TABLEMOE: REINFORCEMENT LEARNING TO ROUTE CONNECTOR EXPERTS FOR MULTIMODAL TABLE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tables are not just pictures of numbers; they are structured objects where rows, headers, units, and grouping all matter. We ask whether a model can learn to combine different table views through answer-level feedback, instead of committing to one rigid representation. TableMoE keeps a pretrained visual connector and adds experts aligned to HTML, JSON, and table-redrawing code. At each visual token, the router selects two experts and mixes their outputs. A reward-guided training stage then adapts the router, experts, and language-model adapters together.

Across six benchmarks, this adaptation improves average accuracy by 4.14 percentage points over dense softmax SFT, with a similar gain from the self-critical variant. Analysis of the learned mixture reveals a dominant General expert with spatially varying companions, while a fixed General+JSON pair retains most of the learned routing accuracy on the development subset. These findings support reward-based adaptation of the aligned mixture, but provide more limited evidence that spatially adaptive expert selection is necessary for accurate answers.

1 INTRODUCTION

A table image is more than a stack of numbers. To answer a question, a model has to connect values to the right header, row, unit, and level of aggregation. That becomes difficult when the structure is subtle: headers stretch across columns, labels carry meaning beyond the cell text, or financial tables layer units and totals on top of one another (Chen et al., 2021; Zhu et al., 2021; Smock et al., 2022). The visual connector is central to this challenge: it determines which table representation reaches the language model.

Different representations make different relationships explicit. HTML keeps hierarchy and spanning structure; JSON captures cell contents and positions; a drawing program describes the table as a layout that is redrawn. These are not interchangeable views, and they motivate separate connector experts. But the useful question is not whether an expert can reproduce a table well; it is whether a model can learn when to combine them in a way that helps an answer. That brings us to a practical question: *does the benefit come from changing experts at every token, or can a strong fixed pair keep most of the performance?*

We study this with TableMoE, a mixture of four visual connectors. One expert keeps the pretrained visual merger, while three others are aligned separately to HTML, JSON, and table-redrawing code. At the token level, a router selects two experts and combines their outputs before the language model reads the visual sequence. This keeps the pretrained representation in play without forcing the same pair everywhere. The expert names refer to their supervision signal, not to a guaranteed division of reasoning tasks.

This design builds on connector mixtures and separately aligned experts such as CuMo and Chart-MoE (Li et al., 2024; Xu et al., 2025). We adapt MoE-GRPO (Ko et al., 2026) so the mixture learns from answer rewards rather than only from supervised targets. For each question, the model explores several routing trajectories while greedily decoding candidate answers, and the variation across those trajectories comes from expert selection. The resulting rewards update the router, experts, and language-model adapters in one RL stage. We call this default configuration **Multi-GRPO**. We

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.

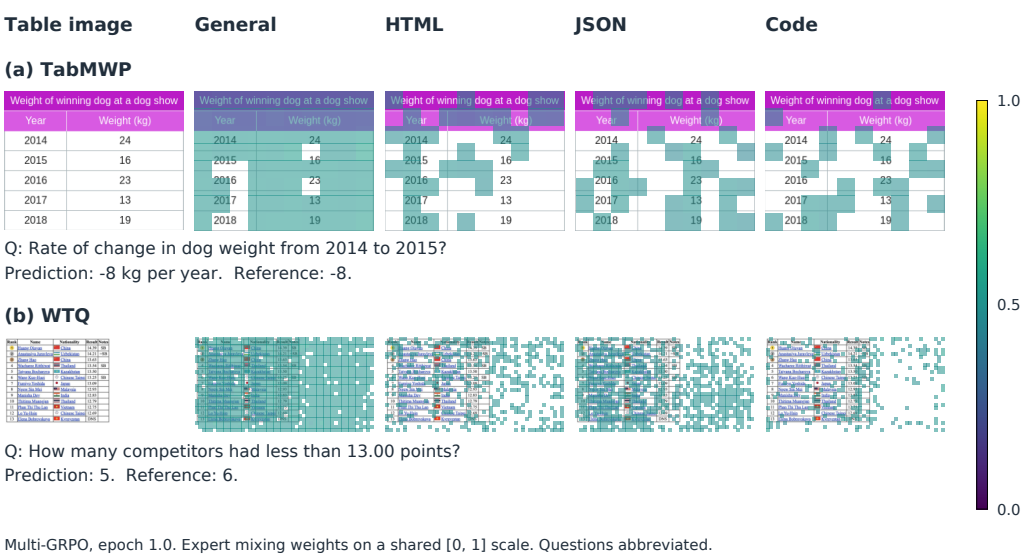


Figure 1: **Token-wise Top-2 expert mixing.** Each token selects two experts. Colored regions show their fusion weights on a shared $[0, 1]$ scale; unselected positions are transparent. Active weights are close to 0.5, while expert identities vary spatially.

also study SC-RPO, which uses the reward of the deterministic deployment route as a self-critical baseline. In both settings, the model deploys one deterministic Top-2 route.

The benchmark results favor reward-based adaptation: Multi-GRPO improves the six-task average over deterministic and dense softmax SFT by 4.83 and 4.14 percentage points, respectively, and SC-RPO performs similarly. The routing analyses explain a different aspect of the model. Figure 1 shows a widely used General expert whose companion changes across the image. Yet frequent selection and sensitivity to expert exclusion have different rankings. Moreover, a fixed General+JSON pair nearly matches learned routing on the intervention subset. The combination of these observations separates gains from adapting the mixture from evidence that each spatial routing decision is needed for the answer.

Our contributions connect model design with an empirical analysis of its use. We build a table connector mixture that preserves the pretrained visual signal alongside experts aligned to HTML, JSON, and table-redrawing code. We show that reward-based adaptation improves the same four-expert architecture beyond the evaluated SFT baselines across six benchmarks. Finally, we examine the learned mixture through spatial maps, expert exclusion, and fixed-pair interventions. These analyses show where expert composition affects answers and where substantial changes in selection leave aggregate accuracy nearly unchanged, delimiting what the benchmark gains establish about adaptive routing.

2 Related Work

Table reasoning has long been approached from two directions: improving how information is encoded, and then deciding how that information is used. Our work sits at the intersection of both. Recent table-specific models such as Table-LLaVA and TabPedia improve the representation that reaches the model, while studies of rendering and source quality show that the way a table is presented affects reasoning performance (Zheng et al., 2024; Zhao et al., 2024; Fu et al., 2025; Yang et al., 2025). TableMoE addresses the same representation problem through a connector mixture that learns to combine differently aligned visual features at the token level.

The closest methodological precedent is the line of work on aligned expert connectors and reward-guided routing. CuMo and Uni-Med show that multimodal systems can benefit from expert-specific processing (Li et al., 2024; Zhu et al., 2024), and ChartMoE is especially relevant because it aligns experts to distinct representation targets before routing (Xu et al., 2025). TableMoE follows that

pattern, but for table images and with a reward-driven routing stage that learns from answer feedback rather than from supervised routing labels. The same broader idea also appears in RL-based table adaptation such as TableMix and V-tableR1 (Liu et al., 2026; Jiang et al., 2026), while MoE-GRPO provides a route-optimized training recipe for expert selection (Shao et al., 2024; Ko et al., 2026). We bring these threads together in one table-focused setting and then test whether the learned routing choices themselves are necessary for the observed gains.

This is also why we include a self-critical comparison: it helps separate the value of the aligned mixture from the value of a particular reward baseline. The comparison follows the logic of sequence-level self-critical training and ReMax, but our main question is not which RL variant is universally superior; it is whether the mixture is useful and whether the learned selection pattern is meaningfully contributing to the answer.

3 METHOD

TableMoE takes a table image I and a question q, and produces an answer y. The connector maps a sequence of V visual features into the language model's embedding space. We first align individual experts to different representations of the same table, then use answer feedback to learn which pairs to combine at each visual token. Figure 2 summarizes both stages.

3.1 TOKEN-LEVEL EXPERT SELECTION

The connector is meant to make a local choice of representation, not a global one. A header and a numeric cell do not need to be processed by the same pair of experts. The frozen vision encoder produces merger tokens $\mathbf{x}_i \in \mathbb{R}^{5120}$. Four independent two-layer MLP experts then map each token into the language hidden size, 3584. Architecture dimensions and window ordering are detailed in Appendix F.

The router is a bias-free map $z_i = W_r \mathbf{x}_i \in \mathbb{R}^4$. Expert selection depends on visual features. Sampling and fusion are different functions of the same logits. With temperature $\tau = 1$,

$$p_i = \mathrm{softmax}(z_i/\tau). \tag{1}$$

After a pair is chosen, fusion uses sigmoid scores $s_{i,e} = \sigma(z_{i,e})$. Because σ is strictly increasing, $\mathrm{Top2}(z_i) = \mathrm{Top2}(s_i)$. Deployment therefore selects the two largest logits:

$$S_i = \mathrm{Top2}(z_i), \qquad \mathbf{h}_i = \sum_{e \in S_i} w_{i,e} f_e(\mathbf{x}_i), \quad w_{i,e} = \frac{s_{i,e}}{\sum_{j \in S_i} s_{i,j}}. \tag{2}$$

Exactly two experts contribute at each token, and the fusion weights sum to one. They differ from the sampling probabilities and are computed rather than fixed to $1/2$. The pair may change from token to token. The language model receives $(\mathbf{h}_1, \ldots, \mathbf{h}_V)$ together with the question.

Training draws an ordered pair without replacement. If e_1 is drawn first,

$$\log P_i(e_1, e_2) = \log p_{i,e_1} + \log \frac{p_{i,e_2}}{1 - p_{i,e_1}}. \tag{3}$$

A routing trajectory consists of one such draw at every visual token. The answer supplies a shared reward for the trajectory, while the router objective averages token-level probability ratios. This trains local selections from answer-level feedback without expert-selection labels. At deployment, the ordered draws are replaced by the deterministic pairs in Eq. 2.

3.2 EXPERT ALIGNMENT

The alignment stage gives each expert a different view of the same visual content before the router ever learns how to use them. The General expert keeps the pretrained visual merger. The HTML, JSON, and Code experts start from the same initialization and are trained separately on their own image-to-target supervision tasks. HTML preserves logical structure, including headers and row/column spans. JSON serializes cell contents and positions. Python `matplotlib` code reconstructs the table's layout and formatting. That code target is not meant to perform arithmetic; it is a way of teaching the model how the table is drawn.

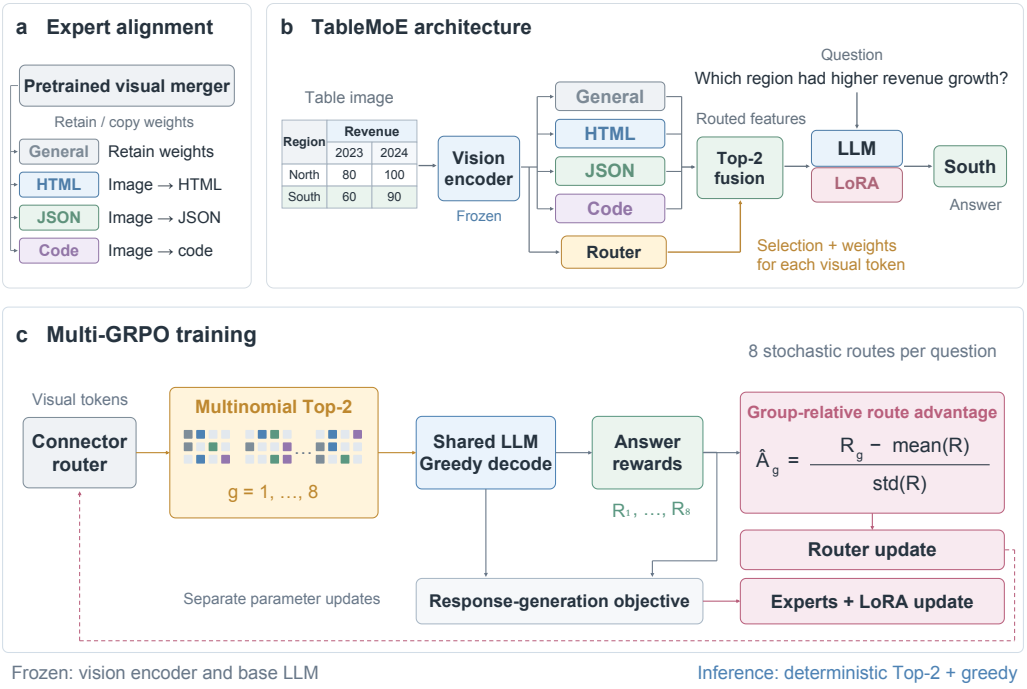


Figure 2: **TableMoE overview.** (a) Independent expert alignment. (b) Token-wise Top-2 fusion. (c) Multi-GRPO training with eight stochastic routes and answer rewards. Inference uses deterministic Top-2 routing and greedy decoding; the vision encoder and base LLM remain frozen during RL.

TableMoE-Align uses roughly 400K source tables from PubTabNet, FinTabNet.c, PubTables-1M, SynthTabNet, and TableBank (Zhong et al., 2020; Smock et al., 2023; 2022; Nassar et al., 2022; Li et al., 2020). The retained training pairs include 372,892 HTML targets, 372,892 JSON targets, and 264,540 code targets: 1,010,324 image-target pairs in all. These counts refer to target-specific pairs rather than unique tables. Appendix E contains the source breakdown and code-based reconstruction examples.

Each aligned merger initializes one expert, and the router starts randomly. Only connector parameters are optimized during alignment, with teacher-forced cross-entropy on target tokens. No expert-selection labels are supplied. The point is not to predefine a semantic split among experts; it is to give the mixture a useful set of structured views before reward-driven routing learns how to deploy them.

3.3 Reward-Guided Adaptation

Alignment teaches the experts how to represent tables. The next step is to learn which combinations actually support correct answers. We optimize the router and response model in a single RL stage, using Multi-GRPO as the main setting and evaluating alternative reward baselines separately.

Exploring routes with answer feedback. For each image question, Multi-GRPO samples eight token-wise routing trajectories and greedily decodes one response per trajectory. The route choices remain fixed during decoding. Each response contains evidence, reasoning, and an answer. A second pass of the same model receives the extracted evidence and question without the image. Rewarding this answer adds a check on whether the evidence can support the response (Li et al., 2026), although it does not directly verify that the evidence is visually grounded. Both branches receive task-based answer rewards and tag-based format rewards. The router learns from the answer rewards; experts and language-model adapters also learn from the format rewards.

Router objective. Let $R_g^{(b)}$ denote branch b's answer reward for route g. Multi-GRPO centers each branch's rewards over all eight routes and normalizes by the population standard deviation, setting advantages to zero when the deviation is at most 10^{-6}. The two alternatives use one deterministic Top-2 route and seven stochastic routes. SC-RPO subtracts the deterministic route's reward; Stochastic-Mean subtracts the mean of the seven stochastic rewards. Neither alternative divides by the standard deviation. For each stochastic route and visual token, let $\rho_{g,i} = P_\theta(e_{g,i,1}, e_{g,i,2})/P_{\text{old}}(e_{g,i,1}, e_{g,i,2})$, where the denominator is saved at sampling time. The gate loss is

$$\mathcal{L}_{\text{gate}} = -\frac{1}{|\mathcal{G}_s|V} \sum_{g\in\mathcal{G}_s} \sum_{i=1}^{V} \sum_{b=1}^{2} \min\Big(\rho_{g,i}A_g^{(b)}, \text{clip}(\rho_{g,i}, 0.8, 1.2)A_g^{(b)}\Big). \tag{4}$$

Here $\mathcal{G}_s$ contains only stochastic routes. Both answer branches have unit weight; format rewards do not enter this objective. A route's advantage is shared across its visual tokens. The loss averages per-token ratios rather than using a product over the full trajectory.

Adapting the selected representations. With the router fixed, we replay its sampled routes and update the experts and LoRA using an advantage-weighted response loss. Answer and format rewards are standardized separately over all eight routes and weighted by 0.5 and 0.05. The token ratio uses a detached current probability as its denominator: its value is one, with a nonzero gradient, so token clipping is inactive. A KL penalty of 0.001 per branch anchors this update to the initialization-time experts with LoRA disabled. Separate router and response updates thus adapt both which representations are selected and how the model uses them. Appendix F gives the full objectives and update order. Inference uses deterministic Top-2 routing and greedy decoding.

4 Experiments

We first ask whether reward-based adaptation improves table reasoning over a supervised version of the same mixture. Then we look at what the trained router is actually doing and how much performance changes when its expert choices are constrained. Together, these comparisons speak to two different questions: whether the full recipe is useful, and whether the learned routing pattern itself is essential.

4.1 Experimental Setup

Models and benchmarks. We use Qwen2.5-VL-7B-Instruct with four connector experts and rank-64 LoRA. Both RL configurations start from the same aligned connector and train on 9,450 questions, including 8,505 image questions; the text-only rows do not contribute local training gradients. We evaluate table question answering on TabMWP, WTQ, HiTab, and TAT-QA, and fact verification on TabFact and InfoTabs. The main aggregate is the unweighted mean of these six accuracies. FeTaQA BLEU results from other systems appear separately in Appendix J.

Det-SFT uses deterministic Top-2 routing, while Soft-SFT mixes all four experts with dense softmax weights. Because they share the same four-expert architecture, these baselines are a fair test of the full adaptation recipe, including the expert and adapter updates. Multi-GRPO and SC-RPO use epoch-1 checkpoints but differ in rollout composition, normalization, and training budgets (Appendix G). Literature-reported results in Table 1 provide context, but their evaluation protocols may differ from ours.

Evaluation and interventions. TQA uses normalized string or unordered-set matching with a containment fallback; TFV maps responses to task labels. Failed parses score zero. Appendix H specifies the scorer and the parser correction. We use Dev392 for training comparisons, Dev300 for routing interventions, and a separate five-task pool of 500 questions for reward baselines. Dev300 contains 100 questions each from TabMWP, WTQ, and HiTab. Every intervention uses the frozen Multi-GRPO checkpoint, greedy decoding, and a 1,024-token limit. The visual cases were selected before inspecting their outcomes. Paired 95% intervals use 2,000 dataset-stratified table-cluster bootstrap replicates. All intervention predictions use the corrected parser; benchmark results retain the original scorer. Appendices A–C provide the subset definitions, scoring checks, and uncertainty estimates.

Table 1: **Table reasoning results.** Avg. is the unweighted mean accuracy across six benchmarks. Superscripts identify literature sources and re-evaluations; protocols may differ across sources.

Method	TQA				TFV		Avg.
	TabMWP	WTQ	HiTab	TAT-QA	TabFact	InfoTabs	
General-Purpose MLLMs							
Qwen2.5-VL-7B$^{\mathrm{TB,TM,MMT}}$	92.48	65.85	67.09	70.54	83.01	77.91	76.15
Qwen2.5-VL-7B$^{\mathrm{TD}}$	63.69	54.37	62.69	51.94	75.81	70.13	63.11
Qwen2.5-VL-7B$^{\diamond}$	78.27	69.22	59.01	47.28	69.79	68.41	65.33
Gemma-4-E4B-it	78.16	65.22	41.12	49.09	72.43	66.02	62.01
Phi-4-Multimodal-Instruct	69.58	54.24	51.33	38.21	69.93	56.61	56.65
Qwen3-VL-8B-Instruct	83.93	81.56	62.50	52.72	75.49	75.44	71.94
Table-Specialized MLLMs							
Table-LLaVA-7B†	57.78	18.43	10.09	12.82	59.85	65.26	37.37
TabPedia-7B†	10.66	23.53	6.54	13.08	35.49	2.43	15.29
SynTab-LLaVA-7B†	88.30	39.59	35.66	51.94	70.78	69.42	59.28
CoReTab-Qwen2.5-VL-7B†	97.14	60.93	67.45	82.60	84.28	76.04	78.07
RL-Enhanced Table MLLMs							
TableMix†	99.20	81.32	82.25	78.52	88.96	88.72	86.50
MM-Table-R1-3B†	97.10	74.58	76.14	74.52	87.13	85.31	82.46
Table-R1-7B†	92.60	50.30	58.20	48.06	73.40	62.80	64.23
V-tableR1-4B†	83.38	63.37	47.24	54.23	87.95	88.94	70.85
HIPPO-8B†	87.50	55.77	63.00	60.75	82.27	75.74	70.84
HIPPO-8B*	48.49	37.11	38.52	21.50	72.70	71.46	48.30
Turbo-8B†	96.75	67.80	72.15	73.21	85.81	81.89	79.60
Routing Baselines							
TableDART (TableGPT2-7B + Ovis2-8B)†,‡	84.54	70.58	74.37	62.05	81.37	76.22	74.86
Qwen2.5-VL-7B + Det-SFT	79.55	57.85	66.50	61.27	78.12	66.87	68.36
Qwen2.5-VL-7B + Soft-SFT	83.71	57.41	68.46	62.56	77.95	64.20	69.05
TableMoE (Ours)							
TableMoE (Qwen2.5-VL-7B, Multi-GRPO)	88.08	64.46	69.54	65.03	81.02	71.00	73.19
TableMoE (Qwen2.5-VL-7B, SC-RPO)	88.47	64.11	69.67	64.90	80.85	72.33	73.39

Superscripts TB, TM, MMT and TD identify Qwen2.5-VL-7B scores reported by Turbo, TableMix, MM-Table-R1 and TableDART. † marks literature-reported results. $^{\diamond}$ marks our Qwen2.5-VL-7B evaluation on the splits adopted by HIPPO and TableDART; * marks our HIPPO re-evaluation. ‡ indicates serialized table text and image inputs, with Gemini 2.0 Flash for fusion. Literature results can differ in subsets, rendering, prompts and scoring. Det-SFT and Soft-SFT use the same four-expert TableMoE architecture. Multi-GRPO is our connector adaptation of MoE-GRPO; SC-RPO uses a self-critical routing baseline. All four configurations use Qwen2.5-VL-7B.

4.2 Reward-Based Adaptation Improves the Evaluated Mixture

Gains across tasks. Both RL configurations improve over both SFT configurations on every benchmark in Table 1. Multi-GRPO reaches 73.19% average accuracy, compared with 68.36% for Det-SFT and 69.05% for Soft-SFT; SC-RPO reaches 73.39%. The gains extend across arithmetic, hierarchical tables, and fact verification. In particular, improvements over Soft-SFT are larger on WTQ and InfoTabs than on HiTab. The pattern suggests that downstream answer feedback matters beyond changing the supervised fusion rule. Since RL also updates experts and language adapters, we turn to fixed-model interventions below to examine selection itself.

The self-critical baseline is not decisive. SC-RPO and Multi-GRPO each lead on three benchmarks. Their 0.20-point average gap is driven by SC-RPO's 1.33-point InfoTabs gain; the other five tasks nearly balance out. The ordering reverses on Dev392, where Multi-GRPO obtains 55.9% micro accuracy and SC-RPO 54.8%. We therefore do not identify either routing configuration as consistently better. A separate ablation fixes the one-deterministic-plus-seven-stochastic rollout group and changes only the routing reward baseline. Stochastic-Mean reaches 57.8% against SC-RPO's 56.2%, with a paired interval spanning zero (Appendix C). The benchmark gains thus do not depend on demonstrating an advantage for the self-critical reference.

Table 2: **Evaluation on TableVQA-Bench.** Accuracy (%) with Qwen2.5-VL-7B. VWQ, VWS, VTF, and FTN denote VWTQ, VWTQ-Syn, VTabFact, and FinTabNetQA; Avg. is their unweighted mean.

Method	VWQ	VWS	VTF	FTN	Avg.
Det-SFT	56.53	62.80	63.20	74.00	64.13
Soft-SFT	56.67	62.40	54.80	78.40	63.07
Multi-GRPO	59.87	66.40	63.20	75.20	66.17
SC-RPO	61.07	64.80	62.00	77.20	**66.27**

The broader comparison also shows where adaptation remains incomplete. Both RL variants fall below our unadapted backbone's 69.22% WTQ score, despite improving its six-task average of 65.33%. The strongest literature-reported averages remain higher. These results position TableMoE as a study of aligned mixtures and their use under reward-based adaptation; they do not establish a new overall state of the art.

4.3 Evaluation on TableVQA-Bench

Table 2 evaluates the same four configurations on TableVQA-Bench. Its VWTQ, VWTQ-Syn, VTabFact, and FinTabNetQA components derive from WTQ, TabFact, and FinTabNet (Pasupat & Liang, 2015; Chen et al., 2020; Smock et al., 2023).[1] These source families also appear in downstream training or expert alignment, so this evaluation does not establish unseen-source generalization. We report the unweighted component average, using the checkpoint choices documented in Appendix H.

The aggregate pattern agrees with the six-task evaluation. Multi-GRPO improves over Det-SFT by 2.04 points and Soft-SFT by 3.10 points. The component results show a trade-off: both RL variants improve VWTQ and VWTQ-Syn over both SFT configurations, but trail Soft-SFT on FinTabNetQA. SC-RPO and Multi-GRPO again split the component wins, with only 0.10 points separating their averages. Reward-based adaptation helps in aggregate, without uniformly improving every component or favoring one RL configuration.

4.4 A Frequent General Expert with Spatially Varying Companions

The accuracy comparisons leave open how the experts are used. Figure 1 provides a first observation: General participates across much of each image, while its companion changes across tokens. Each of the four maps displays one expert's normalized fusion weight; only two experts contribute at any position.

For question q, let $M_{q,i,e}$ be expert e's fusion weight at token i, with zero for an unselected expert. We overlay these weights on token footprints using a shared $[0, 1]$ scale. Across 725,256 visual tokens in the 300 traces, 99.0% of nonzero weights lie in $[0.45, 0.55]$. The nearly uniform colors therefore reveal an approximately equal mixture *within the selected pair*; spatial variation lies mainly in the identities being mixed. These coefficients describe connector fusion rather than attention or answer attribution.

A recurring General component. We summarize all 300 traces rather than relying on the displayed cases. For a question with V_q visual tokens, define mean weight and selection coverage as

$$m_{q,e} = \frac{1}{V_q} \sum_i M_{q,i,e}, \qquad c_{q,e} = \frac{1}{V_q} \sum_i \mathbb{I}[e \in S_{q,i}]. \tag{5}$$

We then average each quantity over questions within a dataset, giving equal weight to images of different sizes. Weight share measures an expert's average scalar fusion coefficient; coverage measures how often it is selected. In particular, coverage is not divided by the two available selection slots.

General receives 46.6–48.7% of the mean fusion weight and appears at 91.2–94.2% of visual tokens across the three datasets (Table 4). Thus the typical pair combines the pretrained connector with an

[1] https://github.com/naver-ai/tablevqabench

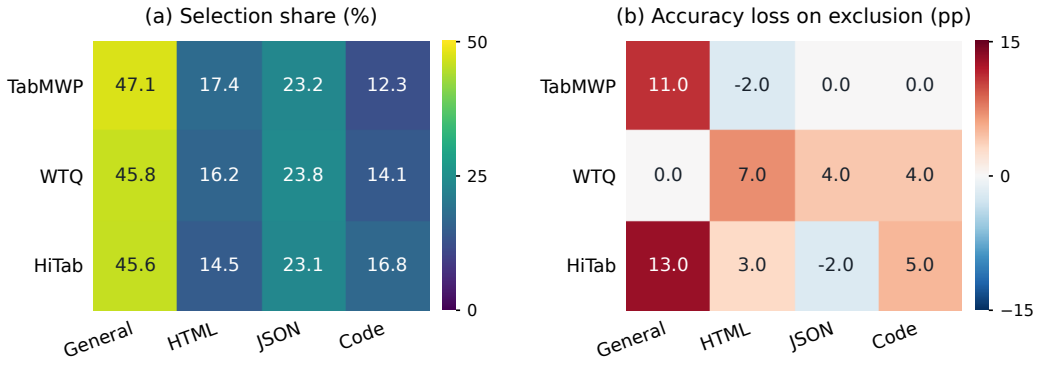


Figure 3: **Expert use and exclusion effects on Dev300.** (a) Share of Top-2 selection slots (maximum 50%). (b) Accuracy loss after excluding an expert and rerouting; positive values indicate degradation. All conditions use the same Multi-GRPO checkpoint and corrected scoring protocol.

aligned expert. JSON has the largest mean weight among the aligned experts, while Code's share increases from TabMWP to HiTab. This is a description of selection, not evidence that the expert names correspond to distinct reasoning skills.

Spatial variation and answer quality. Every question uses at least three expert pairs, and 230 of 300 use all six. The router therefore makes spatially varying selections even though its active weights are close to equal. Observing more pairs also depends on the number of visual tokens, so pair counts alone cannot establish greater task specialization.

The examples show why outcome-based analysis is needed. TableMoE correctly computes the TabMWP change as -8 kg per year, but predicts five rather than six qualifying competitors on WTQ despite using all six pairs. A varied routing map can accompany either outcome. We next change the expert assignments while holding the model parameters fixed to test which aspects of this behavior affect accuracy.

4.5 Which Expert Choices Affect the Answer?

Frequent selection does not rank exclusion effects. We compare how often an expert is used with the accuracy change when it is unavailable. Exclusion removes one expert from the candidate set, selects the two highest-scoring remaining experts at each token, and renormalizes their sigmoid scores. This preserves two active experts and tests sensitivity to replacing a member of the learned mixture. Figure 3(a) reports each expert's share of selection slots, $c_{q,e}/2$ averaged over questions, with maximum 50%. Panel (b) gives the accuracy loss under exclusion.

The General expert has the largest selection share, yet its effect depends on the dataset. Excluding it lowers accuracy by 11 points on TabMWP and 13 on HiTab, with paired 95% intervals of [4.0,19.0] and [3.9,22.9]. On WTQ, the aggregate is unchanged: eight previously correct answers become wrong, and eight previously wrong answers become correct. HTML is selected much less frequently, but its exclusion produces the largest observed WTQ loss, 7 points. The remaining ten exclusion intervals include zero, so the ranking is descriptive rather than a resolved ordering of expert importance.

These results distinguish prevalence from sensitivity. Selection share counts how often an expert participates; exclusion also depends on the replacement selected by the router and how the language model uses the changed features. It consequently measures the behavior of the remaining mixture, rather than an expert's standalone ability. General is a recurring part of the representation, but its high usage does not imply the same accuracy effect on every task. Tracking the individual predictions changed by exclusion is therefore useful alongside the aggregate loss: balanced numbers of improvements and regressions can conceal substantial changes in which questions the model answers correctly.

Table 3: **Learned routing and all six fixed expert pairs.** Rescored accuracy (%) on the same Dev300 questions, with frozen Multi-GRPO weights. Δ is learned minus fixed Macro in percentage points; intervals appear in Appendix B.

Routing	TabMWP	WTQ	HiTab	Macro	Δ
Learned Top-2	81.0	59.0	39.0	59.67	–
General + HTML	80.0	50.0	39.0	56.33	+3.33
General + JSON	83.0	56.0	37.0	58.67	+1.00
General + Code	82.0	53.0	39.0	58.00	+1.67
HTML + JSON	67.0	50.0	27.0	48.00	+11.67
HTML + Code	57.0	46.0	26.0	43.00	+16.67
JSON + Code	79.0	51.0	35.0	55.00	+4.67

Learned routing versus fixed pairs. We next force each of the six expert pairs at every visual token. For a pair P, fusion uses $w_{i,e}^{P} = s_{i,e} / \sum_{j \in P} s_{i,j}$ for $e \in P$. Only the expert identities are fixed; weights still vary with the original router scores, rather than being set to $1/2$. No parameters are retrained. Table 3 reports the three dataset accuracies and their unweighted mean.

Learned routing attains the highest observed Macro, 59.67%, while General+JSON reaches 58.67%. The three-net-answer difference comes from 18 questions answered correctly only by learned routing and 15 only by the fixed pair. The paired 95% interval for learned minus fixed is $[-2.69, 4.94]$ points. General+JSON is selected as the strongest pair on this same subset, and the interval is not adjusted for that selection; this comparison establishes neither superiority nor equivalence.

Expert composition matters within these interventions. General-containing pairs are the three strongest fixed alternatives in Macro. General+JSON exceeds learned routing on TabMWP, whereas learned routing is better on WTQ. Removing spatial adaptation therefore changes which questions are answered correctly, even when the aggregate gap is small. These are interventions on one trained model, not comparisons against separately optimized fixed-pair models. A follow-up comparison should optimize fixed pairs separately under matched training budgets and choose the pair on a validation set before testing it on held-out questions.

Forcing General+JSON changes the selected pair at 58.1% of visual tokens, averaged over questions. The small accuracy gap despite this substantial reassignment shows that variation in a routing map is a poor proxy for the benefit of that variation. The measured gains from RL and the incremental gains from changing expert identities are different quantities: a model can benefit from adapting its mixture while remaining close in accuracy to a strong fixed combination.

Limitations. Our evidence covers one backbone and one seed per RL configuration, and joint expert and adapter updates prevent attributing benchmark gains to router learning alone. The interventions use 300 development questions without retraining or independent fixed-pair selection; shared source families in TableVQA-Bench and scorer sensitivity further limit conclusions about generalization and routing necessity (Appendix K).

5 Conclusion

TableMoE combines a pretrained visual connector with experts aligned to HTML, JSON, and table-redrawing code, then adapts the mixture through answer-level rewards. Both RL configurations improve over the evaluated SFT baselines across all six benchmarks, supporting reward-guided adaptation of the aligned mixture.

The interventions clarify how this mixture is used. General is frequently selected, but selection frequency does not rank the accuracy loss under exclusion. A fixed General+JSON pair retains most of the learned routing accuracy on Dev300 despite substantial changes in expert assignments. These findings separate gains from adapting the mixture from evidence that every spatial routing decision is necessary. For future connector designs, the central test is how expert choices change answer correctness: which questions improve, which regress, and which errors persist even when aggregate accuracy changes little.

AI USE STATEMENT

We used LLMs to improve the grammar and clarity of the manuscript and to generate synthetic data for the alignment pre-training of TableMoE's connector experts. The generated data were filtered and verified before use, and all AI-assisted text was reviewed and revised by the authors. The authors take full responsibility for the final manuscript and all associated artifacts.

REPRODUCIBILITY STATEMENT

To support reproduction, we document expert-alignment data statistics and targets in Appendix E, architecture and training objectives in Appendix F, and reported training configurations in Appendix G. Evaluation subsets, routing interventions, and uncertainty estimation are described in Appendices A and B. Appendix H specifies the scoring protocol and parser correction, while Appendix I records the remaining gaps in run provenance. Our anonymized code repository is available at https://anonymous.4open.science/r/TableMoE-1159/.

REFERENCES

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. TabFact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkeJRhNYDH.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 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. Association for Computational Linguistics, November 2021. doi: 10.18653/v1/2021.emnlp-main.300. URL https://aclanthology.org/2021.emnlp-main.300.

Xingyu Fu, Minqian Liu, Zhengyuan Yang, John Corring, Yijuan Lu, Jianwei Yang, Dan Roth, Dinei Florencio, and Cha Zhang. ReFocus: Visual editing as a chain of thought for structured image understanding, 2025. URL https://arxiv.org/abs/2501.05452.

Yubo Jiang, Yitong An, Xin Yang, Abudukelimu Wuerkaixi, Xuxin Cheng, Fengying Xie, Zhiguo Jiang, Cao Liu, Ke Zeng, and Haopeng Zhang. V-tabler1: Process-supervised multimodal table reasoning with critic-guided policy optimization, 2026. URL https://arxiv.org/abs/2604.20755.

Dohwan Ko, Jinyoung Park, Seoung Choi, Sanghyeok Lee, Seohyun Lee, and Hyunwoo J. Kim. Moe-grpo: Optimizing mixture-of-experts via reinforcement learning in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14957–14967, June 2026.

Jiachen Li, Xinyao Wang, Sijie Zhu, Chia-Wen Kuo, Lu Xu, Fan Chen, Jitesh Jain, Humphrey Shi, and Longyin Wen. CuMo: Scaling multimodal LLM with co-upcycled mixture-of-experts, 2024. URL https://arxiv.org/abs/2405.05949.

Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. TableBank: A benchmark dataset for table detection and recognition, 2020. URL https://arxiv.org/abs/1903.01949.

Zongxia Li, Wenhao Yu, Chengsong Huang, Zhenwen Liang, Rui Liu, Fuxiao Liu, Jingxi Che, Dian Yu, Jordan Boyd-Graber, Haitao Mi, and Dong Yu. Self-rewarding vision-language model via reasoning decomposition, 2026. URL https://arxiv.org/abs/2508.19652.

Chaohu Liu, Shida Wang, Yubo Wang, and Linli Xu. Tablemix: Enhancing multimodal table reasoning in mllms from a data-centric perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 33553–33565, June 2026.

Ahmed Nassar, Nikolaos Livathinos, Maksym Lysak, and Peter Staar. TableFormer: Table structure understanding with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4614–4623, June 2022.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1470–1480. Association for Computational Linguistics, 2015. URL https://aclanthology.org/P15-1142/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Brandon Smock, Rohith Pesala, and Robin Abraham. PubTables-1M: Towards comprehensive table extraction from unstructured documents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4634–4642, June 2022.

Brandon Smock, Rohith Pesala, and Robin Abraham. Aligning benchmark datasets for table structure recognition. In *Document Analysis and Recognition – ICDAR 2023*, volume 14191 of *Lecture Notes in Computer Science*, pp. 371–386. Springer, 2023. doi: 10.1007/978-3-031-41734-4_23.

Zhengzhuo Xu, Bowen Qu, Yiyan Qi, Sinan Du, Chengjin Xu, Chun Yuan, and Jian Guo. ChartMoE: Mixture of diversely aligned expert connectors for chart understanding, 2025. URL https://arxiv.org/abs/2409.03277.

Bohao Yang, Yingji Zhang, Dong Liu, André Freitas, and Chenghua Lin. Does table source matter? benchmarking and improving multimodal scientific table understanding and reasoning, 2025. URL https://arxiv.org/abs/2501.13042.

Weichao Zhao, Hao Feng, Qi Liu, Jingqun Tang, Shu Wei, Binghong Wu, Lei Liao, Yongjie Ye, Hao Liu, Houqiang Li, and Can Huang. TabPedia: Towards comprehensive visual table understanding with concept synergy, 2024. URL https://arxiv.org/abs/2406.01326.

Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. Multimodal table understanding, 2024. URL https://arxiv.org/abs/2406.08100.

Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: Data, model, and evaluation, 2020. URL https://arxiv.org/abs/1911.10683.

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. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3277–3287, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.254. URL https://aclanthology.org/2021.acl-long.254.

Xun Zhu, Ying Hu, Fanbin Mo, Miao Li, and Ji Wu. Uni-Med: A unified medical generalist foundation model for multi-task learning via Connector-MoE, 2024. URL https://arxiv.org/abs/2409.17508.

A SPATIAL ROUTING: PROTOCOL AND EXAMPLES

The analysis uses the Multi-GRPO epoch-1 checkpoint and a single shared manifest containing 100 development questions per dataset. The case selection metadata records readability and question heuristics applied before outcomes. TabMWP uses a readable arithmetic question; WTQ uses a comparison candidate; HiTab falls back to readability because the preferred structural criterion was unavailable. The HiTab example is a direct retrieval question, not evidence of hierarchical reasoning.

For Qwen2.5-VL, routing traces follow the vision encoder's window order. The saved window-index permutation maps each merger token back to its raster location. We verified that the permutation is complete, the footprints cover the original image dimensions after resizing, each token selects two distinct experts, and the saved mixing matrix reconstructs from the selected weights. The selected weights also agree with sigmoid score renormalization in Eq. 2, within float16 precision. Across all 725,256 tokens, there are exactly two nonzero coefficients per token and a maximum weight-reconstruction error of 2.45×10^{-4}. The 1st and 99th percentiles of active weights are 0.4607 and 0.5396. These statistics pool tokens and therefore weight larger images more heavily than the per-question means in Table 4. The overlays cover the full image canvas, including white background; background activation is not evidence of localization to table cells.

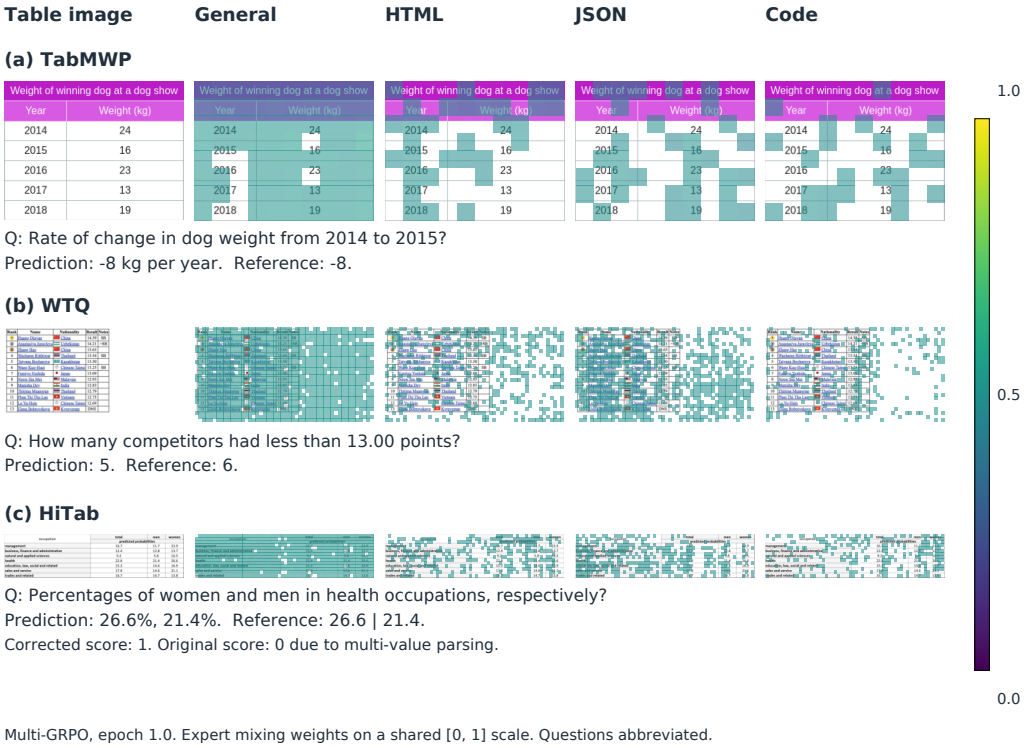


Figure 4: **All three preselected routing cases.** A shared $[0, 1]$ scale shows near-equal active weights; transparent positions are unselected. HiTab receives score 1 after the parser fix.

Case outcomes and answer parsing. The selected TabMWP question (ID `tabmwp:1018`) produces "-8 kg per year" against reference -8 and receives score 1. WTQ (ID `nt-10021`) produces 5 against reference 6 and receives score 0. HiTab (table 981) produces "26.6%, 21.4%" against reference values 26.6 and 21.4. The original scorer assigned this answer zero, whereas the equivalent JSON list received one. We reproduced the discrepancy without model inference: the parser did not split the plain multi-value string before normalization, which then removed its decimal points.

We adopt a parser correction that separates plain multi-value answers before normalization, retaining the existing unordered-set matching and containment fallback. This is not an ordered matcher: the word "respectively" does not introduce an order constraint in this protocol. The TQA implemen-

tation uses normalized string matching rather than a numerical relative-error tolerance; percentage values are not automatically converted between 26.6 and 0.266. The patch fixes the observed representation inconsistency, not every limitation of the scorer.

All 3,300 saved predictions are rescored with the same implementation. Exactly seven scores change from zero to one, all for this HiTab question under different routing conditions. Learned Top-2 HiTab accuracy rises from 38% to 39%, and Macro from 59.33% to 59.67%. The General+JSON score remains 58.67%; the learned-minus-fixed gap therefore changes from 0.67 to 1.00 points. These are automatic rescoring results, with no manual label overrides or removed questions. The correction leaves routing maps, expert usage, model weights, and cached responses unchanged. Offline rescoring reproduces all 3,300 original and corrected scores.

The correction is applied to cached intervention answers without retraining or new generation. It therefore leaves historical training rewards unchanged. Main benchmarks, TableVQA-Bench, Dev392, and intermediate reward-baseline scores retain the original protocol because their raw predictions were unavailable for rescoring. Both parsers reproduce the 1,000 final reward-baseline scores (Appendix C). All 300 routing traces are retained, including five truncated full-model responses; there are no recorded inference errors.

B Intervention Protocol and Uncertainty

The fixed-pair and exclusion experiments reuse the full-model answers from the spatial analysis. Checkpoint and manifest hashes agree across the experiments. There are 300 full-model answers, 1,800 new fixed-pair answers, and 1,200 exclusion answers. All conditions use batch size one, bfloat16 inference, greedy decoding, and a 1,024-token generation limit. The diagnostic manifest was sampled from `data/dev` with seed 42, independently of the three illustrative case choices. Each condition has one prediction for each manifest ID. There are no recorded inference errors. Saved route traces satisfy the two-distinct-expert rule, the imposed pair or exclusion constraint, and normalized sigmoid-pair fusion. Statistics and intervals below are computed from these cached predictions and traces.

Three different usage measures. Table 4 gives fusion-weight shares and General coverage. Figure 3(a) instead divides selection coverage by two: an expert selected at every token occupies 50% of the available slots. Weight shares, coverage, and selection shares should not be interpreted interchangeably. All statistics first average over an image's tokens and then over questions, so larger images do not receive more weight.

Table 4: **Expert usage on 300 development questions** (100 per dataset). Expert columns give mean fusion-weight shares (%). General coverage is the token-selection rate (%); the final column counts images using all six expert pairs. Aggregation is defined in Eq. 5.

Dataset	General	HTML	JSON	Code	General coverage	All six pairs
TabMWP	48.70	16.95	22.49	11.86	94.25	30/100
WTQ	46.85	15.93	23.36	13.87	91.69	100/100
HiTab	46.63	14.26	22.57	16.54	91.20	100/100

Paired bootstrap. Diagnostic intervals use corrected per-question scores, 2,000 paired bootstrap replicates, and NumPy `default_rng(42)`. Within each dataset, each replicate samples its original number of distinct table IDs with replacement, retaining all questions attached to each sampled table. The same sampled tables are used for every condition. There are 100 table clusters in TabMWP, 100 in WTQ, and 79 in HiTab. We compute question-level accuracy within each resampled dataset and average the three dataset accuracies equally for Macro. Intervals are the 2.5th and 97.5th percentiles of paired differences. They reflect sampling uncertainty on this development subset, not training-seed variation, and receive no multiple-comparison or best-pair-selection correction.

Table 5: **Paired fixed-pair comparisons on Dev300.** Δ is learned minus fixed Macro (pp). Wins/losses count questions correct only under learned/fixed routing.

Fixed pair	Δ	95% interval	Wins	Losses
General + HTML	+3.33	[-0.59, 7.33]	24	14
General + JSON	+1.00	[-2.69, 4.94]	18	15
General + Code	+1.67	[-2.01, 5.31]	19	14
HTML + JSON	+11.67	[6.33, 17.07]	49	14
HTML + Code	+16.67	[10.68, 22.67]	70	20
JSON + Code	+4.67	[-0.33, 9.59]	36	22

Table 6: **Expert-exclusion effects.** Rescored accuracy and paired 95% intervals; Δ is full minus excluded accuracy (pp). Full accuracies are 81%, 59%, and 39%.

Dataset	Excluded	Accuracy (%)	Δ	95% interval
TabMWP	General	70.0	+11.0	[4.00, 19.00]
	HTML	83.0	-2.0	[-6.00, 2.00]
	JSON	81.0	+0.0	[-4.00, 4.00]
	Code	81.0	+0.0	[-3.00, 3.00]
WTQ	General	59.0	+0.0	[-8.00, 8.00]
	HTML	52.0	+7.0	[0.00, 14.00]
	JSON	55.0	+4.0	[-2.00, 10.00]
	Code	55.0	+4.0	[-3.00, 11.00]
HiTab	General	26.0	+13.0	[3.92, 22.92]
	HTML	36.0	+3.0	[-3.19, 9.89]
	JSON	41.0	-2.0	[-7.78, 3.34]
	Code	34.0	+5.0	[-1.02, 11.54]

Completion and score limitations. No condition has a recorded inference error. Truncations are retained in the denominators: the full model has 5; excluding General, HTML, JSON, and Code yields 17, 4, 6, and 7, respectively. The six fixed pairs in Table 3 order have 5, 4, 5, 20, 14, and 7 truncated responses. Differences therefore include the effects of routing on completion within the shared token budget. A zero net accuracy change can also conceal correct-to-wrong and wrong-to-correct transitions. All conditions use the same corrected parser described in Appendix A. The resulting intervals remain conditional on this scoring protocol; resampling does not remove metric bias.

C Alternative Reward Baselines

Controlled comparison. SC-RPO uses one deterministic route with reward R_0 and seven stochastic routes with rewards $R_{1:7}$, yielding $A_g^{\text{route}} = R_g - R_0$ for the stochastic routes. Stochastic-Mean retains the same rollout composition but uses $A_g^{\text{route}} = R_g - \bar{R}_{1:7}$. Neither divides the routing advantage by the group standard deviation; Stochastic-Mean is distinct from the default eight-stochastic-route Multi-GRPO configuration. The supplied configurations differ only in the routing-baseline setting and output paths. Both use seed 42, greedy language rollouts, and the same aligned initialization recipe. Training reaches 9,408 processed prompts out of 9,450 per epoch (99.56%). These are separate ablation runs, not the checkpoints in Table 1.

Validation protocol. We report deterministic Top-2 validation on WTQ, HiTab, and TAT-QA, and supplement it with TabMWP and TabFact. Each intended subset contains 100 questions: 60 from `data_lean/dev` and 40 from `data/dev`, selected with seed 42 after excluding training overlap. Macro_3 averages the first three tasks and Macro_5 all five. The initial SC-RPO evaluation has 140 questions per original task; we flag it in Table 8 and omit initialization from the accuracy curves. These subsets differ from the spatial-routing subset.

At 40.6% progress, the SC-RPO log contains a complete 300-question evaluation and a later 280-question evaluation. We use the former consistently: WTQ/HiTab/TAT-QA are 54/46/37%, giving Macro_3 45.67% and Macro_5 52.0%. The later evaluation reports 52.5/47/37% with only 80 WTQ questions. The supplied five-task summary mixed WTQ from the first evaluation with HiTab from the second; we correct that merge rather than treating it as a new evaluation. All macro scores below are recomputed from the corresponding task scores.

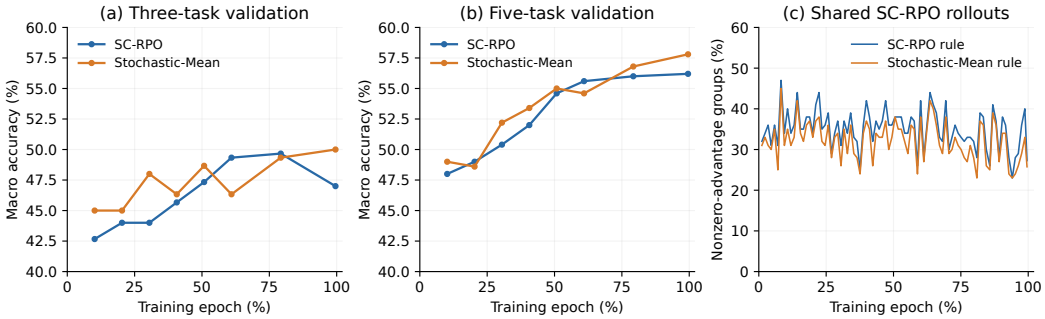


Figure 5: **Reward baselines through one training epoch.** (a,b) Three- and five-task validation accuracy after initialization. (c) Nonzero-advantage groups under both rules on shared SC-RPO rollouts.

Accuracy. At the final checkpoint, Stochastic-Mean obtains 50.0% Macro_3 versus 47.0% for SC-RPO, and 57.8% Macro_5 versus 56.2%. The five-task difference is eight correct answers out of 500. Stochastic-Mean leads on WTQ, TAT-QA, and TabMWP; SC-RPO leads on HiTab and TabFact. Intermediate rankings change: SC-RPO leads on Macro_3 at 61.0% and 79.2% progress, and on Macro_5 at 20.3% and 61.0%. Thus the endpoint favors Stochastic-Mean in this run, while the trajectory does not establish a uniformly better baseline. Paired endpoint predictions permit a direct comparison. Of 500 questions, both methods answer 265 correctly and 195 incorrectly; SC-RPO alone answers 16 correctly and Stochastic-Mean alone answers 24 (Table 7). The Stochastic-Mean minus SC-RPO gaps are $+3.00$ points on Macro_3 and $+1.60$ on Macro_5. Their paired 95% bootstrap intervals are $[-0.33, 6.33]$ and $[-0.97, 4.02]$ points, respectively. Both intervals include zero; they establish neither superiority nor equivalence. These estimates condition on the two checkpoints and do not measure training-seed variation.

Paired uncertainty. The final cache contains 500 unique, paired dataset–sample IDs per method, with 100 questions per task, raw responses, references, table identifiers, and checkpoint hashes. Prediction records agree with the evaluation outputs and data manifest. Offline evaluation reproduces all 1,000 original scores. Applying the corrected multi-value parser from Appendix A changes none of them, so the endpoint values remain comparable with the original-scoring curves. This is a targeted parser check; the existing unordered matching and containment fallback remain in the scorer. In particular, the implementation does not impose an answer-order constraint for "respectively."

We independently recompute uncertainty using the same resampling procedure as Appendix B: 2,000 paired replicates, NumPy `default_rng(42)`, sorted table IDs, and table-cluster resampling within each dataset. Each replicate samples the original number of table clusters with replacement, retains their questions, computes question-weighted accuracy within each task, and averages tasks equally. WTQ, HiTab, TAT-QA, TabMWP, and TabFact have 100, 82, 100, 100, and 100 distinct manifest table IDs, respectively. Macro_3 and Macro_5 use the same replicate draws; reported intervals are the 2.5th and 97.5th percentiles, without adjustment for multiple comparisons. These are endpoint intervals, not confidence bands for the trajectory. Intermediate checkpoints do not have complete raw-response caches.

Advantage activity. Panel (c) recomputes both rules on the same SC-RPO reward groups; it does not compare rollouts collected by the two trained policies. For each stochastic group we check whether any $|A_g| > 10^{-6}$. We retain the last logged occurrence of each (rank, step, sample ID), removing 192 duplicate records, and exclude 942 text-only records. This leaves 8,466 visual groups,

Table 7: **Paired outcomes at the final checkpoint.** SC denotes SC-RPO; SM denotes Stochastic-Mean.

Dataset	n	Both correct	SC only	SM only	Both wrong
WTQ	100	54	0	9	37
HiTab	100	38	7	4	51
TAT-QA	100	40	2	5	53
TabMWP	100	83	1	3	13
TabFact	100	50	6	3	41
All five	500	265	16	24	195

summarized in nonoverlapping windows of 100 (66 in the final window). The SC-RPO rule produces a nonzero advantage in 2,971 groups (35.09%), versus 2,722 (32.15%) for Stochastic-Mean. The 249 additional groups have equal stochastic rewards but a different deterministic reward: the stochastic-mean reference cancels their advantages, whereas R_0 does not. This algebraic difference describes nonzero advantages for the logged reward channel, not the total frequency of optimizer updates or their gradient quality. It does not establish an accuracy benefit. Indeed, the larger activity fraction does not translate into a higher final validation score here.

Table 8: **Reward-baseline validation accuracy (%).** Macro_3 averages WTQ, HiTab, and TAT-QA; Macro_5 includes all five tasks.

Epoch %	Method	WTQ	HiTab	TAT-QA	TabMWP	TabFact	M_3	M_5
0.0	SC†	54.3	44.3	36.4	59.0	53.0	45.0	49.4
0.0	SM	56.0	43.0	38.0	59.0	53.0	45.7	49.8
10.2	SC	53.0	43.0	32.0	60.0	52.0	42.7	48.0
10.2	SM	56.0	45.0	34.0	59.0	51.0	45.0	49.0
20.3	SC	52.0	43.0	37.0	61.0	52.0	44.0	49.0
20.3	SM	53.0	44.0	38.0	59.0	49.0	45.0	48.6
30.5	SC	51.0	44.0	37.0	67.0	53.0	44.0	50.4
30.5	SM	58.0	50.0	36.0	64.0	53.0	48.0	52.2
40.6	SC‡	54.0	46.0	37.0	72.0	51.0	45.7	52.0
40.6	SM	55.0	45.0	39.0	72.0	56.0	46.3	53.4
50.8	SC	54.0	48.0	40.0	75.0	56.0	47.3	54.6
50.8	SM	58.0	49.0	39.0	75.0	54.0	48.7	55.0
61.0	SC	60.0	48.0	40.0	78.0	52.0	49.3	55.6
61.0	SM	55.0	44.0	40.0	78.0	56.0	46.3	54.6
79.2	SC	56.0	49.0	44.0	79.0	52.0	49.7	56.0
79.2	SM	59.0	49.0	40.0	81.0	55.0	49.3	56.8
99.6	SC	54.0	45.0	42.0	84.0	56.0	47.0	56.2
99.6	SM	63.0	42.0	45.0	86.0	53.0	50.0	57.8

SC: SC-RPO; SM: Stochastic-Mean. Each task has $n = 100$ except †SC at initialization, where WTQ, HiTab, and TAT-QA have $n = 140$ each. ‡Uses the complete 300-question evaluation described above. Epoch progress is computed from processed prompts; the nominal final checkpoint is at 99.56% of one epoch. Both final rows are reproduced from 500 per-question records per method; all 1,000 scores are unchanged by the corrected parser.

D Development-Set Comparison

Table 9 records the development results separately from both the benchmark table and the spatial-routing subset. There are eight accuracy datasets and no FeTaQA in this split. Micro pools all questions; Macro averages the eight dataset scores equally. Excluding MetaMath and TablePerception leaves six sets of 60 questions, for which the two averages coincide. Soft-SFT uses dense softmax mixing, its best epoch-2 checkpoint, and sampling with temperature 0.1, `top_p`=0.8 and `top_k`=20. Both RL variants use epoch-1 checkpoints, deterministic Top-2 routing and greedy decoding. All runs use `max_new_tokens`=1024. The results are from a single run; paired predictions

and multiple training seeds would be needed to assess the uncertainty of the difference between the RL variants.

Table 9: **Development accuracy (%) on 392 questions.** Macro averages eight datasets. The last row excludes MetaMath and TablePerception; the remaining six datasets are equally sized.

Dataset	n	Soft-SFT	SC-RPO	Multi-GRPO
HiTab	60	45.0	55.0	48.3
InfoTabs	60	26.7	33.3	38.3
MetaMath	20	30.0	55.0	55.0
TAT-QA	60	43.3	48.3	51.7
TabFact	60	58.3	61.7	60.0
TabMWP	60	70.0	85.0	88.3
TablePerception	12	33.3	25.0	33.3
WTQ	60	45.0	51.7	53.3
Micro	392	46.7	54.8	55.9
Macro	–	44.0	51.9	53.5
Six-set Micro / Macro	360	48.1	55.8	56.7

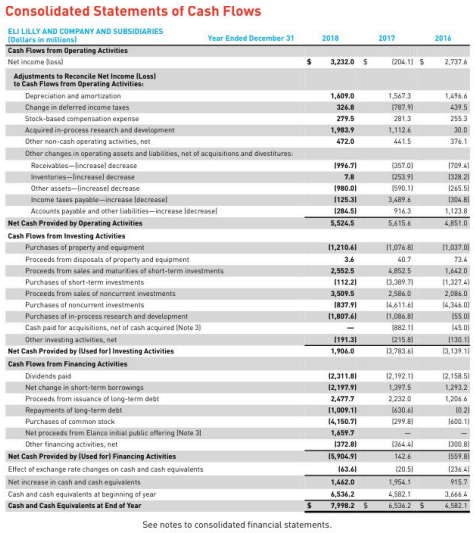
E Expert Alignment Details

Table 10 reports the retained image–target pairs. These are task-specific pair counts, not the number of unique source tables. Figure 6 illustrates the table-redrawing target used for code alignment.

Table 10: **Training data statistics for expert alignment.** Counts denote image–target pairs for each structured representation task.

Dataset	**Table2Code**	**Table2HTML**	**Table2JSON**
FinTabNet.c	46,998	90,631	90,631
PubTabNet	95,350	110,853	110,853
PubTables-1M	59,247	78,722	78,722
SynthTabNet	61,521	79,679	79,679
TableBank	1,424	13,007	13,007
Total	**264,540**	**372,892**	**372,892**

What the code target represents. The code target expresses table appearance as a drawing program: cell text and coordinates, background fills, and ruling lines are represented explicitly. Figure 7 illustrates this representation with a rendered table and an abbreviated Matplotlib program. This illustrates the representation used for alignment; it is not an evaluation of programs predicted by the trained Code expert. The program reconstructs a table image rather than computing an answer to a downstream question. During table reasoning, the expert contributes visual embeddings to the selected mixture; no rendering program is executed to produce the answer.

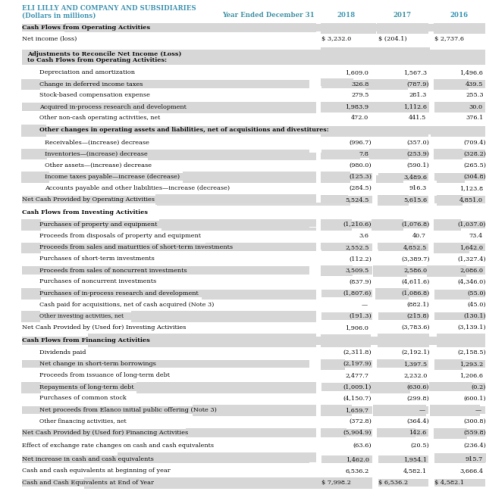


Consolidated Statements of Cash Flows

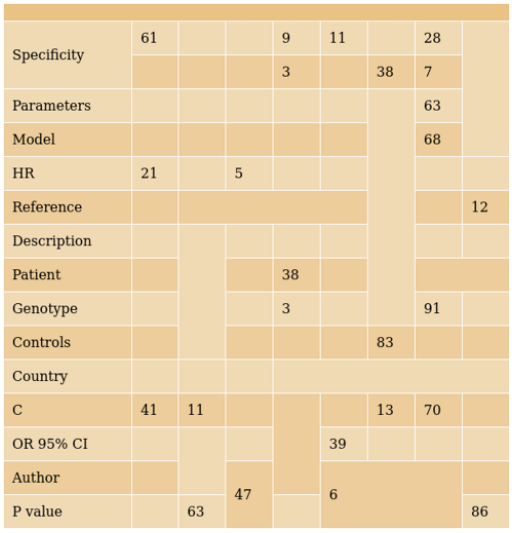
ELI LILLY AND COMPANY AND SUBSIDIARIES (Dollars in millions)	Year Ended December 31	2018	2017	2016
Cash Flows from Operating Activities				
Net income (loss)		$ 3,232.0	$ (204.1)	$ 2,737.6
Adjustments to Reconcile Net Income (Loss) to Cash Flows from Operating Activities:				
Depreciation and amortization		1,609.0	1,567.3	1,496.6
Change in deferred income taxes		326.8	(787.9)	439.5
Stock-based compensation expense		279.5	281.3	255.3
Acquired in-process research and development		1,983.9	1,112.6	30.0
Other non-cash operating activities, net		472.0	441.5	376.1
Other changes in operating assets and liabilities, net of acquisitions and divestitures:				
Receivables—(increase) decrease		(996.7)	(357.0)	(709.4)
Inventories—(increase) decrease		7.8	(253.9)	(328.2)
Other assets—(increase) decrease		(980.0)	(590.1)	(265.5)
Income taxes payable—increase (decrease)		(125.3)	3,489.6	(304.8)
Accounts payable and other liabilities—increase (decrease)		(284.5)	916.3	1,123.8
Net Cash Provided by Operating Activities		5,524.5	5,615.6	4,851.0
Cash Flows from Investing Activities				
Purchases of property and equipment		(1,210.6)	(1,076.8)	(1,037.0)
Proceeds from disposals of property and equipment		3.6	40.7	73.4
Proceeds from sales and maturities of short-term investments		2,552.5	4,852.5	1,642.0
Purchases of short-term investments		(112.2)	(3,389.7)	(1,327.4)
Proceeds from sales of noncurrent investments		3,509.5	2,586.0	2,086.0
Purchases of noncurrent investments		(837.9)	(4,611.6)	(4,346.0)
Purchases of in-process research and development		(1,807.6)	(1,086.8)	(55.0)
Cash paid for acquisitions, net of cash acquired (Note 3)		—	(882.1)	(45.0)
Other investing activities, net		(191.3)	(215.8)	(130.1)
Net Cash Provided by (Used for) Investing Activities		1,906.0	(3,783.6)	(3,139.1)
Cash Flows from Financing Activities				
Dividends paid		(2,311.8)	(2,192.1)	(2,158.5)
Net change in short-term borrowings		(2,197.9)	1,397.5	1,293.2
Proceeds from issuance of long-term debt		2,477.7	2,232.0	1,206.6
Repayments of long-term debt		(1,009.1)	(630.6)	(0.2)
Purchases of common stock		(4,150.7)	(299.8)	(600.1)
Net proceeds from Elanco initial public offering (Note 3)		1,659.7	—	—
Other financing activities, net		(372.8)	(364.4)	(300.8)
Net Cash Provided by (Used for) Financing Activities		(5,904.9)	142.6	(559.8)
Effect of exchange rate changes on cash and cash equivalents		(63.6)	(20.5)	(236.4)
Net increase in cash and cash equivalents		1,462.0	1,954.1	915.7
Cash and cash equivalents at beginning of year		6,536.2	4,582.1	3,666.4
Cash and Cash Equivalents at End of Year		$ 7,998.2	$ 6,536.2	$ 4,582.1

See notes to consolidated financial statements.

(a) Original table

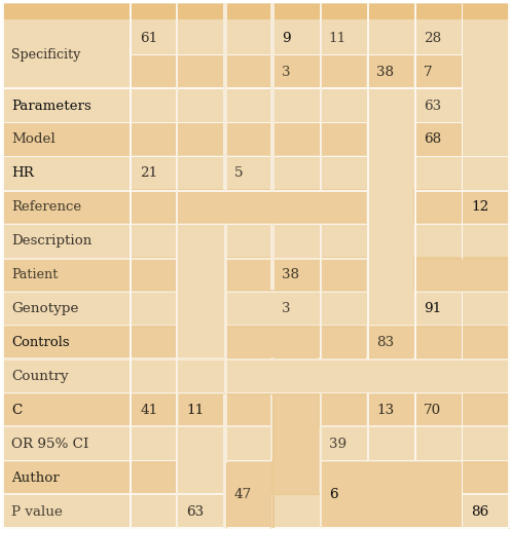


(b) Table replotted using Matplotlib code



Specificity	61			9	11		28	
				3		38	7	
Parameters							63	
Model							68	
HR	21		5					
Reference								12
Description								
Patient				38				
Genotype				3			91	
Controls						83		
Country								
C	41	11				13	70	
OR 95% CI					39			
Author			47		6			
P value		63						86

(c) Original table



Specificity	61			9	11		28	
				3		38	7	
Parameters							63	
Model							68	
HR	21		5					
Reference								12
Description								
Patient				38				
Genotype				3			91	
Controls						83		
Country								
C	41	11				13	70	
OR 95% CI					39			
Author			47		6			
P value		63						86

(d) Table replotted using Matplotlib code

Figure 6: **Table-to-code alignment examples.** Original table images (left) and their replotted counterparts obtained by executing the generated Matplotlib code (right), with examples from FinTabNet.c (top) and SynthTabNet (bottom).

Mean ± SD	Genotype	All	Method	Age	Baseline	P
OR	Model		P-value	df	OR 95% CI	48.60%
SNP			OR			No
Female		38			36	
Country	83		37			
Mean SD	78					18
Name	60		85		6	
Source			21	92		
95 % CI				14	85	
Variables		92				

```
import matplotlib.pyplot as plt
from matplotlib.patches import Rectangle
W, H, DPI = 512, 326, 100
CELLS = [  # (box, text, bold)
    ([9, 9, 92, 25], "Mean +/- SD", True),
    ([109, 9, 178, 25], "Genotype", True),
    ([196, 9, 216, 25], "All", True),
]  # +34 cells omitted
FILLS = [(2, 131, 510, 163, "#d3f1fb"), ...]
RULINGS = [(0, 0, 511, 0), (0, 98, 511, 98)]
ax = plt.gca(); ax.axis("off")
ax.set_xlim(0, W); ax.set_ylim(H, 0)
for x1, y1, x2, y2, c in FILLS:
    ax.add_patch(Rectangle(
        (x1, y1), x2-x1, y2-y1, fc=c))
for x1, y1, x2, y2 in RULINGS:
    ax.plot([x1, x2], [y1, y2], color="#282828")
for (x1, y1, x2, y2), text, bold in CELLS:
    ax.text(x1, (y1+y2)/2, text, va="center",
        fontweight="bold" if bold else "normal")
```

Figure 7: **Table-redrawing target example.** A rendered table and its abbreviated Matplotlib representation. Omitted entries make the displayed excerpt non-executable as written.

F Architecture and Training Objectives

Shapes. The vision width is 1280 with a 2×2 spatial merge, giving $\mathbf{x}_i \in \mathbb{R}^{5120}$. Each independent expert applies $\mathrm{Linear}(5120, 5120)$, GELU, then $\mathrm{Linear}(5120, 3584)$. Router logits are $z_i \in \mathbb{R}^4$. Sampling probabilities p_i are a softmax over experts. An ordered route id tensor has shape $[B, G, V, 2]$; the stored without-replacement log-probability has shape $[B, G_{\text{sampled}}, V]$. For Multi-GRPO, $G = G_{\text{sampled}} = 8$. For SC-RPO and Stochastic-Mean, $G = 8$ and $G_{\text{sampled}} = 7$, because the deterministic row is concatenated for generation but omitted from the gate action. Fusion weights have the same layout as the selected ids and sum to one on the last dimension. Expert outputs are $[B, V, 4, 3584]$ before the selected pair is mixed to $[B, V, 3584]$, or $[B, G, V, 3584]$ when several routes are replayed. Qwen2.5-VL traces follow window order. The saved permutation maps each merger token back to its raster footprint (Appendix A).

Normalization before spatial packing. The Qwen connector hook preserves the pretrained merger's input normalization. For encoder features $H \in \mathbb{R}^{L \times 1280}$ in the backbone's merge order, the implementation computes

$$X = \mathrm{reshape}(\mathrm{ln_q}(H), (L/4, 5120)) .$$

Thus, normalization operates on each 1280-dimensional feature before four features are packed into a connector token. The resulting tensor is reshaped for batched route replay and passed to the expert mixture. Normalizing an already packed vector would change this computation. This ordering preserves the input normalization used by the pretrained visual merger.

What is optimized. Table 11 records requires-gradient ownership in the RL trainer. Alignment puts connector parameters in one AdamW optimizer. During RL the vision encoder and base LLM stay frozen. The query gate is not in the optimizer when `force_alpha_zero` is true.

Table 11: **Parameter ownership.** "Step" means the optimizer includes the parameter and a nonzero advantage can produce a gradient. Text-only examples contribute no local RL gradient.

Module	Alignment	Router step	Policy step
Vision encoder	frozen	frozen	frozen
Base LLM	not stepped	frozen	frozen
Query gate	–	excluded	excluded
Visual router	random init	stepped	frozen
Experts	stepped	frozen	stepped
LoRA	not used	frozen	stepped

F.1 Reward computation and optimizer updates

We specify the implemented reward functions and optimization objectives. Run provenance and settings are separated in Appendices G and I.

Rollouts. For an image question, the trainer draws $G = 8$ routes and $M = 1$ response per route. Image-route language decoding is greedy, so exploration comes from the connector. Temperature 0.6 and top-p 0.95 are unused while sampling is off. Multi-GRPO's eight routes are all without-replacement multinomial draws. SC-RPO and Stochastic-Mean instead use one deterministic Top-2 route plus seven multinomial routes. Each selected pair is mixed by Eq. 2. The route is fixed for the whole response.

Branch 1 is produced by the same model on the image and question. It is asked for `<evidence>`, `<think>`, and `<answer>`. Branch 2, used only for image examples, sees the question and the extracted evidence text and does not see the image. It is not an external judge. Text-only examples are rolled out with language sampling and group size 8, but they have no visual tokens. They contribute no local router or policy gradient, although their ranks still participate in synchronized optimizer steps.

Rewards. The answer reward y_1 is the same 0/1 match used at evaluation, except on FeTaQA and ToTTo, where training uses sentence BLEU-4 scaled to $[0, 1]$. Format reward is 1 when branch 1 has evidence, think, and answer in that order, $2/3$ when only think and answer are well formed, and 0 otherwise. Branch 2 format reward is 1 or 0. If branch 2 declares insufficient evidence, its answer reward is 0; otherwise it uses the same matcher. These four rewards are standardized separately. With $M = 1$, each is centered across the routes in the group and divided by the population standard deviation. If that deviation is at most 10^{-6}, the advantage is zero. Format advantages enter the response loss only. They do not enter the router loss.

Router update. Let R_g be the route's answer reward. Multi-GRPO standardizes all eight stochastic rewards with the population deviation $\sigma^2 = G^{-1} \sum_g (R_g - \bar{R})^2$ and $\epsilon = \epsilon_\sigma = 10^{-6}$:

$$\widehat{A}_g^{\mathrm{MG}} = \begin{cases} 0, & \sigma \le \epsilon_\sigma, \\ (R_g - \bar{R})/(\sigma + \epsilon), & \text{otherwise.} \end{cases} \tag{6}$$

SC-RPO drops the deterministic route from the gate loss and does not divide by σ:

$$\widehat{A}_g^{\mathrm{SC}} = R_g - R_0, \qquad g = 1, \dots, 7. \tag{7}$$

Stochastic-Mean keeps that same 1+7 rollout but changes only the router baseline,

$$\widehat{A}_g^{\mathrm{SM}} = R_g - \frac{1}{7} \sum_{j=1}^{7} R_j, \qquad g = 1, \dots, 7. \tag{8}$$

It is not Multi-GRPO. A second copy of the chosen rule is computed from branch-2 rewards and added with weight $\lambda_{\mathrm{vis}}^{\mathrm{route}} = 1$. The stored sampling-time log-probability $\log P^{\mathrm{old}}$ is the denominator. The current router is rescored on the same ordered pairs,

$$\rho^{\mathrm{route}} = \exp\big(\log P_\theta - \log P^{\mathrm{old}}\big), \tag{9}$$

and the clipped surrogate uses $\epsilon_{\mathrm{clip}} = 0.2$. The deterministic route is excluded from this gate loss and retained in the response loss. If all router advantages are numerically zero, the example contributes no gate gradient even when a format advantage is nonzero. Synchronized optimizer calls may still apply gradients from other accumulated examples.

Response update. Experts and LoRA are updated with the router frozen. Route weights are replayed under stop-gradient. For a response token log-probability ℓ, the implementation calls the clipped token loss with denominator $\mathrm{stopgrad}(\ell)$. The ratio therefore equals 1 in value, while

$$\frac{\partial}{\partial \ell} \exp\big(\ell - \mathrm{stopgrad}(\ell)\big) = 1 \tag{10}$$

Table 12: **Route groups used by the three reported RL variants.** Inference for all three is deterministic Top-2 and greedy decoding.

	Multi-GRPO	SC-RPO	Stochastic-Mean
Rollout	8 multinomial	1 det. + 7 multinomial	1 det. + 7 multinomial
Router baseline	group mean	deterministic reward R_0	mean of 7 stochastic rewards
Divide by σ	yes, population	no	no
Routes in the gate loss	all 8	the 7 stochastic routes	the 7 stochastic routes
Routes in the token loss	all 8	all 8	all 8
Image language sampling	greedy	greedy	greedy

at that point. Because the ratio lies inside $[0.8, 1.2]$, clipping does not bind. For the minimized negative surrogate, the gradient in ℓ is $-A$, before token averaging and loss coefficients. This is an advantage-weighted current-log-probability update; greedy language rollouts do not justify treating it as an unbiased on-policy estimator. The route ratio in Eq. 9 is a different quantity: its old log-probability was stored at sampling time.

Answer, format, branch-2 answer, and branch-2 format advantages are normalized separately and weighted by 0.5, 0.05, 0.5, and 0.05. Within a completion the surrogate is a mean over tokens; completions in a branch are then combined in proportion to their token counts. A tied answer group can still update experts and LoRA through a nonzero format advantage. The group contributes no gate gradient if both branches have tied answer rewards.

Reference and schedule. The KL term is the sampled estimator

$$\widehat{\mathrm{KL}} = \exp(\ell_{\mathrm{ref}} - \ell) - (\ell_{\mathrm{ref}} - \ell) - 1, \tag{11}$$

averaged over response tokens, with weight 0.001 on each branch. ℓ_{ref} is stop-gradient. The reference swaps in the expert weights saved at the start of RL, before any resume, and disables LoRA. The reference remains fixed at initialization, distinct from the detached current probability used in the response ratio. The stored route is replayed rather than restoring the initial router.

Both optimizers use a constant learning rate of 5×10^{-6}. The router and response gradients are accumulated separately, with synchronized optimizer steps at each accumulation boundary.

Update order. For each image question the trainer (i) samples routes and responses without gradient and stores ordered ids, sigmoid weights, and sampling-time route log-probabilities; (ii) if any router advantage is nonzero, rescores those stochastic pairs, backpropagates the clipped gate loss, and accumulates router gradients; (iii) if any token advantage, including format, is nonzero, replays the stored routes with stopped fusion weights, backpropagates the token loss and sampled KL, and accumulates expert and LoRA gradients. Micro-batches accumulate with scale $1/(B_{\mathrm{micro}}K)$. The optimizer steps on the last question of every K micro-batches. A rank with no active loss still enters the synchronized step so that data-parallel ranks stay aligned. One optimizer step is therefore router-then-policy, not a single combined backward.

Configuration matrix. Table 12 separates the three router advantages. Language GroupZ for the response loss always covers every route in the rollout, including the deterministic route when one is present. Stochastic-Mean changes the router baseline only.

G Run Configuration

Both epoch-1 runs use Qwen2.5-VL-7B-Instruct, seed 42, LoRA $r = 64$, $\alpha = 128$, dropout 0, four experts, and the same aligned connector. Table 13 reports the applied learning rates and effective batches. The optimizer is AdamW with gradient clipping at 1.0 and no learning-rate schedule. The SC-RPO run records $\beta = (0.9, 0.999)$, weight decay 0, bfloat16, FlashAttention-2, and gradient checkpointing. Multi-GRPO uses the same optimizer constructor and records gradient checkpointing; its dtype and attention-kernel overrides are unspecified.

Table 13: **Reported epoch-1 RL configuration.** Learning rates follow the optimizer constructor. Effective batch counts questions, not routes, and includes text-only rows without local gradients.

	Multi-GRPO	SC-RPO
Gate group	8 multinomial	1 deterministic + 7 multinomial
Router advantage	Eq. 6	Eq. 7
Trainable	router, experts, LoRA	router, experts, LoRA
Frozen	vision, base LLM, query gate	vision, base LLM, query gate
Router / policy LR	5×10^{-6} / 5×10^{-6}	5×10^{-6} / 5×10^{-6}
Micro-batch $\times$ accum. $\times$ GPUs	$2 \times 16 \times 4$	$2 \times 16 \times 7$
Effective batch	128 questions	224 questions
Logged optimizer steps	74	44
Seed / LoRA	42 / r=64, α=128	42 / r=64, α=128
Decode in the trainer	greedy, 1,024 new tokens	greedy, 1,024 new tokens
Hardware	4$\times$ NVIDIA H800 PCIe 80GB	7$\times$ A800-80GB
Wall clock	48 h	40 h
Checkpoint	epoch 1	epoch 1

The lean training set contains 9,450 questions: WTQ 1,612, TabMWP 1,479, MetaMath 945, TabFact 1,100, InfoTabs 1,000, HiTab 700, FEVEROUS 400, FeTaQA 400, TAT-QA 350, TablePerception 472, MMathCoT-1M 292, FinQA 200, ToTTo 200, HybridQA 150, and MultiModalQA 150. Of these, 8,505 have images and 945 are text-only. The distributed samplers process 9,448 questions for Multi-GRPO and 9,450 for SC-RPO per epoch. Training was configured for up to 20 epochs with early stopping on validation macro accuracy (patience 3); the reported checkpoints are from epoch 1. Development results are reported once, in Appendix D.

Stochastic-Mean belongs to the separate reward-baseline experiment in Appendix C. Its configuration uses a micro-batch of four; the hardware and batch settings of the benchmark runs should not be assumed for this ablation.

H Scoring Protocol and the Parser Patch

Training answer rewards and reported accuracy call the same matcher. TQA normalizes case, articles, punctuation, and comma-grouped numbers, then accepts either exact set equality or a containment fallback. There is no relative numeric tolerance. Set equality ignores order, so a "respectively" answer is not required to preserve gold order. TabFact maps both sides through a true/false lexicon. InfoTabs buckets entailment, contradiction, and neutral. FeTaQA and ToTTo are BLEU-4 in the metric code; the training reward uses sentence BLEU-4 divided by 100. Corpus BLEU in that code is SacreBLEU. This submission does not add a TableMoE FeTaQA score. Failed parses score 0 and stay in the accuracy denominator.

The corrected parser changes multi-value extraction. On the 3,300 diagnostic predictions, 7 scores change and all 7 are the same HiTab example. Routing weights are untouched. The six-benchmark table, TableVQA-Bench, Dev392, and intermediate reward-baseline points were not recomputed with that parser. Offline scoring of the 1,000 final reward-baseline answers matches the original scores under both parsers. Historical training rewards may still contain the old parse. The exact six-benchmark sample counts and evaluation command are not recorded in the available run notes.

TableVQA-Bench checkpoint provenance. For this evaluation, Det-SFT uses the v4.1 MoCE epoch-1 checkpoint selected on development performance, while Soft-SFT uses the corresponding dense softmax recipe and its best epoch-2 checkpoint. SC-RPO uses the v4.1 `rl_lean` epoch-1 checkpoint with deterministic Top-2 routing and greedy decoding. The reported averages give each of the four accuracy scores equal weight. Checkpoint ages differ across training configurations, so these results compare the evaluated recipes rather than a fixed-budget optimization experiment.

I REPRODUCIBILITY

Appendices F and G specify route sampling, reward definitions, parameter updates, and training settings. The available records omit a training-source git revision and the exact six-benchmark evaluation command. The source snapshot documents the implemented objectives but does not establish the code version of every historical checkpoint. The Stochastic-Mean experiment is a separate reward-baseline run, not the Multi-GRPO benchmark checkpoint. SC-RPO resumed once during epoch 1, repeating samples 1,025–1,047 on one rank. This and the differing batch sizes limit comparisons of the two benchmark runs as optimization experiments.

J ARCHIVED FETAQA RESULTS

Table 14 preserves the available FeTaQA scores from the earlier comparison table. TableMoE, Det-SFT, and Soft-SFT do not have verified FeTaQA results in the current submission, so this task is excluded from the main comparison. These BLEU values are never included in the six-task accuracy average. Source protocols and BLEU implementations may differ; no new FeTaQA evaluation or rescoring was performed in this revision.

Table 14: **Archived FeTaQA BLEU scores.** Provenance symbols follow Table 1; scores are retained for reference and do not establish a TableMoE comparison.

Method	BLEU
Qwen2.5-VL-7B$^{\mathrm{TD}}$	10.99
Qwen2.5-VL-7B$^{\diamond}$	13.16
Gemma-4-E4B-it	4.97
Phi-4-Multimodal-Instruct	16.17
Qwen3-VL-8B-Instruct	4.86
Table-LLaVA-7B†	25.60
TabPedia-7B†	14.31
SynTab-LLaVA-7B†	35.45
HIPPO-8B†	33.18
HIPPO-8B*	19.74
TableDART (TableGPT2-7B + Ovis2-8B)†,‡	36.11

K LIMITATIONS

The results cover one backbone and one seed per RL configuration. Multi-GRPO and SC-RPO differ in route composition, normalization, and training budgets; their benchmark gap does not identify the effect of one design choice. The SFT comparisons likewise include changes to experts and language adapters, so they do not isolate router learning. TableVQA-Bench shares source families with training or alignment, and source-table overlap has not been ruled out.

The interventions cover one checkpoint and 300 development questions. Excluded-expert and fixed-pair models are not retrained, and the best fixed pair is chosen on the evaluation subset. Bootstrap intervals are unadjusted for selection and multiple comparisons and do not capture seed variation. Routing maps show fusion coefficients, not cell-level explanations of an answer. A small aggregate gap also leaves room for substantial changes in individual predictions.

Scoring remains sensitive to answer parsing and the containment rule. The corrected multi-value parser is used for the intervention results; main benchmarks, TableVQA-Bench, Dev392, and intermediate reward-baseline points retain the original scorer. Both parsers reproduce all 1,000 final reward-baseline scores, but this does not validate every aspect of the matcher. Appendix H documents the scope of the correction, and Appendix F specifies the implemented objectives and their differences from a lagged token-policy update.