TLM: A Preliminary Exploration on Table-Language Model for Understanding Table Knowledge

Anonymous ACL submission

Abstract

Reasoning over structured tabular data remains a persistent challenge for large language models (LLMs), primarily due to the textual bottleneck: standard approaches serialize tables into raw text, fragmenting cell-level semantics into incoherent token sequences, while specialized models like TableGPT2 often sacrifice fine-grained detail through aggressive column-wise aggregation. We propose TLM (Table-Language Model), a multimodal framework that treats each table cell as an atomic semantic unit and preserves structural integrity natively. TLM integrates a lightweight, structure-aware table encoder with a frozen LLM backbone, explicitly modeling row–column dependencies and enabling the model to perceive tables as coherent two-dimensional grids rather than disjointed strings. Crucially, by aligning compact table representations with the LLM's embedding space, our approach leverages pre-trained linguistic priors as a stable reasoning engine—without exposing it to raw numerical noise or excessive sequence length. Evaluated on zero-shot table classification benchmarks, TLM achieves a mean accuracy of 0.7503, outperforming TableGPT2 by 20.39% and even surpassing XGBoost by 14.75% on complex relational reasoning tasks. The model attains this performance with dramatically shorter inputs, offering not only higher accuracy but also improved reliability and interpretability through preserved cell-level granularity. Code will be released upon publication.

1 Introduction

Large language models (LLMs) have achieved impressive performance across a wide range of natural language tasks (Sun et al., 2025). Yet, when applied to table-based question answering (TableQA), their effectiveness is fundamentally constrained by the mismatch between the sequential inductive bias of LLMs and the inherently two-dimensional structure of tabular data. To circumvent this, prevailing approaches adopt sequence-based paradigms, which linearize tables into HTML, Markdown, or natural language descriptions before feeding them into LLMs (Zhang et al., 2025).

Although straightforward to implement, this strategy suffers from three interrelated limitations. First, structural semantics are inevitably lost: the serialization process discards critical spatial relationships—such as row–column alignment, header hierarchies, and cross-cell dependencies—that are essential for accurate reasoning over complex queries. Second, it leads to severe computational inefficiency: even modest-sized tables yield extremely long token sequences, inflating attention complexity and often exceeding GPU memory capacity during training or inference. Third, these issues jointly limit real-world applicability, particularly in zero-shot settings where models cannot rely on task-specific fine-tuning to compensate for impoverished input representations.

To enable LLMs to truly understand tables-not merely read serialized approximations — we propose TLM (Table-Language Model), an end-to-end multimodal framework for native table understanding. As illustrated in Fig. 1, TLM departs from conventional text-based table serialization (Zhang et al., 2025; Saha et al., 2023; Wang et al., 2024; Hegselmann et al., 2023) by directly modeling the table's topology. Our architecture incorporates a lightweight yet expressive table encoder that jointly processes headers and cells, preserving column order, row-wise distributions, and inter-cell interactions. This encoder compresses the full tabular context into a compact set of dense semantic vectors, retaining structural fidelity while achieving high encoding efficiency.

To facilitate joint reasoning with textual inputs, we align these table representations with the embedding space of the LLM through a dedicated alignment module. During training, the encoder learns to map tabular semantics into a fixed-length

Method	Interpretability	Structural Knowledge	Cell Integrity	Feature Focus
XGBoost	✗	✓	✓	Numerical Patterns
TabPFN	✗	✓	✓	Distribution Statistics
Vanilla LLMs	✓	✗	✗	Surface Text Semantics
TableGPT2	✓	✓	△	Columnar Alignment
TLM (Ours)	✓	✓	✓	**Cell-centric Semantics**

Table 1: Comparison of key capabilities. While TableGPT2 (Su et al., 2024) introduces a table encoder, it focuses on **column-wise semantic alignment** (marked by △), which abstracts cell details into columnar tokens. **TLM** uniquely preserves **cell-level semantic integrity**, maintaining the holistic meaning of individual cell values.

sequence of table token embeddings, which are then seamlessly integrated into the LLM's input. This design enbales the language model's attention mechanism to dynamically focus on relevant table regions—such as specific columns or cell values—in response to the query, thereby supporting fluent, structure-aware answer generation. The resulting framework synergistically combines the structured precision of a purpose-built table encoder with the open-domain reasoning and generative power of LLMs. We evaluate TLM on multiple public benchmarks for zero-shot table classification, a demanding testbed for structural comprehension. Experimental results show that TLM significantly outperforms state-of-the-art baselines, achieving a mean accuracy of 0.7503, which represents a 20.39% absolute improvement over the specialized TableGPT2 (Su et al., 2024). Moreover, on complex relational reasoning tasks, TLM narrows the gap with traditional tree-based models and even surpasses XGBoost (Chen, 2016) by 14.75% on specific benchmarks.

In summary, this work makes three core advances: we demonstrate that LLM can effectively reason over structured tabular data through a simple yet structure-preserving table encoder, eliminating the need for lossy serialization; we introduce TLM, a novel multimodal framework that enables native, efficient, and interpretable table-language joint reasoning by aligning compact structural representations with LLM embeddings; and we provide comprehensive empirical validation across multiple datasets, showing that our approach significantly outperforms existing sequence-based methods (Zhang et al., 2025; Saha et al., 2023; Wang et al., 2024; Hegselmann et al., 2023) in both accuracy and efficiency, thereby establishing a new paradigm for scalable and practical table understanding. In essence, TLM shifts the TableQA paradigm from treating tables as serialized text to treating them as first-class structured objects, paving the way toward more capable and efficient multimodal LLMs.

2 Related Work

2.1 Traditional and Foundation Tabular Models

The landscape of tabular data modeling was long dominated by Gradient-Boosted Decision Trees (GBDTs) such as XGBoost (Chen, 2016) and LightGBM (Ke et al., 2017), which excel at capturing skewed numerical distributions. Early attempts to apply deep learning to tables (Yoon et al., 2020; Popov et al., 2019) struggled to surpass these baselines consistently. Recently, the field has shifted toward **Tabular Foundation Models (TFMs)**. Models like TabPFN (Hollmann et al., 2022) and TabDPT (Ma et al., 2024) leverage Transformers to perform in-context learning on small-scale data. However, these TFMs are often limited to pure classification or regression tasks, lacking the linguistic reasoning required for open-ended table understanding.

2.2 LLMs for Tabular Data Understanding

Directly prompting LLMs via **serialization** (e.g., Markdown or HTML) is the most common approach for TableQA (Zhang et al., 2025; Cheng et al., 2025). While fine-tuning on table-specific corpora (Gardner et al., 2024) improves performance, it does not resolve the fundamental issue of *token-wise dispersion*, where the internal semantics of a single cell are fragmented across arbitrary token boundaries, leading to high computational redundancy and structural loss.

To mitigate this, recent multi-modal approaches have introduced specialized encoders. TableGPT2 represents a significant milestone, utilizing a semantic encoder to capture tabular topology. However, TableGPT2 is fundamentally built upon a column-wise abstraction strategy, where multiple cell values are aggregated into a fixed set of

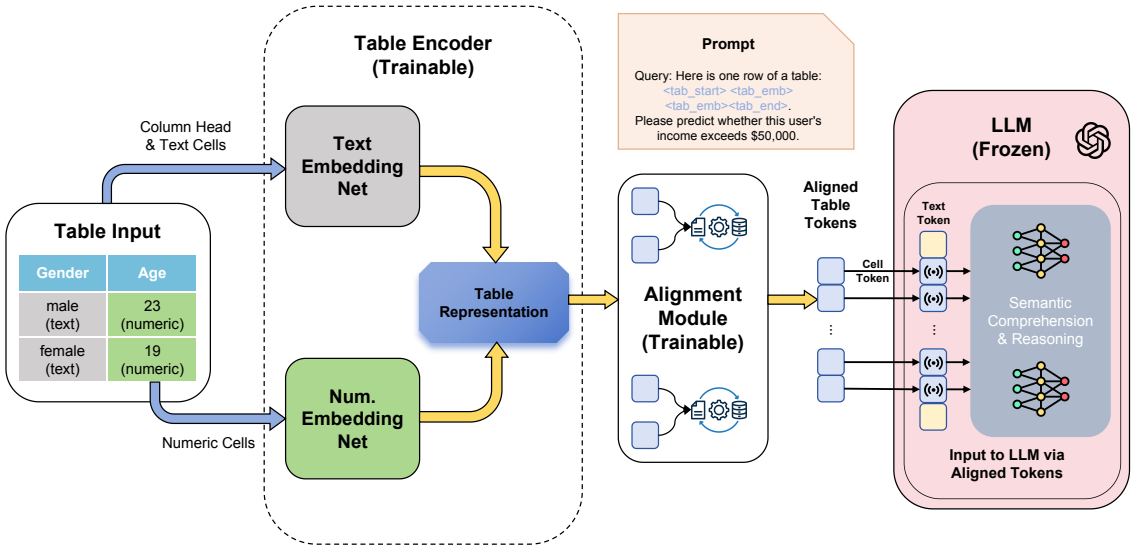

Figure 1: Overall architecture or TLM. Table encoder encodes column name and value information as table representations, which are aligned by alignment module to be further understood by LLM.

columnar tokens to maintain scalability for large databases. While effective for schema alignment, such over-aggregation inevitably leads to semantic distortion, as it discards the granular details and local context of individual cells. In contrast, our TLM prioritizes cell-level semantic integrity. By treating each cell as an atomic unit and preserving its holistic representation, TLM bridges the gap between high-level structural awareness and fine-grained data reasoning.

2.3 Multi-modal Alignment for Structured Data

Inspired by the success of vision-language models like LLaVA (Liu et al., 2024b), researchers have explored aligning diverse modalities with LLMs. In the graph and time-series domains, researchers either utilize frozen encoders with linear projections (Su et al., 2024) or sacrifice structural inductive biases for textual compatibility (Chen et al., 2024). Our framework introduces a fully trainable, cell-centric pre-training approach. Unlike previous methods, it maps structured representations into a semantic space grounded in LLM token embeddings, ensuring that the model retains its open-domain reasoning capacity while gaining native table understanding capabilities.

3 Methodology

In this section, we present **TLM**, an end-to-end multi-modal framework for native table understanding and reasoning. TLM consists of three key components: (1) a **Table Encoder** that captures both structural and semantic relationships within tables, (2) a **Table–Language Alignment Module** that projects table representations into a fixed number of table tokens compatible with the LLM, and (3) an **LLM Decoder** that performs both reasoning and zero-shot table classification. The main objective is to jointly train the table encoder and alignment module via instruction-based supervision adaptable to various table tasks. The overall framework is illustrated in Fig. 1, where table features are encoded, aligned, and fused with textual prompts for tabular task reasoning.

3.1 Task Formulation

We denote a tabular dataset as $\mathcal{D} = \{x_i^{cat}, x_i^{num}, y_i\}_{i=1}^{M}$, where $x_i^{cat} \in \mathbb{R}^{n_s \times d}$ represents categorical/textual features, $x_i^{num} \in \mathbb{R}^{n_d \times d}$ represents numerical features, and $y_i \in \{1, 2, \ldots, N\}$ is the target label, with N varying across datasets. Each sample is accompanied by an instruction or prompt x_i^{ins}, specifying the task type (e.g., cell reconstruction, reasoning explanation, or classification).

As illustrated in Fig. 2, our goal is to enable the model to perform structured reasoning over tabular data through a three-stage process: table reconstruction, intermediate reasoning, and final classification. Formally, let x_i^{rec} denote the reconstructed table, r_i the intermediate reasoning trace, and y_i the final prediction for sample i. Given the categorical and numerical features x_i^{cat}, x_i^{num}

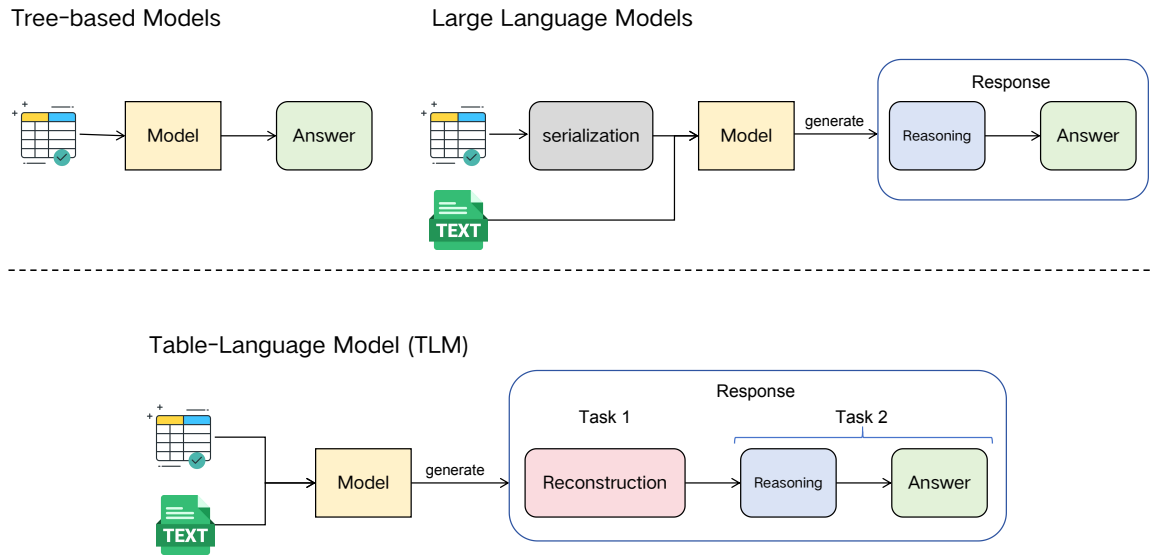

Figure 2: Comparision of different table model paradigm.

and the task instruction x_i^{ins}, the joint conditional probability can be factorized as:

$$\begin{aligned} &P_\theta(x_i^{rec}, r_i, y_i \mid x_i^{cat}, x_i^{num}, x_i^{ins}) \\ &\quad = P_\theta(x_i^{rec} \mid x_i^{cat}, x_i^{num}, x_i^{ins}) \\ &\quad \cdot P_\theta(r_i \mid x_i^{rec}, x_i^{cat}, x_i^{num}, x_i^{ins}) \\ &\quad \cdot P_\theta(y_i \mid r_i, x_i^{rec}, x_i^{cat}, x_i^{num}, x_i^{ins}). \end{aligned} \tag{1}$$

Here, the first term models the reconstruction of the table, providing a structured context of the original input. The second term generates an intermediate reasoning trace conditioned on the reconstructed table and original features, offering step-by-step interpretability. Finally, the third term predicts the target label by integrating both the reconstructed table and reasoning trace, allowing the model to leverage semantic, structural, and logical dependencies simultaneously. This decomposition encourages more interpretable and reliable predictions across tasks such as cell reconstruction, reasoning explanation, and classification.

3.2 Model architecture

3.2.1 Table Model

The table encoder is designed to process structured tabular data containing both sparse and dense features. Inspired by prior work such as TransTab (Wang and Sun, 2022), we treat a table as a structured composition of two fundamental feature types: categorical/textual and numerical. Each feature type is encoded with a modality-aware strategy that preserves both semantic and structural information.

Sparse features. Categorical and textual features are treated as sparse modalities. For each sparse cell, we embed both the column name and the cell value using a shared text embedding network and combine them element-wise:

$$E_s = E_{\text{cell}} \oplus E_{\text{col}}, \tag{2}$$

where $\oplus$ denotes element-wise addition, producing $E_s \in \mathbb{R}^{n_s \times d}$.

Dense features. Numerical columns are treated as dense modalities and processed in the same way as sparse features. For each numerical cell, the scalar value x is first mapped to a vector via a numerical embedding net:

$$E_{\text{num}} = f_{\text{num}}(x), \tag{3}$$

where f_{num} denotes a learnable linear embedding function. The cell embedding is then obtained by combining it with the column embedding E_{col} through element-wise addition:

$$E_d = E_{\text{num}} \oplus E_{\text{col}}. \tag{4}$$

This design preserves value sensitivity and column semantics while keeping dense and sparse embeddings in the same representation space.

Unified table representation. Finally, sparse and dense embeddings are concatenated into the overall table representation:

$$\tilde{E} = \text{CONCAT}(E_s, E_d) \in \mathbb{R}^{n \times d}, \tag{5}$$

where $n = n_s + n_d$ is the total number of cells. This unified design treats all cells in a consistent way, preserves both semantic and structural information, and provides a robust input for alignment with the LLM and downstream reasoning tasks.

3.2.2 Table-Language Alignment Module

To effectively bridge the gap between structured table representations and language-based reasoning, TLM employs a lightweight MLP-based table-language alignment module that maps table embeddings into the semantic space of the frozen LLM. This module serves as a differentiable bridge, transforming heterogeneous table features—derived from categorical and numerical encoders—into a unified latent representation suitable for language reasoning.

Specifically, each table row is encoded into a sequence of contextualized cell embeddings $\{e_1, e_2, \ldots, e_m\}$, where m denotes the number of cells in the row, and each $e_i \in \mathbb{R}^d$ integrates both the semantic meaning of the column and statistical properties of its values. The alignment module projects these embeddings into a shared latent space:

$$z_i = \mathrm{MLP}(e_i), \quad i = 1, \ldots, m, \tag{6}$$

where the MLP consists of two fully connected layers with non-linear activation GELU to capture higher-order feature interactions.

To maintain structural awareness, we insert special boundary tokens $\langle$tab_start$\rangle$ and $\langle$tab_end$\rangle$, and sequentially concatenate the projected tokens to form a table-aware prompt:

$$\text{Prompt} = [\langle\text{tab\_start}\rangle, z_1, z_2, \ldots, z_m, \langle\text{tab\_end}\rangle]. \tag{7}$$

The boundary tokens explicitly signal row or table boundaries, enabling the LLM to preserve structural information during cross-modal attention.

This prompt is then fused with the textual query and fed into the frozen LLM. The LLM jointly attends to both linguistic and tabular embeddings, performing contextual reasoning without fine-tuning. By maintaining a lightweight, fully differentiable mapping, the alignment module achieves efficient cross-modal fusion while preserving the pre-trained linguistic competence of the LLM. In practice, this design enables *multi-modal table understanding*, *zero-shot table reasoning*, and enhanced interpretability through intermediate token representations, which can be further analyzed for explainable reasoning.

Overall, the MLP-based alignment provides a robust, scalable, and easily trainable bridge between structured table semantics and language models, complementing the table encoder and supporting downstream tasks efficiently.

3.3 Joint Multi-modal Training

To bridge the gap between tabular and linguistic modalities, TLM employs a unified Next-Token Prediction (NTP) objective. We jointly optimize the table encoder and the alignment module while keeping the LLM backbone frozen, thereby preserving its vast linguistic reasoning priors.

Training Data and Supervision We supervise the model using a *Progressive Reasoning-Augmented* dataset. Crucially, we adopt a dual-stream strategy for numerical values: while continuous data are discretized into quantile-based interval ranges (e.g., "[1050.0, 1200.0]") to provide explicit semantic anchors for textual reasoning traces, the table encoder always receives the raw float values as input. This design allows the model to ground its high-level logical deductions in precise numerical representations without losing granular information to discretization. Detailed data construction and the interval-mapping mechanism are provided in Appendix A.1.

Formulation and Objective Given a table $\mathcal{T}$, the encoder produces cell-level representations $\mathbf{E} \in \mathbb{R}^{n \times d}$. The alignment module f_{align} transforms these into a sequence of table tokens $\mathbf{Z} = f_{\text{align}}(\mathbf{E})$. Let $\mathcal{I}$ denote the instruction tokens (comprising the query and task-specific prompts). The table tokens $\mathbf{Z}$ are then interleaved into the input stream of the frozen LLM decoder alongside $\mathcal{I}$ to form the multimodal context.

The model autoregressively generates a target sequence $\mathcal{Y} = [rec; r; y]$, where rec is the cell reconstruction, r the reasoning trace, and y the final answer. The training objective minimizes the negative log-likelihood:

$$\mathcal{L} = -\sum_{t=1}^{|\mathcal{Y}|} \log P_\theta(\mathcal{Y}_t \mid \mathcal{Y}_{<t}, \mathbf{Z}, \mathcal{I}), \tag{8}$$

where θ represents the trainable parameters of the encoder and alignment module. By conditioning on diverse tasks—from surface-level reconstruction to deep logic synthesis—TLM learns to produce table embeddings that are both semantically grounded and instruction-aware.

4 Experiments

4.1 Experimental Setup

Datasets We evaluate TLM on a comprehensive suite of nine publicly available tabular datasets from OpenML (Vanschoren et al., 2014) and OpenTab (Kong et al., 2024), as summarized in Table 2. These benchmarks cover diverse domains and feature types, ranging from low-dimensional categorical tasks (e.g., *wilt*) to high-dimensional numerical reasoning (e.g., *sick*). We follow the standard 80/20 train-test split for all datasets. To maintain consistency, all tasks share a unified instruction-based template, where TLM utilizes structured embeddings while baselines use textual serialization. A detailed prompt example for the Adult dataset is provided in Appendix A.3.

Name	Instances	Features	Source
sick	3,772	30	OpenML
jm1	10,885	22	OpenML
churn	5,000	21	OpenML
adult	48,842	15	OpenML
wilt	4,839	6	OpenML
bank	11,162	16	OpenTab
credit	16,714	10	OpenTab
eye_movements	7,608	22	OpenTab
law_school_admission	20,800	11	OpenTab

Table 2: Summary of datasets used in the experiments.

Baselines We compare TLM against a diverse set of baselines: (1) **Tree-based Models**: Random Forest and XGBoost, which represent the state-of-the-art in classical tabular learning. (2) **Large Language Models**: Standard instruction-tuned models including Qwen2.5 (7B/72B), Qwen3-235B-Thinking (Bai et al., 2023), gpt-oss-120b (Agarwal et al., 2025), and DeepSeek-V3.1 (Liu et al., 2024a). (3) **Specialized Tabular LLMs**: TableGPT2 (Su et al., 2024), a state-of-the-art model explicitly pre-trained on massive tabular corpora.

For all LLM-based baselines (categories 2 and 3), tabular inputs are converted into optimized textual sequences (e.g., Markdown or CSV formats) to ensure the best possible performance within their native linguistic modality.

Evaluation Metrics We adopt **Accuracy** as the primary metric. Additionally, we report the **Reconstruction Rate** in our ablation studies to assess the fidelity of cell-level latent representations.

Implementation TLM is built upon the InternVL framework (Wang et al., 2025) using a frozen Qwen2.5-7B-Instruct backbone. Only the table encoder and alignment module are trainable. Training is performed on 8 NVIDIA A100 GPUs with a total batch size of 512 for 50 epochs. Numerical features are log-scaled as $\tilde{x} = \text{sign}(x)\log(1+|x|)$ to improve optimization stability by normalizing disparate feature scales and preventing gradient explosion. To facilitate reproducibility, we list the detailed hyper-parameters for training TLM in Table 3. All experiments are implemented in PyTorch with DeepSpeed `zero-stage1`.

Configuration	Value
Optimizer	Adam
Learning Rate	2×10^{-4}
Learning Rate Schedule	Cosine Decay (5% warmup)
Batch Size	512
Weight Decay	0.01
Numerical Scaling	Logarithmic
Sampling (Temp/Top-p)	0.7 / 1.0
Hardware	8 × NVIDIA A100 (80GB)

Table 3: Hyper-parameter settings for TLM training.

4.2 Overall Performance

Table 4 summarizes the performance of TLM against traditional tree-based models and SOTA LLMs.

Dominance over Tabular LLMs TLM demonstrates consistent superiority over LLM baselines of significantly larger scales. Despite utilizing the Qwen2.5-7B backbone, TLM achieves a mean accuracy of **0.7503**, outperforming the specialized TableGPT2 (0.5464) by 20.39%. We attribute this to the "textual bottleneck" inherent in linearized baselines. In traditional LLMs, tabular data is flattened into token sequences, forcing the model to reconstruct spatial dependencies from scratch. This process often leads to *structural distortion*, where the categorical boundaries and numerical magnitudes are confounded by the language model's vocabulary bias. In contrast, TLM's cell-level encoding natively preserves the atomic integrity of each field. By bypassing textual serialization, TLM ensures that the LLM perceives the table through aligned embeddings that maintain explicit coordinate-awareness, thereby preserving the tabular topology essential for precise inference.

Narrowing the Gap with Tree Models Tree-based ensembles (mean 0.8298) remain strong benchmarks due to their inherent inductive bias for axis-aligned splitting and handling of non-smooth

Method	adult	jm1	sick	churn	wilt	eye_mov.	bank	credit	law.	Mean
Tree-based Models (Traditional Baselines)										
Random Forest	0.8512	0.8163	0.9714	0.9210	0.9752	0.6005	0.7922	0.7727	0.7067	0.8230
XGBoost	0.8579	0.8158	0.9732	0.9430	0.9783	0.6084	0.8061	0.7667	0.7195	0.8298
Large Language Models										
Qwen2.5-7B	0.7306	0.4768	0.8415	0.7550	0.3801	0.5026	0.5584	0.5109	0.7156	0.6079
Qwen2.5-72B	0.7894	**0.7344**	0.7446	0.7730	0.8614	0.4862	0.5759	0.4454	0.6963	0.6785
Qwen3-235B	0.7103	0.6100	0.7482	0.7300	0.4896	0.5088	0.5848	0.4469	0.5521	0.5978
DeepSeek-V3.1	**0.8248**	0.6313	0.8245	0.7117	0.1592	**0.5129**	0.5924	0.4332	0.3870	0.5641
TableGPT2	0.5824	0.5269	0.6052	0.4785	0.5666	0.4942	0.4935	0.4862	0.6845	0.5464
TLM (ours)	0.7728	0.6463	**0.9321**	**0.8360**	**0.9163**	0.4894	**0.7532**	**0.5402**	**0.8670**	**0.7503**

Table 4: Overall performance comparison. **Bold** indicates the best among LLMs. Underline denotes surpassing all baselines. Shaded blue cells highlight TLM's superior alignment capabilities.

data distributions. However, TLM significantly narrows this gap, even outperforming XGBoost on the *law_school_admission* dataset by 14.75% (0.8670 vs. 0.7195). We argue that while GBDTs excel at local pattern recognition, they often lack the global relational reasoning required for complex, schema-conditioned datasets. TLM's unified embedding space allows the LLM to perform *cross-feature synthesis*—leveraging its vast pre-trained knowledge to interpret feature correlations that a static tree might treat as independent partitions. This suggests that for tasks requiring high-level semantic grounding, the neural-based relational bias in TLM provides a more robust alternative to traditional gradient boosting.

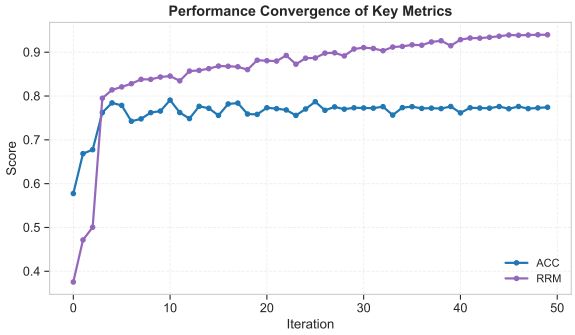

Figure 3: Evaluation performance of TLM on adult dataset during training.

4.3 In-depth Analysis and Ablation

We conduct a comprehensive analysis of TLM's design choices on the *adult* dataset (Table 5).

Structural Alignment as a Semantic Anchor As monitored in Fig. 3, ACC and RRM exhibit synchronous convergence. This coupling proves the cell reconstruction task acts as a "semantic anchor." By grounding latent table tokens in their original values, we ensure the embedding space $\mathbf{Z}$ remains faithful to the data, effectively mitigating the hallucination issues common in pure-textual tabular models.

Task Integration vs. Decomposition To investigate the optimal multi-task learning paradigm, we compare our Sequential Task Integration with a Parallel Task Decomposition baseline. The former integrates reconstruction and reasoning into a unified, sequential pipeline (Reconstruction $\rightarrow$ Reasoning), while the latter treats them as independent, simultaneous objectives sharing a common encoder.

As revealed in Table 5, while both paradigms maintain comparable classification accuracy, the reconstruction rate (RRM) collapses under the decomposition setting (0.9398 $\rightarrow$ 0.4989). We hypothesize that this stems from a severe gradient conflict between semantic alignment and discriminative classification. In a decomposition framework, the table encoder is forced to optimize for two potentially disjoint manifolds: one preserving high-fidelity structural details for reconstruction, and another abstracting away such details for category-level discrimination. This lack of hierarchy leads to mutual interference, where classification signals overwrite the fragile structural priors. In contrast, our sequential integration respects the functional dependency of tabular tasks—establishing structural grounding as a prerequisite for reasoning—thereby ensuring that the latent space $\mathbf{Z}$ remains robustly anchored to the table's atomic structure.

Category	Configuration	ACC ↑	RRM ↑
Ours	**TLM (Full Model)**	**0.7728**	**0.9398**
Components	w/o Alignment Training	0.5531	0.1802
	w/o Encoder Training	0.7699	0.9170
Paradigm	Task Integration	**0.7728**	**0.9398**
	Task Decomposition	0.7707	0.4989
Architecture	Base Encoder	**0.7728**	**0.9398**
	+4 Transformer Layers	0.5809	0.2340
Parameters	Frozen LLM Backbone	**0.7728**	**0.9398**
	Unfrozen LLM Backbone	0.7030	0.5552

Table 5: Systematic ablation study on the Adult dataset. Each color block highlights a coherent category of experimental settings. **Bold** denotes the optimal configuration selected for TLM.

Optimization Dynamics: Frozen vs. Unfrozen LLMs We observe a significant performance decay when unfreezing the 7.6B LLM backbone. This is primarily driven by gradient dominance, where the massive LLM parameter space overwhelms the 16M alignment module. To address this, we explored a differential learning rate strategy (2×10^{-4} for the encoder vs. 1×10^{-6} for the LLM), yet performance remained suboptimal.

We hypothesize that this failure stems from the nature of tabular knowledge: unlike text, tabular data is predominantly composed of numerical distributions and structural dependencies, which contain little "linguistic knowledge" beneficial for updating a pre-trained LLM. Updating the backbone may inadvertently corrupt the LLM's vast linguistic priors with low-level data noise. Instead, the optimal strategy is to treat the LLM as a reasoning engine rather than a knowledge storage for tables. By keeping the LLM frozen, we allow it to focus on structural manipulation and reasoning over the table—essentially learning "how to read and process" the aligned embeddings $\mathbf{Z}$—rather than attempting to assimilate the raw data distribution into its weights. This decoupling of structural perception and linguistic reasoning yields the most stable and effective alignment.

The Non-linearity Bottleneck in Encoders Increasing encoder complexity (stacking Transformer layers) surprisingly hurts performance. Tabular features, unlike images or text, often have more direct mappings. Excessive non-linear transformations may disrupt the explicit spatial-semantic correspondence that the LLM expects, creating a "representation drift" that hinders zero-shot reasoning.

5 Conclusion

In this paper, we presented TLM, a multi-modal Table-Language Model designed to bridge the gap between structured tabular data and linguistic reasoning. The core of our approach lies in the preservation of cell-level semantic integrity, treating each cell as an atomic unit to avoid the information distortion prevalent in high-level columnar abstraction. By integrating a structure-aware encoder with a progressive reasoning-augmented training paradigm, TLM successfully aligns raw numerical floats with quantile-based interval ranges, enabling precise and interpretable zero-shot reasoning. Extensive experiments across multiple benchmarks demonstrate that TLM significantly outperforms both vanilla LLMs and specialized tabular models, achieving state-of-the-art performance while substantially reducing input sequence lengths. Our results underscore the importance of native structural encoding and cell-centric representations in unlocking the full potential of LLMs for tabular data. In future work, we plan to extend TLM to large-scale database reasoning and explore more complex multi-table relational tasks.

Limitations

Despite the performance gains and structural insights provided by TLM, several limitations remain to be addressed in future work:

Gap with Tree-based Models While TLM significantly narrows the performance gap between neural-based models and gradient-boosted decision

trees (GBDTs), it still falls short of XGBoost and Random Forest on certain datasets. This remains a common challenge across the LLM-for-tabular field. GBDTs possess an inherent inductive bias for axis-aligned splitting and handling non-smooth data distributions, which even high-capacity LLMs struggle to fully replicate through modal alignment alone.

Scope of Tabular Tasks This study primarily focuses on binary classification within single-table scenarios. The generalizability of our progressive training paradigm to more complex tasks—such as *multi-class classification*, *regression*, or *TableQA* involving multiple relational tables—has not yet been extensively validated.

Schema Rigidity A practical constraint of the current TLM architecture is its schema dependency. Specifically, the model requires the table headers during inference to be strictly consistent with those encountered during the structural alignment phase. This lack of *zero-shot schema adaptability* means that any alteration in feature names or the introduction of unseen columns requires re-alignment or fine-tuning. Future work should explore more flexible encoding strategies to handle dynamic and heterogeneous schemas.

Scalability to Large-scale Tables Although our cell-level embedding approach significantly reduces the token count compared to textual serialization, the model still operates within the fixed context window of the backbone LLM. For massive tables with thousands of rows or extremely high-dimensional feature spaces, the current architecture may still encounter memory constraints and increased computational latency during inference compared to lightweight tree models.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tianqi Chen. 2016. Xgboost: A scalable tree boosting system. *Cornell University*.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and 1 others. 2024. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61.

Mingyue Cheng, Qingyang Mao, Qi Liu, Yitong Zhou, Yupeng Li, Jiahao Wang, Jiaying Lin, Jiawei Cao, and Enhong Chen. 2025. A survey on table mining with large language models: Challenges, advancements and prospects. *Authorea Preprints*.

Josh Gardner, Juan C Perdomo, and Ludwig Schmidt. 2024. Large scale transfer learning for tabular data via language modeling. *Advances in Neural Information Processing Systems*, 37:45155–45205.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. Tabllm: Few-shot classification of tabular data with large language models. In *International conference on artificial intelligence and statistics*, pages 5549–5581. PMLR.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. 2022. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Kezhi Kong, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Chuan Lei, Christos Faloutsos, Huzefa Rangwala, and George Karypis. 2024. Opentab: Advancing large language models as open-domain table reasoners. *arXiv preprint arXiv:2402.14361*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024b. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306.

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

Sergei Popov, Stanislav Morozov, and Artem Babenko. 2019. Neural oblivious decision ensembles for deep learning on tabular data. *arXiv preprint arXiv:1909.06312*.

Swarnadeep Saha, Xinyan Yu, Mohit Bansal, Ramakanth Pasunuru, and Asli Celikyilmaz. 2023. Murmur: Modular multi-step reasoning for semi-structured data-to-text generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11069–11090.

Aofeng Su, Aowen Wang, Chao Ye, Chen Zhou, Ga Zhang, Gang Chen, Guangcheng Zhu, Haobo Wang, Haokai Xu, Hao Chen, and 1 others. 2024. Tablegpt2: A large multimodal model with tabular data integration. *arXiv preprint arXiv:2411.02059*.

Jiankai Sun, Chuanyang Zheng, Enze Xie, Zhengying Liu, Ruihang Chu, Jianing Qiu, Jiaqi Xu, Mingyu Ding, Hongyang Li, Mengzhe Geng, and 1 others. 2025. A survey of reasoning with foundation models: Concepts, methodologies, and outlook. *ACM Computing Surveys*, 57(11):1–43.

Joaquin Vanschoren, Jan N Van Rijn, Bernd Bischl, and Luis Torgo. 2014. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, and 1 others. 2025. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Zifeng Wang and Jimeng Sun. 2022. Transtab: Learning transferable tabular transformers across tables. *Advances in Neural Information Processing Systems*, 35:2902–2915.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and 1 others. 2024. Chain-of-table: Evolving tables in the reasoning chain for table understanding. *arXiv preprint arXiv:2401.04398*.

Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela Van der Schaar. 2020. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in neural information processing systems*, 33:11033–11043.

Xiaokang Zhang, Sijia Luo, Bohan Zhang, Zeyao Ma, Jing Zhang, Yang Li, Guanlin Li, Zijun Yao, Kangli Xu, Jinchang Zhou, and 1 others. 2025. Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10315–10344.

A Data Construction and Implementation

A.1 Numerical Interval Mapping

To preserve numerical fidelity, we implement a Dynamic Interval Mapping strategy. While the table encoder processes raw `float` inputs, the reasoning traces utilize quantile-based interval ranges (e.g., "$[1050.0, 1200.0]$") as semantic anchors. This allows the frozen LLM to ground its logic in structured partitions without losing fine-grained information.

A.2 Progressive Training Paradigm

To synchronize the disparate modalities of the structure-aware encoder and the frozen decoder, we adopt a Progressive Training Paradigm. We generate high-quality reasoning traces using a fill-in-the-middle (FIM) paradigm with *Qwen3-235B-Thinking*. The training is partitioned into three incremental stages to stabilize the cross-modal alignment:

1. **Feature Reconstruction**: The model is first tasked with mapping latent table tokens $\mathbf{Z}$ back to their human-readable cell values. This ensures the alignment module successfully translates the encoder's grid-based features into the LLM's vocabulary.

2. **Latent Reasoning**: Building upon the reconstruction, the model predicts intermediate reasoning logic r. Here, the model learns to synthesize multiple cell attributes into cohesive evidence.

3. **Task Conclusion**: Finally, the model generates the classification/regression label y. This stage bridges the gap between structured evidence and high-level decision making.

Algorithm 1: Progressive Data Construction

Input: Table T, Query q, Answer a
Output: Dataset $\mathcal{D} = \{\mathcal{D}_{recon}, \mathcal{D}_{reason}, \mathcal{D}_{conclude}\}$
foreach *row* $r \in T$ **do**
  foreach *column* $c \in r$ **do**
    if $c.type$ == *numerical* ***and*** `IsContinuous`(c) **then**
      $v \leftarrow$ `GetIntervalRange`$(c.value)$;
    else
      $v \leftarrow c.value$ (or normalized);
    `row_info` $\leftarrow$ `Append`$(c.name, v)$;
  `trace` $\leftarrow$ `FIMGenerate`(`TeacherLLM`, `row_info`, q, a);
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{$`row_info`$, q, a,$ `trace`$\}$;

A.3 Prompt and Reasoning Example

Figure 4 showcases a comprehensive sample from the *Adult* dataset. A critical observation is the

model's ability to utilize the <tab_emb> tokens to perform **reconstructive grounding**. Before making a judgment, the model explicitly restates the input features. This self-verification step confirms that the cell-level integrity is preserved throughout the transformation from numerical embeddings to linguistic tokens.

B Granular Alignment Behavior

As illustrated in Fig. 5, categorical features (e.g., *race*, *sex*) achieve near-perfect reconstruction rapidly. In contrast, continuous attributes (e.g., *age*, *hours-per-week*) converge more slowly, requiring the alignment module to learn more complex mapping functions between the float-based encoder space and token-based decoder space.

Input

Task Description: Please predict whether the user's annual income exceeds $50,000 based on the given features.
Output Requirements:

- First, restate the user's features.
- Then, provide the prediction with detailed reasoning explaining the judgment logic.
- Only output 0 or 1, where 0 means income $\leq$ $50,000 and 1 means income $>$ $50,000.

User Features:
`<tab_start> <tab_emb> * n <tab_end>`

Expected Output

Input Features:

```
age: 25
workclass: Private
education: HS-grad
education-num: 9
marital-status: Never-married
occupation: Craft-repair
relationship: Own-child
race: White
sex: Male
capital-gain: 0
capital-loss: 0
hours-per-week: 40
native-country: United-States
```

Reasoning:

- Age 25: Early career stage, limited experience and earning potential.
- Workclass Private: Moderate earnings without seniority or specialization.
- Education HS-grad: High school education typically below $50,000 annual income.
- Occupation Craft-repair: Skilled trade, rarely exceeds $50,000.
- Capital-gain/loss 0: No additional investment income.
- Hours per week 40: Standard full-time, moderate wage.
- Marital status & relationship: Limited impact on income.
- Demographics White male, US: Slight statistical advantage, insufficient to offset other factors.

Classification: 0

Figure 4: **Example of prompt and expected output for TLM.** The input specifies a schema-conditioned prediction task, and the model is required to reason step-by-step before outputting the classification label.

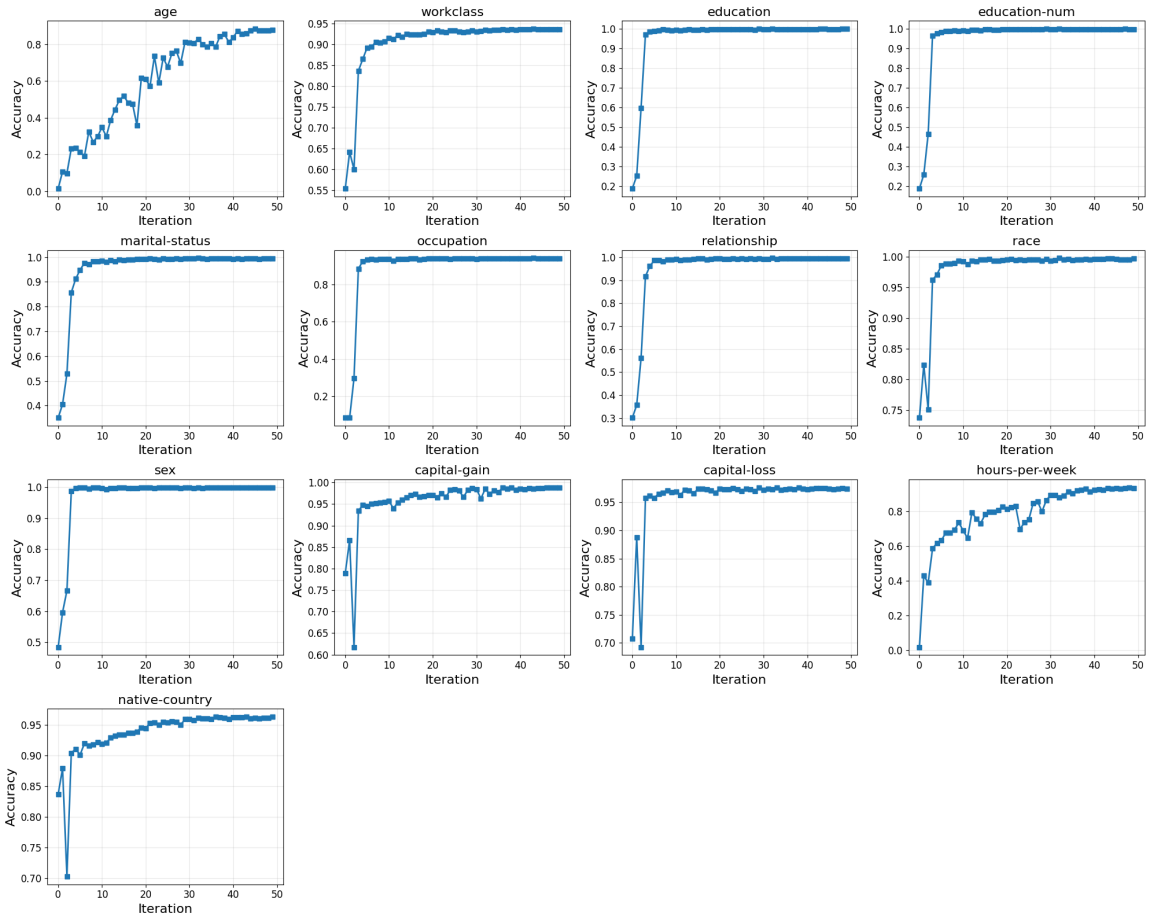

Figure 5: Fine-grained alignment performance on different columns of adult dataset (measured by cell reconstruction ratio).