact composition of the Gemma 3 pretraining data is proprietary, our empirical results suggest that the model has not memorized these benchmarks in a way that bypasses tabular reasoning. As shown in our various ablations, the model's predictive accuracy is highly sensitive to the number of context rows provided in the KV cache. If the model were relying on memorized labels, we would expect a high baseline performance with zero or minimal context; instead, we observe a clear monotonic improvement as more in-context information is provided. This behavior indicates that the model is actively conditioning on the provided tabular distribution at inference time rather than retrieving stored labels from its weights.

F Evaluation on STRABLE Benchmark

To further assess the generalization of CASE on string-heavy tabular datasets, we extend our empirical evaluation to the recently introduced STRABLE benchmark suite [2]. STRABLE evaluates tabular learners across a diverse collection of classification and regression tasks characterized by high-cardinality unstructured text and complex string features.

We evaluate TabICLv2 augmented with CASE-12B against established tabular baselines, specialized string-tabular models (such as ConTextTab), and automated machine learning frameworks (AutoGluon).

CASE-12B achieves the top overall rank across the benchmark, outperforming domain-specific string pipelines like ConTextTab as well as strong AutoML ensembles like AutoGluon. Integrating CASE-12B into TabICLv2 provides a large performance boost over standard TabICLv2, lifting classification accuracy from 65.1% to 76.3% ($+11.2\%$) and regression R^2 from 49.1% to 72.8% ($+23.7\%$). Critical Difference (CD) diagram analysis ($\alpha = 0.05$) confirms that TabICLv2 [CASE-12B] is statistically superior to all evaluated baselines, with AutoGluon being the only competitor within the critical distance threshold, as shown in Figure 18.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We show the efficacy of our approach across a broad range of tabular benchmarks, comparing with a multitude of best-in-class tabular prediction baselines. We provide ablations on all core contributions of our paper, supporting the main claims of our contribution.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our approach in the final section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: We do not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We make the inference code and model checkpoints publicly available to reproduce our experiments on the publicly available benchmarks.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The inference code and model checkpoints will be made publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide all experimental details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide critical difference diagrams as well as average+min/max ranges across multiple seeds of our main results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report on the resources quired and used for training.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We conform to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: Beyond the societal impact that any tabular prediction model may have, our approach does not yield any specific potential societal impact unique to our work. Hence, we do not discuss general broader impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Does not apply.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We reference all original sources of data or model assets used in our work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not provide new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Does not apply.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Does not apply.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We describe the training procedure and application of the table-tuned Gemma LLM.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.