TWT: ENHANCING MULTI-MODAL TABULAR UNDERSTANDING VIA NEURO-SYMBOLIC REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal Large Language Models (MLLMs) have demonstrated remarkable reasoning capabilities across modalities such as images and text. However, tabular data, despite being a critical real-world modality, remains relatively underexplored in multimodal learning. In this paper, we focus on the task of Tabular-Vision Multi-Modal Understanding (TVMU) and identify three core challenges: (1) high structural variability and data incompleteness in tables, (2) implicit and complex feature dependencies, and (3) significant heterogeneity in problem-solving pipelines across downstream tasks. To address these issues, we propose Thinking with Tables (TWT). TWT employs a program-aided code-based neuro-symbolic reasoning mechanism that facilitates key operations, such as information extraction and element modeling, by interacting with external environments. We evaluate TWT on eight representative datasets. Experimental results demonstrate that TWT consistently outperforms existing baselines by an average of 10% in accuracy, achieving performance comparable to, or even surpassing, proprietary commercial SOTA LLMs on TVMU tasks.

1 INTRODUCTION

In recent years, MLLMs (Achiam et al., 2023; Bai et al., 2025; Team, 2025; Team et al., 2023) have advanced rapidly. By jointly modeling heterogeneous modalities such as text and images, MLLMs have achieved remarkable success in tasks including image understanding (Shen et al., 2025) and visual question answering (Zhang et al., 2025b), demonstrating strong complex reasoning capabilities (Peng et al., 2025; Team, 2024; Guo et al., 2025). In real-world applications, tabular data constitute an equally common and important modality (Altman & Krzywinski, 2017), widely present in domains such as finance (Ozbayoglu et al., 2020), healthcare (Ching et al., 2018), and industrial manufacturing (Hein et al., 2017), where they play a crucial role in information organization, data analysis, and decision support (Salehpour & Samadzamini, 2024). However, existing research on MLLMs has primarily focused on modalities such as text and images, with relatively limited attention paid to multimodal understanding involving tabular data.

Tabular-Vision Multi-Modal Understanding (TVMU) (Pasupat & Liang, 2015; Howard et al., 2018; Alonso et al., 2025) aims to extract and reason over tables from images using textual and table-associated visual information (Dang et al., 2024). It primarily encompasses two core tasks: multimodal table question answering (Lu et al., 2022) and multimodal table prediction (Howard et al., 2021). Code-based neuro-symbolic approaches that use symbolic code for structured logic and computation are widely regarded as promising solutions for such tasks (Chen et al., 2022; Gao et al., 2023). In the Table QA domain, methods represented by Chain-of-Table (Wang et al., 2024) decompose complex queries into a sequence of predefined table operations and derive the final answer by executing these operations step by step. For Table Prediction tasks, OCTree (Nam et al., 2024) leverages the background knowledge of language models to assist feature engineering in the form of executable code, thereby improving model performance.

However, these methods typically rely on complete and well-structured tables (Jiang et al., 2025b; Jiang et al.), are constrained by a limited set of predefined operations (Chegini et al., 2025; Yang et al., 2024), and are often designed for a single task setting. To enhance the capability of MLLMs in TVMU tasks, a unified framework that simultaneously supports different downstream tasks is required. For example, in financial analysis (Ozbayoglu et al., 2020), key information in complex

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

research reports is often scattered across structured tables, unstructured textual analyses, and trend charts. Analysts must not only predict future growth from historical tabular data but also reason over textual market dynamics to identify the causes of profit fluctuations. In such settings, tables may be incomplete or poorly formatted, and critical decision-relevant information spans multiple modalities, making reliance on tables alone insufficient. As a result, robust modeling and reasoning for TVMU in realistic multimodal environments face three major challenges: 1) structural variability and data incompleteness in real-world applications, 2) implicit and complex feature dependencies, and 3) the significant heterogeneity of task requirements.

To address the above challenges, we propose **Thinking with Tables** (TWT), **a program-aided neuro-symbolic reasoning framework** designed for Tabular-Vision Multi-Modal Understanding tasks. TWT performs code-based neuro-symbolic reasoning by modeling table understanding as an interactive reasoning process within an external tabular environment, enabling multi-step interactions for dynamic information extraction, key element modeling, and so on. This design decomposes complex table reasoning into a sequence of interpretable intermediate steps. Experiments on eight datasets covering both multimodal table question answering and multimodal table prediction tasks demonstrate consistent and significant performance improvements, validating the effectiveness and robustness of TWT in complex TVMU tasks.

The main contributions of this work are summarized as follows:

1. **TVMU Problem Formulation and Analysis**. We provide a systematic analysis of the TVMU task, pointing out key challenges including table structural variability, information incompleteness, complex feature dependencies, and task heterogeneity.
2. **Noval Neuro-Symbolic Reasoning Mechanism**. We introduce **Thinking with Tables**, a novel neuro-symbolic framework that enables robust modeling of complex tabular information through a code-based interactive reasoning mechanism.
3. **Superior Empirical Performance** Extensive experiments on diverse datasets confirm that TWT consistently outperforms baselines for both question answering and prediction tasks, validating its robustness and adaptability in handling complex table-related tasks.

2 Related Works

2.1 Multi-Modal Tabular Understanding

Multimodal tabular understanding (Jiang et al., 2025b) aims to parse table structures from document images or scanned copies and to perform deep semantic reasoning by jointly leveraging textual context and table-associated visual information (Gu et al., 2022; Kim et al., 2021). It mainly involves two categories of tasks: multimodal table question answering (Jiang et al.; Peng et al., 2023) and multimodal table prediction (Ye et al., 2024; Jiang et al., 2025a; Hollmann et al., 2025; Howard et al., 2018; Zhou et al., 2025a; Cheng et al., 2025). For multimodal table question answering, many existing studies (Zhuang et al., 2024; Zhang et al., 2025a; Tian et al., 2025a) enhance reasoning capability by constructing multiple reasoning paths or introducing chain-of-thought style mechanisms (Wei et al., 2022; Lei et al., 2025). In contrast, methods for multimodal table prediction (Jiang et al., 2024), such as TabLLM (Hegselmann et al., 2023), are largely designed for unimodal tabular or textual inputs and thus cannot effectively handle multimodal and multi-task scenarios (Du et al., 2024; Luo et al., 2025; Tian et al., 2024). Moreover, research that unifies multimodal table question answering and prediction within a single framework remains relatively limited.

2.2 Neuro-Symbolic Reasoning

Neuro-symbolic reasoning (Yang et al., 2025; Manhaeve et al., 2018) aims to combine traditional symbolic reasoning with learning-based mechanisms to improve the reliability of intelligent systems (Wang et al., 2019). In the era of large language models, reasoning is mainly enhanced through three symbolic paradigms: **symbolic solver–aided methods** (Pan et al., 2023) that formalize natural language and delegate reasoning to external solvers, **program-aided** (Gao et al., 2023) methods that execute generated programs via interpreters, and **tool-aided methods** (Zhang et al., 2025b) that coordinate multiple external tools for flexible task-specific reasoning via SFT and RL. Existing

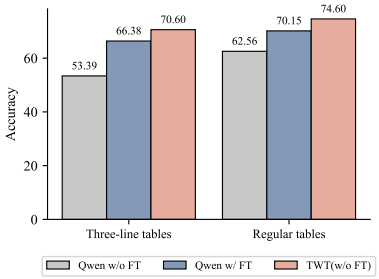


Figure 1: Performance on TAT-QA dataset with different situations. FT means a full table message is provided.

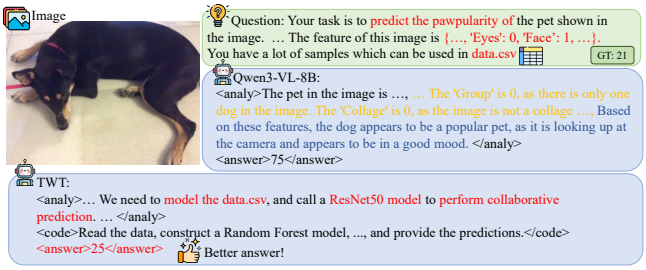


Figure 2: Predictions from different models. Yellow highlights indicate insufficient feature modeling by existing methods, while red highlights show that our model performs more comprehensive feature modeling through code-based interactions.

neuro-symbolic systems have demonstrated notable success in domains such as mathematics (Tian et al., 2025b; Hao et al., 2025; Zhou et al., 2025b), coding (Sapkota et al., 2025), and planning (Shao et al., 2024). Owing to the structured nature of tabular data, neuro-symbolic reasoning is particularly well-suited for table question answering and understanding, enabling more accurate, robust, and interpretable table reasoning.

3 Problem and Analysis

3.1 Problem Definition

Tabular-Vision Multi-Modal Understanding (TVMU) aims to parse table structures from document images or scanned copies and perform deep reasoning by jointly leveraging textual context and table-associated visual information. It primarily encompasses two core tasks: multimodal table question answering and multimodal table prediction. In this task, models do not have access to the complete table in the query. Instead, they are required to extract table structures from the visual modality, interpret table-associated images, and perform operations such as information extraction, mathematical computation, and table modeling to answer user queries.

In the TVMU task, the MLLM $\mathcal{M}$ is provided with only the table headers and question-relevant images as visual inputs V, while the accompanying textual context of the table and the user query Q are given as textual inputs. Meanwhile, an interactive sandbox environment is introduced as the external environment E, which contains the raw tabular data files, image files, and potentially pre-trained models that may be required during reasoning. We expect the model to perform multimodal visual understanding, table modeling, and related reasoning through iterative interactions with the external environment, and ultimately generate the final answer. The generated answer is expected to be highly consistent with the ground-truth answer y_i, thereby improving accuracy and reliability in real-world multimodal tabular reasoning scenarios.

$$\min_{\mathcal{M}} \sum_{i=1}^{N} \ell\big(\mathcal{M}(V_i, Q_i, E_i), y_i\big) \tag{1}$$

where $\ell(\cdot)$ denotes a loss function that measures the discrepancy between the model prediction and the ground-truth answer, and $\mathcal{M}(V_i, Q_i, E_i)$ represents the prediction generated by the model given the visual input V_i, the query Q_i, and the external environment E_i for the i-th sample. Our objective is to guide the model to effectively leverage visual information and environment interaction capabilities to accomplish table parsing, key information localization, and necessary computational reasoning.

3.2 Problem Analysis

Challenge 1: Structural variability and data incompleteness in real applications. Tables can take highly varied forms, such as three-line tables, merged-row or merged-column tables, and tables embedded within document pages rather than presented independently (Zhu et al., 2021; Chang et al.,

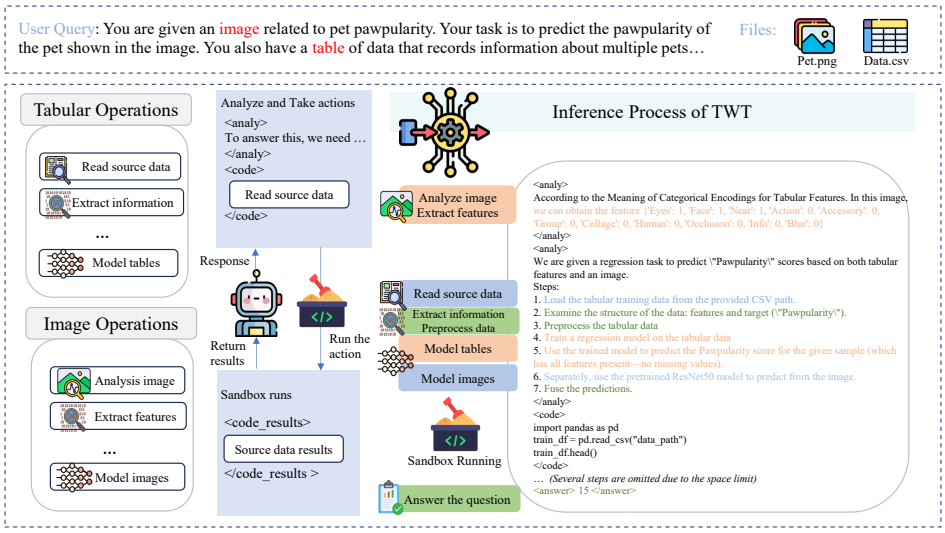


Figure 3: TWT framework. It realizes robust and good performance via code-base neuro-symbolic reasoning in TVMU tasks.

2008). Moreover, in scenarios involving scanned documents, low-resolution screenshots, or tables affected by occlusion or layout degradation, numerical and textual information may be missing or incorrectly recognized, substantially increasing the difficulty of table parsing and comprehension. To quantitatively assess the impact of these issues on model performance, we conducted an experimental evaluation using Qwen3-VL-8B (Team, 2025) as a representative model. As shown in 1, the model's reasoning performance significantly declines when the full table (FT) is not accessible (w/o FT vs w/ FT). Additionally, performance also deteriorates in more structurally complex table formats, such as three-line tables. These observations indicate that relying solely on visual inputs for table understanding exhibits notable limitations in complex and incomplete scenarios.

Challenge 2: Implicit and complex feature dependencies. Current language models struggle to identify the feature dependencies that are relevant for prediction tasks. In TVMU tasks, models typically need to capture the dependencies between features and the prediction targets. This process involves not only the direct influence of individual cells or attributes on the target but also the interrelationships among features. For example, in a table prediction task concerning pet popularity (Howard et al., 2021) (2 using Qwen3-VL-8B (Team, 2025) as an example), the model may analyze only the pet images and the corresponding features in the table. However, it fails to recognize the specific contributions of each feature to popularity and their interdependencies. This limitation results in significantly degraded performance.

Challenge 3: The significant heterogeneity of task requirements. TVMU tasks exhibit significant heterogeneity across different task settings, leading to substantially different problem-solving pipelines. Tasks differ in their reliance on information sources and reasoning processes. For example, question answering often requires integrating tabular, textual, and visual information, whereas classification tasks primarily depend on the internal structure and explicit features of tables. In addition, variations in data preprocessing further exacerbate this heterogeneity. Moreover, task heterogeneity poses additional challenges to model generalization: models must not only capture complex feature dependencies within and across tables, but also adapt to diverse reasoning logics and information integration strategies. Such multi-level differences complicate the design of unified models and make it difficult for a existing approach to maintain robust performance across diverse tasks.

4 METHODS

Inspired by the challenge above, we propose TWT, a **program-aided neuro-symbolic reasoning framework** for TVMU tasks. We design a two-stage training strategy for TWT to address the three

challenges in TVMU tasks. In the first stage, Task-Oriented Supervised Fine-Tuning guides the model to master the processing pipelines of different table understanding tasks, thereby mitigating task heterogeneity, while enabling the model to initially learn how to parse complex table formats and model feature dependencies. In the second stage, Adaptive Loss-Scaled Reinforcement Learning is introduced to further enhance the model's ability to handle diverse table structures and complex feature dependencies, as well as to improve the correctness and stability of its outputs. As shown in 3, under this training framework, TWT integrates code execution with neuro-symbolic reasoning, enabling explicit step-by-step planning and execution, while supporting flexible processing of multimodal table data and efficient information extraction, computation, and reasoning.

4.1 Environment Setup

Interaction Pattern. To facilitate systematic problem analysis, interaction with external environments, acquisition of execution feedback, and generation of final answers, we design a unified and strictly constrained interaction protocol, which the model is required to follow for output. The protocol consists of the following components:

- `<analy></analy>`: explicitly represents the model's reasoning process for the given problem;
- `<code></code>`: specifies the code to be executed within a sandboxed environment;
- `<code_results></code_results>`: returns the execution results or error messages produced by the sandbox;
- `<answer></answer>`: outputs the final answer.

In practical applications, this interaction protocol enables accurate extraction of the model's intent via simple regular expression matching and allows safe execution of generated code in a controlled sandbox environment, thereby closing the loop between reasoning, execution, and feedback and improving the system's usability and reliability.

Sandbox Environment. To ensure the efficient and safe interaction with files in the environment via code, we construct a controlled sandbox environment designed to execute model-generated code. This environment imposes strict constraints on code execution and provides the model with complete and accurate feedback on both outputs and any errors encountered during execution.

In practice, we observe that insufficiently trained models may generate faulty code, such as scripts containing infinite loops or unexpected behavior. To mitigate this, the sandbox environment executes code within a Python REPL equipped with a timeout mechanism, preventing unbounded execution. The sandbox pre-integrates all task-relevant resources. Within this environment, the model can flexibly access and manipulate these resources according to specified paths, with execution results returned for subsequent reasoning and decision-making.

4.2 Task-Oriented Supervised Fine-Tuning

As discussed earlier, there exists a significant challenge of heterogeneity between different task types in TVMU tasks. Therefore, to construct a unified task-solving framework, the model must not only comprehend the table content itself but also possess the capability to adapt to and execute the pipelines corresponding to different tasks.

To achieve this, we first perform Task-Oriented Supervised Fine-Tuning (TO-SFT) to preliminarily train the model. This stage guides the model to effectively internalize the solution paradigms of different tasks as part of the model's capability.

Data Synthesis Process. We first need to construct a set of high-quality training data that explicitly contains problem-solving trajectories. Existing TVMU datasets, such as WikiTQ (Pasupat & Liang, 2015) and Adoption (Howard et al., 2018), while large in scale, typically provide only final answers and lack corresponding reasoning process annotations. This limitation restricts the model's ability to learn and generalize reasoning skills. To address this, we design a verifiable data synthesis pipeline to automatically construct training samples with reliable reasoning traces, as follows.

Specifically, for the i-th sample, we submit its visual input (V_i), problem description (Q_i), and external environment (E_i) to a more capable multimodal large language model (Qwen3-VL-Plus (Team, 2025))

according to a predefined input format and problem-solving prompts.

$$x_i = (V_i, Q_i, E_i)$$

$$(\hat{y}_i, \hat{r}_i) = \mathcal{M}(x_i)$$

During this process, the model is required to explicitly output both its reasoning trajectory $\hat{r}_i$ and the final answer $\hat{y}_i$. The generated answers are then compared with the ground-truth annotations. For regression answers, a relative error of up to 10% is tolerated; samples satisfying this criterion are accepted, while those exceeding it are rejected.

$$\mathcal{A}(\hat{y}_i, y_i) = \begin{cases} \mathbb{I}\left(\dfrac{|\hat{y}_i - y_i|}{\max(|y_i|, \epsilon)} \leq \delta\right), & i \in \mathcal{R}, \\ \mathbb{I}(\hat{y}_i = y_i), & i \in \mathcal{C}, \end{cases}$$

$$\mathcal{D}_{\text{accept}} = \left\{ (x_i, \hat{r}_i, \hat{y}_i) \;\middle|\; \mathcal{A}\big(\hat{y}_i, y_i\big) = 1 \right\}.$$

where $\mathcal{R}$ denotes the set of indices for the regression task in table prediction, and $\mathcal{C}$ denotes the set of indices for others.

However, in practical applications, even state-of-the-art MLLMs exhibit notable limitations when performing advanced operations such as table modeling and precise numerical computation. Given that large language models (LLMs) (Guo et al., 2025) demonstrate more stable performance in instruction following and complex operations, we adopt a data synthesis strategy that combines MLLMs and LLMs. Specifically, the MLLM is primarily responsible for semantically understanding the input images and generating intermediate steps and structured representations for problem-solving. These outputs, along with subsequent reasoning steps, are then provided as input to the LLM, which performs more refined reasoning and computation to produce the final answer.

Training Process. The core objectives of this training process include internalizing problem-solving pipelines corresponding to different task types, which reduces the heterogeneity influence, and learning how to interpret and utilize results or error messages returned from code execution, thereby supporting reasoning based on execution feedback, which is helpful to process diverse table structures, incomplete data, and complex feature dependencies.

To achieve this, during the TO-SFT stage, we adopt a single-turn dialogue training setup. Specifically, for each sample in the synthesized dataset, the multi-turn reasoning path r are concatenated to form a complete training sequence.

During training, to prevent the model from directly memorizing code execution results during the TO-SFT stage (Zhang et al., 2025a;b), we mask the outputs of code execution, computing the loss only over the model-generated text and code sequences. Accordingly, the TO-SFT stage optimization objective is defined as:

$$\mathcal{L}_{\text{TO-SFT}} = -\mathbb{E}_{(x_i, y_i) \sim \mathcal{D}_{\text{accept}}} \sum_{t \in \mathcal{T}_{\text{gen}}} \log p_\theta\big(y_{i,t} \mid x_i, y_{i,<t}\big) \tag{2}$$

where $\mathcal{T}_{\text{gen}}$ denotes the set of generated tokens that are not masked.

4.3 Adaptive Loss-Scaled Reinforcement Learning

After completing TO-SFT training, the model has largely mastered the prescribed output conventions and basic problem-solving procedures. To further enhance the model's ability to process TVMU tasks, we apply Adaptive Loss-Scaled Group Relative Policy Optimization (AL-GRPO) for reinforcement learning. Unlike supervised learning, AL-GRPO optimizes the model via policy gradients derived from reward signals, minimizing token-level prediction errors while encouraging the model to explore a broader solution space. This facilitates learning more diverse and complex reasoning behaviors.

In prior work (Guo et al., 2025), GRPO reward design often emphasizes the format correctness of model outputs. However, we observe that after high-quality TO-SFT training, the model can reliably generate responses according to predefined formats, making format constraints no longer a primary bottleneck. Based on this observation, during the reinforcement learning stage, we focus the reward

Table 1: Comparison of different models on multimodal table question answering and multimodal table prediction benchmarks. Best results are shown in **bold**. The second-best results are underlined.

Models	Multimodal Table QA				Multimodal Table Prediction				AVG_rank
	TabMWP ACC ↑	WikiTQ ACC ↑	FinQA ACC ↑	TAT-QA ACC ↑	Adoption ACC ↑	SkinCA ACC ↑	Pawpularity nRMSE ↓	Paintings nRMSE ↓	-
API-based MLLMs									
GPT-5.1	92.50	40.82	47.68	56.38	<u>26.12</u>	<u>65.51</u>	<u>0.366</u>	<u>0.098</u>	3.63
Qwen3VL-Plus	**98.70**	<u>76.10</u>	**64.41**	<u>72.39</u>	18.67	51.55	0.539	0.178	4.25
General Open-Source MLLMs									
Qwen3VL-8B	86.70	50.64	38.24	62.53	25.37	40.38	0.501	0.172	5.63
Qwen3VL-30B	88.10	58.35	55.76	59.17	18.74	44.67	0.493	0.130	5.00
InternVL3.5-8B	82.80	47.97	42.21	54.87	19.41	43.20	0.466	0.119	5.75
InternVL3.5-38B	90.10	67.84	57.77	60.37	20.52	43.47	0.495	0.110	4.38
Tabular Related Open-Source MLLMs									
HIPPO	79.90	30.68	7.86	18.69	23.68	29.76	0.485	0.290	7.25
Table-LLAVA	30.15	9.16	4.50	11.75	23.19	30.28	0.494	0.282	7.88
TWT (Ours)	<u>97.10</u>	**78.95**	<u>61.95</u>	**72.82**	**32.90**	**73.65**	**0.199**	**0.093**	**1.25**

function on the correctness of the final answer, which can be formalized as follows:

$$\text{Reward}(\hat{y}_i, y_i) = \begin{cases} 1 - \min\left(\dfrac{|\hat{y}_i - y_i|}{|y_i| + \epsilon},\ 1\right), & i \in \mathcal{R}, \\ \mathbb{I}(\hat{y}_i = y_i), & i \in \mathcal{C}. \end{cases}$$

This reward formulation combines both binary (0/1) signals and continuous-valued feedback, effectively alleviating the issue of reward sparsity.

While prioritizing the final answer correctness, we also aim for the model to generate executable and stable code, reducing invalid interactions with the external environment. Directly using code correctness as a reward is challenging: (1) code complexity varies significantly across samples, and (2) rewarding only based on successful execution leads to extremely sparse rewards for code involving complex modeling or multi-step computation, which destabilizes training.

To address this, we introduce an adaptive loss-scaled strategy to determine which code generation steps contribute to the AL-GRPO loss. Only code segments that terminate successfully are included in the AL-GRPO loss computation, while segments that fail or terminate abnormally receive no gradient updates. By excluding code-related information from the reward, this approach both prevents reward hacking and allows the model to gradually favor executable and stable code without requiring complex code-level reward definitions.

For each question x_i in the training dataset $\mathcal{D}$, AL-GRPO samples a group of G multi-turn outputs $\{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_G\}$ from the old policy π_{old}. We obtain $\hat{\boldsymbol{o}}$, which satisfies.

$$\hat{\boldsymbol{o}}_i = (o_i^1, o_i^2, \ldots, o_i^{n_i}), \quad \text{where } S(o_i^k) = 1\ \forall k.$$

Here $S(\cdot)$ is the sandbox output about whether the code exited normally. The policy model π_θ is then optimized by maximizing the following objective:

$$\mathcal{L}_{\text{AL-GRPO}} = \mathbb{E}_{q\sim\mathcal{D},\, \{\hat{\boldsymbol{o}}_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \left[\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(\hat{\boldsymbol{o}}_i \mid q)}{\pi_{\theta_{\text{old}}}(\hat{\boldsymbol{o}}_i \mid q)}A_i,\ \text{clip}\Big(\frac{\pi_\theta(\hat{\boldsymbol{o}}_i \mid q)}{\pi_{\theta_{\text{old}}}(\hat{\boldsymbol{o}}_i \mid q)}, 1-\epsilon,\ 1+\epsilon\Big)A_i\right) - \beta\, D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}})\right)\right] \tag{3}$$

where A_i is the advantage calculated from the rewards $\{r_1, \ldots, r_G\}$ of outputs within each group, π_{ref} is the reference model, and ϵ and β are hyperparameters.

5 Experiments

5.1 Experimental Setup

We adopt Qwen3-VL-8B (Team, 2025) as the backbone of TWT and construct approximately 1.5K QA training samples using Qwen3VL-Plus and Qwen3-Max, alongside 1.2K prediction samples from

Table 2: Experimental Results of Qwen3-VL-8B under TO-SFT and AL-GRPO on Table QA and Table Prediction Benchmarks. Best is shown in **bold**

Method	Table QA		Table Prediction	
	WikiTQ ACC ↑	TAT-QA ACC ↑	SkinCA ACC ↑	Paintings nRMSE ↓
Qwen3VL-8B	50.64	62.53	40.38	0.172
+TO-SFT	78.38	71.31	71.08	0.099
+GRPO	78.39	71.37	70.92	0.101
+AL-GRPO	**78.95**	**72.82**	**73.65**	**0.093**

Table 3: Experiment results of TWT under situation w and w/o table features in multimodal table prediction tasks

Method	Adoption ACC ↑	Pawpularity nRMSE ↓
Qwen3VL-8B w features	25.37	0.501
TWT w features	32.90	0.199
Δ	+7.53	-0.302
TWT w/o features	31.40	0.200
Δ	+6.03	-0.303

four datasets. TO-SFT runs for three epochs on $4\times$A800 GPUs with a learning rate of 10^{-5}, followed by AL-GRPO for two epochs on $8\times$A800 GPUs using 0.5K QA and 0.4K prediction samples with a learning rate of 5×10^{-7}. We evaluate QA on WikiTQ (Pasupat & Liang, 2015), TabMWP (Lu et al., 2022), FinQA (Chen et al., 2021), and TAT-QA (Zhu et al., 2021); classification on Adoption (Howard et al., 2018) and SkinCA[1]; and regression on Pawpularity (Howard et al., 2021) and Paintings[2]. We report accuracy for QA and classification, and normalized root mean squared error (nRMSE) for regression. During evaluation, visual inputs contain only table headers, requiring models to retrieve full table data through code-based interaction with the external environment. Baselines include Qwen3-VL-8B/30B (Team, 2025), InternVL-3.5-8B/38B (Wang et al., 2025), HIPPO (Liu et al., 2025), and Table-LLaVA (Zheng et al., 2024), with GPT-5.1 and Qwen3-VL-Plus as proprietary reference models. All baselines receive identical sandbox instructions and resource access.

5.2 Main Results

TWT achieves the best overall performance across eight TVMU datasets, consistently outperforming open-source baselines (Table 1). On QA tasks, it surpasses the strongest baseline by nearly 10% and performs comparably to API-based models. We attribute these gains to training that enables robust understanding and information extraction across diverse table structures and incomplete data, where baselines such as HIPPO (Liu et al., 2025) exhibit limited robustness. On table prediction tasks, TWT improves performance by capturing complex feature–target dependencies through code execution and leveraging pretrained image models. These results support the effectiveness of the two-stage training strategy in internalizing task-solving pipelines and addressing the three TVMU challenges. Further analysis of TWT's contributions beyond distillation is provided in Appendix B.

5.3 Further Analysis

Ablation Study Ablation experiments on four datasets (Table 2) demonstrate the complementary roles of TO-SFT and AL-GRPO. TO-SFT establishes foundational task-solving and environment-interaction capabilities, improving WikiTQ accuracy from 50.64 to 78.38. Building on this strong baseline, AL-GRPO further improves performance on challenging tasks, including SkinCA (71.08→73.65) and TAT-QA (71.31→72.82). Beyond accuracy gains, it improves code execution stability (Appendix C) and substantially reduces reasoning non-convergence, defined as failure to obtain an answer after ten sandbox interactions (Fig. 4). Together, these results show that TO-SFT builds basic task-solving competence, while AL-GRPO enhances reasoning quality and execution reliability.

Performance on Situations without tabular features. To further evaluate TWT's performance under incomplete information, we deliberately removed the table content from the sample images in the multimodal table prediction. When the table features in the images are missing, TWT cannot directly rely on complete table modeling and must instead extract relevant features from the images to accomplish the task. As shown in Table 3, even under incomplete information, TWT maintains competitive performance. This demonstrates that, in multimodal table prediction scenarios, TWT can effectively align table and visual information based on image understanding, handle missing values, and achieve robust predictions.

[1] https://www.kaggle.com/datasets/mahdavi1202/skin-cancer

[2] https://www.kaggle.com/datasets/denozavrus/paintings-price-prediction/data

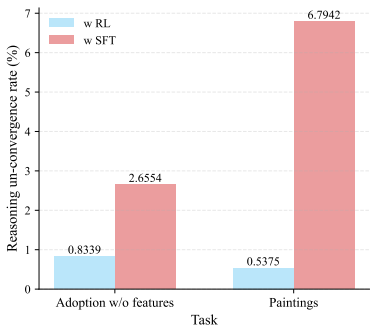


Figure 4: Comparison of reasoning convergence rate.

Table 4: Comparison to tabular-only and image-only baselines on prediction tasks. Best results are shown in **bold**. The second-best results are underlined.

Type	Models	SkinCA	Pawpularity
		ACC ↑	nRMSE ↓
Tabular-only	**Random Forest**	**74.78**	0.214
Image-only	**ResNet50**	70.00	<u>0.202</u>
Multi-modal	**Qwen3VL-8B**	40.38	0.501
	TIP (Du et al., 2024)	71.50	0.216
	Codex+GPT5.5	68.91	0.213
	TWT (Ours)	<u>73.65</u>	**0.199**

Additional Metrics on Prediction Tasks. To provide a more comprehensive evaluation beyond Accuracy and nRMSE, we additionally report Balanced Accuracy for classification and Spearman rank correlation (ρ) for regression in Table 5. TWT achieves the best Balanced Accuracy on Adoption and the highest Spearman ρ on Pawpularity, while remaining competitive with GPT-5.1 on SkinCA and Paintings. These results show that TWT's strong performance in the main evaluation is consistent across different metrics, rather than being tied to a specific evaluation criterion.

Table 5: Additional metrics on prediction tasks. Best are in **bold**. The second-best are underlined.

Models	Classification		Regression	
	Adoption	SkinCA	Pawpularity	Paintings
	Bal. ACC ↑	Bal. ACC ↑	Spearman ρ ↑	Spearman ρ ↑
API-based MLLMs				
GPT-5.1	<u>21.95</u>	**65.96**	<u>0.142</u>	**0.486**
Qwen3VL-Plus	20.42	57.54	0.008	0.413
TWT (Ours)	**25.49**	<u>63.69</u>	**0.316**	<u>0.445</u>

Comparison to Tabular/Image baselines TWT enables code-based tabular reasoning from incomplete visual inputs without predefined operation sets. On WTQ, it achieves 78.95% accuracy, surpassing RePanda (Chegini et al., 2025), Chain-of-Table (Wang et al., 2024), and TURBO (Jiang et al., 2025b) (75.1%, 67.3%, and 67.80%, respectively), despite their access to complete table structures. For prediction, Table 4 compares TWT with a Random Forest trained on raw CSV data and an image-only ResNet50. These baselines perform well on SkinCA and Pawpularity, respectively, but poorly on the other task. TWT achieves the best Pawpularity nRMSE and SkinCA accuracy within 1.2 percentage points of the tabular-only baseline, suggesting effective use of both modalities. Qwen3-VL-8B performs substantially worse on both tasks with identical sandbox access, indicating that TWT's gains stem from its training framework rather than tool access alone.

6 Conclusion

In this paper, we study the problem of TVMU tasks and, through an in-depth analysis, identify three fundamental challenges inherent to this task. To address these challenges, we propose TWT, **a program-aided neuro-symbolic reasoning** framework that equips models with code-based reasoning capabilities through a two-stage training strategy. By learning to plan and execute key problem-solving steps via effective interaction with external code execution environments, TWT enables reliable reasoning over multimodal tabular data. Experiments on eight datasets demonstrate the effectiveness of our approach. One limitation of our work is that we have not evaluated TWT on general benchmarks such as MMLU and GSM8K. However, catastrophic forgetting is likely limited by the small training scale (<7K samples), the frozen visual encoder, and the inclusion of about 3K math and code reasoning samples. A more systematic evaluation is left for future work.

AI USE STATEMENT

Generative AI tools were used to support English translation, language refinement, and the presentation of figures and tables. This included improving phrasing and clarity, as well as revising captions and layouts. The authors remain fully responsible for the paper's technical content, results, citations, and conclusions, and for the accuracy and integrity of the final manuscript.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Iñigo Alonso, Imanol Miranda, Eneko Agirre, and Mirella Lapata. Tablet: A large-scale dataset for robust visual table understanding. *arXiv preprint arXiv:2509.21205*, 2025.

Naomi Altman and Martin Krzywinski. Tabular data. *Nature Methods*, 14(4):329–330, 2017.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)*, 26(2):1–26, 2008.

Atoosa Chegini, Keivan Rezaei, Hamid Eghbalzadeh, and Soheil Feizi. Repanda: pandas-powered tabular verification and reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32200–32212, 2025.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, et al. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, 2021.

Zi-Jian Cheng, Zi-Yi Jia, Kun-Yang Yu, Zhi Zhou, and Lan-Zhe Guo. Fully test-time adaptation for feature decrement in tabular data. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 4940–4948, 2025.

Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, et al. Opportunities and obstacles for deep learning in biology and medicine. *Journal of the Royal Society Interface*, 15(141):20170387, 2018.

Long Hoang Dang, Thao Minh Le, Vuong Le, Tu Minh Phuong, and Truyen Tran. Sadl: An effective in-context learning method for compositional visual qa. *arXiv preprint arXiv:2407.01983*, 2024.

Siyi Du, Shaoming Zheng, Yinsong Wang, Wenjia Bai, Declan P O'Regan, and Chen Qin. Tip: Tabular-image pre-training for multimodal classification with incomplete data. In *European Conference on Computer Vision*, pp. 478–496. Springer, 2024.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pp. 10764–10799. PMLR, 2023.

Zhangxuan Gu, Changhua Meng, Ke Wang, Jun Lan, Weiqiang Wang, Ming Gu, and Liqing Zhang. Xylayoutlm: Towards layout-aware multimodal networks for visually-rich document understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4583–4592, 2022.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yilun Hao, Yongchao Chen, Yang Zhang, and Chuchu Fan. Large language models can solve real-world planning rigorously with formal verification tools. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3434–3483, 2025.

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*, pp. 5549–5581. PMLR, 2023.

Daniel Hein, Stefan Depeweg, Michel Tokic, Steffen Udluft, Alexander Hentschel, Thomas A. Runkler, and Volkmar Sterzing. A benchmark environment motivated by industrial control problems. In *Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence*, pp. 1–8, 2017.

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

Addison Howard, MichaelApers, and Mongrel Jedi. Petfinder.my adoption prediction, 2018. Kaggle.

Addison Howard, Mongrel Jedi, and Ryan Holbrook. Petfinder.my - pawpularity contest, 2021. Kaggle.

Jun-Peng Jiang, Tao Zhou, De-Chuan Zhan, and Han-Jia Ye. Compositional condition question answering in tabular understanding. In *Forty-second International Conference on Machine Learning*.

Jun-Peng Jiang, Han-Jia Ye, Leye Wang, Yang Yang, Yuan Jiang, and De-Chuan Zhan. Tabular insights, visual impacts: transferring expertise from tables to images. In *Forty-first International Conference on Machine Learning*, 2024.

Jun-Peng Jiang, Si-Yang Liu, Hao-Run Cai, Qile Zhou, and Han-Jia Ye. Representation learning for tabular data: A comprehensive survey. *arXiv preprint arXiv:2504.16109*, 2025a.

Jun-Peng Jiang, Yu Xia, Hai-Long Sun, Shiyin Lu, Qing-Guo Chen, Weihua Luo, Kaifu Zhang, De-Chuan Zhan, and Han-Jia Ye. Multimodal tabular reasoning with privileged structured information. *arXiv preprint arXiv:2506.04088*, 2025b.

Geewook Kim, Teakgyu Hong, Moonbin Yim, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Donut: Document understanding transformer without ocr. *arXiv preprint arXiv:2111.15664*, 7(15):2, 2021.

Fangyu Lei, Jinxiang Meng, Yiming Huang, Tinghong Chen, Yun Zhang, Shizhu He, Jun Zhao, and Kang Liu. Reasoning-table: Exploring reinforcement learning for table reasoning. *arXiv preprint arXiv:2506.01710*, 2025.

Zhenghao Liu, Haolan Wang, Xinze Li, Qiushi Xiong, Xiaocui Yang, Yu Gu, Yukun Yan, Qi Shi, Fangfang Li, Ge Yu, et al. Hippo: Enhancing the table understanding capability of large language models through hybrid-modal preference optimization. *arXiv preprint arXiv:2502.17315*, 2025.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610*, 2022.

Jiaqi Luo, Yuan Yuan, and Shixin Xu. Time: Tabpfn-integrated multimodal engine for robust tabular-image learning. *arXiv preprint arXiv:2506.00813*, 2025.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *Advances in neural information processing systems*, 31, 2018.

Jaehyun Nam, Kyuyoung Kim, Seunghyuk Oh, Jihoon Tack, Jaehyung Kim, and Jinwoo Shin. Optimized feature generation for tabular data via llms with decision tree reasoning. *Advances in Neural Information Processing Systems*, 37:92352–92380, 2024.

Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer. Deep learning for financial applications: A survey. *Applied Soft Computing*, 93:106384, 2020.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 3806–3824, 2023.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*, 2015.

Li Peng, He Yeye, Yashar Dror, Cui Weiwei, Ge Song, Zhang Haidong, Fainman Danielle Rifinski, Zhang Dongmei, and Chaudhuri Surajit. Table-gpt: Table-tuned gpt for diverse table tasks. *arXiv (Cornell University)*, 2023.

Yingzhe Peng, Gongrui Zhang, Miaosen Zhang, Zhiyuan You, Jie Liu, Qipeng Zhu, Kai Yang, Xingzhong Xu, Xin Geng, and Xu Yang. Lmm-r1: Empowering 3b lmms with strong reasoning abilities through two-stage rule-based rl. *arXiv preprint arXiv:2503.07536*, 2025.

Arash Salehpour and Karim Samadzamini. A bibliometric analysis on the application of deep learning in economics, econometrics, and finance. *International Journal of Computer Sciences and Engineering*, 27(2):167–181, 2024.

Ranjan Sapkota, Konstantinos I Roumeliotis, and Manoj Karkee. Vibe coding vs. agentic coding: Fundamentals and practical implications of agentic ai. *arXiv preprint arXiv:2505.19443*, 2025.

Jie-Jing Shao, Bo-Wen Zhang, Xiao-Wen Yang, Baizhi Chen, Si-Yu Han, Wen-Da Wei, Guohao Cai, Zhenhua Dong, Lan-Zhe Guo, and Yu-feng Li. Chinatravel: An open-ended benchmark for language agents in chinese travel planning. *arXiv preprint arXiv:2412.13682*, 2024.

Wei Shen, Jiangbo Pei, Yi Peng, Xuchen Song, Yang Liu, Jian Peng, Haofeng Sun, Yunzhuo Hao, Peiyu Wang, Jianhao Zhang, et al. Skywork-r1v3 technical report. *arXiv preprint arXiv:2507.06167*, 2025.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown. *Hugging Face*, 2024.

Qwen Team. Qwen3 technical report, 2025.

Shi-Yu Tian, Zhi Zhou, Wei Dong, Kun-Yang Yu, Ming Yang, Zi-Jian Cheng, Lan-Zhe Guo, and Yu-Feng Li. Tabularmath: Understanding math reasoning over tables with large language models. *arXiv preprint arXiv:2505.19563*, 2025a.

Shi-Yu Tian, Zhi Zhou, Kun-Yang Yu, Ming Yang, Lin-Han Jia, Lan-Zhe Guo, and Yu-Feng Li. Vcsearch: Bridging the gap between well-defined and ill-defined problems in mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 12721–12742, 2025b.

Shiyu Tian, Hongxin Wei, Yiqun Wang, and Lei Feng. Crosel: Cross selection of confident pseudo labels for partial-label learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19479–19488, 2024.

Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning*, pp. 6545–6554. PMLR, 2019.

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

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

June Yong Yang, Geondo Park, Joowon Kim, Hyeongwon Jang, and Eunho Yang. Language-interfaced tabular oversampling via progressive imputation and self-authentication. In *The Twelfth International Conference on Learning Representations*, 2024.

Xiao-Wen Yang, Jie-Jing Shao, Lan-Zhe Guo, Bo-Wen Zhang, Zhi Zhou, Lin-Han Jia, Wang-Zhou Dai, and Yu-Feng Li. Neuro-symbolic artificial intelligence: Towards improving the reasoning abilities of large language models. *arXiv preprint arXiv:2508.13678*, 2025.

Han-Jia Ye, Huai-Hong Yin, De-Chuan Zhan, and Wei-Lun Chao. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. *arXiv preprint arXiv:2407.03257*, 2024.

Shaolei Zhang, Ju Fan, Meihao Fan, Guoliang Li, and Xiaoyong Du. Deepanalyze: Agentic large language models for autonomous data science. *arXiv preprint arXiv:2510.16872*, 2025a.

Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, et al. Thyme: Think beyond images. *arXiv preprint arXiv:2508.11630*, 2025b.

Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. Multimodal table understanding. *arXiv preprint arXiv:2406.08100*, 2024.

Zhi Zhou, Kun-Yang Yu, Lan-Zhe Guo, and Yu-Feng Li. Fully test-time adaptation for tabular data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23027–23035, 2025a.

Zhi Zhou, Kun-Yang Yu, Shi-Yu Tian, Xiao-Wen Yang, Jiang-Xin Shi, Pengxiao Song, Yi-Xuan Jin, Lan-Zhe Guo, and Yu-Feng Li. Lawgpt: Knowledge-guided data generation and its application to legal llm. *arXiv preprint arXiv:2502.06572*, 2025b.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. *arXiv preprint arXiv:2105.07624*, 2021.

Alex Zhuang, Ge Zhang, Tianyu Zheng, Xinrun Du, Junjie Wang, Weiming Ren, Stephen W Huang, Jie Fu, Xiang Yue, and Wenhu Chen. Structlm: Towards building generalist models for structured knowledge grounding. *arXiv preprint arXiv:2402.16671*, 2024.

A Detailed Experiment Setup

A.1 Dataset Details

We use eight datasets in our experiments: four multimodal table question-answering datasets and four multimodal table-prediction datasets.

Implementation Details. We adopt Qwen3-VL-8B (Team, 2025) as the base model for TWT. During the data synthesis stage, approximately 1.5K high-quality multimodal question–answering training samples are generated with the assistance of Qwen3VL-Plus and Qwen3-Max (Team, 2025). For multimodal table prediction tasks, we collect data from four datasets, resulting in a total of 1.2K synthesized training samples. In the TO-SFT stage, the model is trained for three epochs on a 4× A800 server with a learning rate of 1e-5. In the AL-GRPO stage, we select 0.5K QA samples and 0.4K prediction samples from the synthesized data to perform training on an 8×A800 server for two epochs, using a learning rate of 5e-7.

Evaluation Datasets and Metrics. We evaluate model performance primarily on two types of tasks: multimodal table question answering and table prediction. For QA tasks, we adopt four benchmark datasets: WikiTQ (Pasupat & Liang, 2015), TabMWP (Lu et al., 2022), FinQA (Chen et al., 2021), and TAT-QA (Zhu et al., 2021). Among them, WTQ and TabMWP are standard table question answering datasets, while FinQA and TAT-QA additionally include table-related contextual text for each question. For table prediction, we evaluate on four multimodal tabular datasets: Adoption (Howard et al., 2018) and SkinCA[3] for classification tasks, and Pawpularity (Howard et al., 2021) and Paintings[4] for regression tasks. In terms of evaluation metrics, we use accuracy (ACC) for question answering and classification tasks, and normalized Root Mean Squared Error (nRMSE) for regression tasks. Notably, during evaluation, we do not directly provide the model with complete table contents. Instead, only the table headers are included in the visual input, and the full table data must be retrieved by the model through code-based interaction with the external environment. This evaluation setting more closely reflects real-world application scenarios and enables a more realistic assessment of the model's capabilities in complex multimodal table understanding tasks.

Comparison Methods. In our comparative experiments, we select a diverse set of representative models covering both mainstream open-source multimodal large language models and tabular understanding approaches. Specifically, the evaluated open-source models include Qwen3-VL-8B / 30B (Team, 2025), InternVL-3.5-8B / 38B (Wang et al., 2025), HIPPO (Liu et al., 2025), and Table-LLaVA (Zheng et al., 2024). In addition, we include two proprietary API-based models as reference upper bounds: GPT-5.1 and Qwen3-VL-Plus. All baselines are provided with identical sandbox instructions and guaranteed access to their resources.

Multimodal table question answering.

1. **WikiTQ** (Pasupat & Liang, 2015) is a table-based question-answering dataset focused on general knowledge and compositional reasoning.
2. **TabMWP** (Lu et al., 2022) is a collection of table-based mathematical word problems that require numerical calculation and reasoning over tables.
3. **FinQA** (Chen et al., 2021) evaluates numerical understanding and computation over financial tables.
4. **TAT-QA** (Zhu et al., 2021) requires complex reasoning over both textual and tabular information.

Multimodal table prediction.

1. **Adoption** (Howard et al., 2018) is a classification task that predicts how quickly a pet will be adopted from its PetFinder listing.
2. **SkinCA**[5] is a classification task based on skin-lesion images and associated metadata. The dataset includes both malignant and benign lesion types.

[3] https://www.kaggle.com/datasets/mahdavi1202/skin-cancer

[4] https://www.kaggle.com/datasets/denozavrus/paintings-price-prediction/data

[5] `https://www.kaggle.com/datasets/mahdavi1202/skin-cancer`

3. **Pawpularity** (Howard et al., 2021) is a regression task based on pet-profile images and metadata. Its target is the Pawpularity Score, a normalized engagement measure that excludes duplicate clicks, bots, and sponsored profiles.
4. **Paintings**[6] is a regression task that predicts painting prices from images and tabular features.

A.2 Implementation Details

TO-SFT. The TO-SFT training set contains 1.5K question-answering examples and 1.2K prediction examples generated by our data-synthesis pipeline. We additionally use approximately 3.0K examples from mathematical-reasoning and code datasets, including DataScience-Instruct[7]. The synthesized data will be released publicly.

We perform full-parameter fine-tuning of Qwen3-VL-8B on four NVIDIA A800 80GB GPUs. Parameters are trained in `bfloat16`, while the visual encoder is frozen. Training runs for three epochs with a per-GPU batch size of 1 and an effective batch size of 16 after gradient accumulation. The maximum input length is 15,360 tokens, the learning rate is 1×10^{-5}, and the warmup ratio is 0.05. We use ZeRO-3 optimization.

AL-GRPO. The AL-GRPO training set contains 0.5K question-answering examples and 0.4K prediction examples generated by our data-synthesis pipeline.

We train AL-GRPO on eight NVIDIA A800 80GB GPUs for two epochs. The per-GPU batch size is 1, and gradients are accumulated over four steps, yielding an effective batch size of 32. The maximum input and output sequence lengths are both 10,240 tokens, and the maximum pixel count is 2,408,448. We use a learning rate of 5×10^{-7}, a cosine learning-rate schedule, a warmup ratio of 0.03, and ZeRO-3 optimization.

B Framework contribution beyond distillation.

A natural question is whether TWT's gains arise mainly from distillation, rather than from the proposed framework itself. We provide two key pieces of evidence showing that the improvements cannot be explained by distillation alone. **First**, TWT (8B) outperforms its teacher model, Qwen3-VL-Plus, on 6 of the 8 tasks, including WikiTQ, TAT-QA, and all four prediction tasks by large margins. If the gains came purely from distillation, the student would not be expected to consistently surpass their teacher. This effect is especially pronounced on prediction tasks, where the teacher itself performs poorly. This suggests that the framework equips the model with capabilities, such as sandbox-based model training and multi-modal fusion, that the teacher cannot reliably demonstrate within a single inference pass. **Second**, all baseline models are given the same sandbox access and identical instructions, yet their performance remains substantially below that of TWT. This shows that the gain does not come from tool availability alone, but from the training framework, which teaches the model how to effectively leverage sandbox interaction.

C Additional Analysis on Code Execution Accuracy.

During the RL training stage, we adopt a dynamic loss scaling strategy to improve the stability and accuracy of the generated code. To evaluate its effectiveness, we measure the code execution accuracy during the problem-solving process across multiple datasets, using the same datasets as in the ablation study. The results are shown in Fig. 5. As can be observed, after AL-GRPO training, TWT achieves higher code execution accuracy on most tasks.

On the TAT-QA dataset, a slight decrease in code execution accuracy is observed. However, when considered together with the overall performance improvements on this dataset, the results indicate that the model's reasoning capability on complex tasks is still significantly enhanced after AL-GRPO training. This suggests that dynamic loss scaling helps guide the model toward more effective

[6] https://www.kaggle.com/datasets/denozavrus/paintings-price-prediction/data

[7] https://huggingface.co/datasets/RUC-DataLab/DataScience-Instruct-500K/tree/main/reasoning

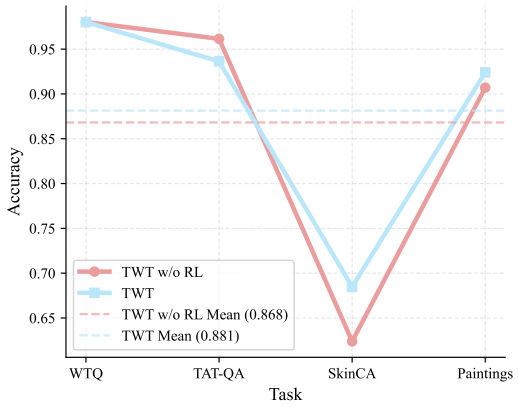


Figure 5: Comparison of code execution accuracy.

reasoning and decision-making processes; even when code execution accuracy does not consistently increase, it still contributes to improved overall reasoning performance and final task accuracy.

D Prompts Used for Data Synthesis

We use the following prompts to synthesize question-answering and prediction data.

Prompt for QA data synthesis

In a table QA task, you need to answer the following question: {}.
The image shows a partial sample of the table (`df.head()`) and is provided only to help you understand the table format. Based on the format shown in the image, analyze the table's rows and columns and write Python code to interact with the complete table in order to answer the question. All information required to answer the question must be obtained from the table. When extracting information, be robust to variations such as capitalization and abbreviations. Do not use background knowledge.
Wrap your analysis in `<analy></analy>`. Wrap code that must be executed in `<code></code>`. The code will run in a sandbox, and its result will be returned to you for further reasoning. Perform one operation at a time until you can answer the question. Wrap the final answer in `<answer></answer>`.
CSV file path: {}
Previous operation paths and outputs: {}

Prompt for prediction-data synthesis with MLLMs

You are given an image and must make a prediction for it. The task description and instructions are as follows.
Task: {}
Extract the requested features from the image. If a feature cannot be determined, leave it blank.
Feature descriptions: {}
Output format: `<answer>{JSON object}</answer>`

Prompt for prediction-data synthesis with LLMs

You are given a sample whose features were extracted from an image. These features correspond to columns in a tabular data file. Build a predictive model from the tabular data and use it to predict the target for the given sample.
Solve the table problem step by step, executing at most one code block at a time. You may write Python code to support your reasoning. The code will be executed by an external sandbox, and the resulting output will be returned inside `<code_result></code_result>`.
Task: {}
Instructions:

1. Analyze the tabular data and the sample, build a predictive model, optimize it when possible, and evaluate its performance.

2. Because the sample was obtained from an image, some features may be missing. Handle missing values appropriately during prediction; do not leave them as `NaN` or rely only on the shared features.
3. This is a multimodal prediction task. A pretrained ResNet50 model is available through `predict = ResNet50Predict(task, image_path)`.
4. In the final answer, report the tabular prediction in `predict_proba` format and the image prediction as `(tabular_prediction, image_prediction)`.

Sample: {}
Tabular data path: {}
Image path: {}
Use the following interaction protocol:
<analy>Your detailed reasoning.</analy>
<code>Optional code to execute in the sandbox.</code>
<answer>Your final answer.</answer>
<code_result></code_result> is produced only by the external sandbox and must not be generated by the model.

	Ministry	Portrait	Minister	Term of office	Term of office.1
0	Deputy Prime Minister	CF Skinner.tif	Jerry Skinner	12 December 1957	12 December 1960
1	Attorney-General	-	Rex Mason	12 December 1957	12 December 1960
2	Minister of Finance	-	Arnold Nordmeyer	12 December 1957	12 December 1960
3	Minister of Defence	-	Philip Connolly	12 December 1957	12 December 1960
4	Minister of Education	-	Philip Skoglund	12 December 1957	12 December 1960

Figure 6: Table preview supplied to the model in the question-answering example.

E EXAMPLES OF TWT RESPONSES

E.1 TABLE QUESTION ANSWERING

Query

The model is given the table preview shown in Figure 6.
Question: Who served as the Minister of Foreign Affairs for the Second Labour Government of New Zealand, and on what dates did the minister serve?
Table file path: `xxx`

Model analysis 1

The question asks for the Minister of Foreign Affairs during New Zealand's Second Labour Government and the corresponding term dates. The displayed sample contains ministries such as Deputy Prime Minister and Attorney-General, but it does not contain the requested ministry. I therefore need to inspect the complete CSV file. I will first verify the column names and table structure, including possible variants such as *Minister for Foreign Affairs*.

Code

```
import pandas as pd

df = pd.read_csv("xx")
print(df.head().to_markdown(index=False))
```

Tool output

```
(Omitted due to space.)
```

Model analysis 2

The first five rows do not contain *Minister of Foreign Affairs* or a similar title. I will search the complete `Ministry` column, case-insensitively, for both *External Affairs* and *Foreign Affairs*.

Code

```
foreign_affairs_rows = df[
    df["Ministry"].str.contains(
        "External Affairs|Foreign Affairs",
        case=False,
        na=False,
    )
]
```

```
columns = ["Ministry", "Minister", "Term of office", "Term of office
    .1"]
print(foreign_affairs_rows[columns].to_markdown(index=False))
```

Tool output

```
(Omitted due to space.)
```

Model analysis 3

The search returns *Minister of External Affairs*, the historical title used for the foreign-affairs portfolio during this period. The listed minister is Walter Nash, whose term ran from 12 December 1957 to 12 December 1960.

Final answer

Walter Nash served as Minister of External Affairs from 12 December 1957 to 12 December 1960.



Figure 7: Pet image supplied to the model in the Pawpularity-prediction example.

E.2 Multimodal Table Prediction

Query

You are given the pet image shown in Figure 7. Predict the pet's Pawpularity score using a table of pet metadata and a pretrained ResNet50 image model. Account for the nature of the task and combine the tabular and image predictions.
Instructions:

1. Extract the required tabular features from the image.
2. Build a tabular model and predict the target for this sample.
3. Obtain the image-model prediction using `ResNet50Predict(task, image_path)`.
4. Fuse the tabular and image predictions to produce the final result.

Photo metadata: Omitted due to space.
Image path: `xx`
Tabular data path: `xx`

Model analysis 1: feature extraction

The image provides the following categorical features.

Extracted features

```
{"Eyes": 1, "Face": 1, "Near": 1, "Action": 0,
 "Accessory": 0, "Group": 0, "Collage": 0, "Human": 0,
 "Occlusion": 0, "Info": 0, "Blur": 0}
```

Model analysis 2: prediction plan

This is a regression task in which the Pawpularity score must be predicted from both tabular features and an image. I will load and inspect the training data, prepare the feature matrix and target, train and validate a tabular regressor, retrain it on all available data, and then obtain the independent image-model prediction. Finally, I will combine the two predictions.

Code

```
import pandas as pd

train_df = pd.read_csv("xx")
train_df.head()
```

Tool output

```
(Omitted due to space.)
```

Model analysis 3: model training

The training table contains an identifier, binary image-derived features, and the target column `Pawpularity`. I will exclude the identifier and target from the feature matrix and use a random-forest regressor. A held-out validation split provides an estimate of the tabular model's error.

Code

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
import numpy as np

feature_cols = [
    column
    for column in train_df.columns
    if column not in ["Id", "Pawpularity"]
]
X = train_df[feature_cols]
y = train_df["Pawpularity"]

X_train, X_val, y_train, y_val = train_test_split(
    X,
    y,
    test_size=0.2,
    random_state=42,
)

model = RandomForestRegressor(
    n_estimators=100,
    random_state=42,
)
model.fit(X_train, y_train)

y_pred_val = model.predict(X_val)
rmse = np.sqrt(mean_squared_error(y_val, y_pred_val))
rmse
```

Tool output

```
21.601680444759133
```

Model analysis 4: multimodal prediction

The random-forest model obtains a validation RMSE of approximately 21.6. I will retrain it on the complete training set. The sample does not contain `Subject Focus`, so I will impute that feature with its mode, 0. I will then obtain both the tabular prediction and the pretrained ResNet50 prediction.

Code

```
full_model = RandomForestRegressor(
    n_estimators=100,
    random_state=42,
)
full_model.fit(X, y)

sample_features = {
    "Eyes": 1,
    "Face": 1,
    "Near": 1,
    "Action": 0,
    "Accessory": 0,
    "Group": 0,
    "Collage": 0,
    "Human": 0,
    "Occlusion": 0,
    "Info": 0,
    "Blur": 0,
    "Subject Focus": 0,
}

sample_df = pd.DataFrame([sample_features])
sample_X = sample_df[feature_cols]

tabular_pred = full_model.predict(sample_X)[0]
image_pred = ResNet50Predict("pawpularity", "xxx")

(tabular_pred, image_pred)
```

Tool output

```
(np.float64(38.02495413188204), 94.54036712646484)
```

Model analysis 5: prediction fusion

The tabular and image predictions are 38.02495413188204 and 94.54036712646484, respectively. Because Pawpularity is primarily visual in this example, I assign weights of 0.1 and 0.9 to the tabular and image predictions, respectively.

Final answer

88.88873581602657

The final prediction is obtained by fusing the two modalities. The example above uses a simple weighted average based on the visual nature of the target. In other cases, the model may also consider factors such as missing-feature rates and the estimated reliability of each modality.