SCOPE-ROUTER: COST-AWARE OPEN-SET VLM ROUTING FOR EXECUTION-ORIENTED TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Model routing aims to select the most suitable model from a candidate pool for each query, balancing quality and cost. Existing VLM routing research is limited to traditional VQA evaluation, lacks systematic calibration optimization for open-set scenarios, and employs training objectives that dilute multi-positive signals via softmax normalization without incorporating cost. We address these limitations with three contributions: (1) VLM-ExecRouterBench, the first execution-oriented VLM routing benchmark covering Code, Agentic, and Search domains with 11 candidate models spanning nearly two orders of magnitude in pricing; (2) SCOPE-Router, a dual-tower router that matches queries to model behavior profiles constructed via hybrid calibration (random/diagnostic/diversity sampling), enabling new models to join routing without retraining; (3) CRM+RCCR, an architecture-agnostic cost-aware objective that encodes cost preference into continuous relevance targets through per-pair independent scoring, eliminating multi-positive dilution while regularizing queries with similar routing preferences to be closer in the routing space. Empirically, SCOPE-Router achieves the best Rank Score on all three benchmarks, surpassing the runner-up by 1.84 points under OOD settings and by 6.75 points under doubly OOD open-set evaluation. When applied to four diverse routers, CRM+RCCR improves Rank Score by 1.25–6.21 points.

1 INTRODUCTION

The rapid advancement of LLMs and VLMs has given rise to a heterogeneous model ecosystem. From open-source models with fewer than 8B parameters to flagship commercial APIs, models differ substantially in capability, cost, and latency. No single model achieves optimal accuracy at minimal cost across all tasks (Hu et al., 2024; Huang et al., 2026; Lu et al., 2025), motivating model routing: selecting the most suitable model per query to balance quality and cost (Ong et al., 2025; Somerstep et al., 2025; Zhou et al., 2026). In practice, VLMs are increasingly deployed not through raw API calls for simple question answering, but within execution pipelines, including code generation, tool-calling agents, and multi-step web retrieval, where routing decisions directly impact both task success and deployment cost.

Recent years have seen progress in LLM/VLM routing. Methodologically, existing work spans classifier-based predictive routing, contrastive learning matching (Chen et al., 2024b), cluster-based generalization (Jitkrittum et al., 2025), and reinforcement learning orchestration (Qian et al., 2025). On the evaluation side, VL-RouterBench first extends routing evaluation to vision-language models, covering 14 datasets and 17 candidate VLMs; MMR-Bench (Ma et al., 2026) further introduces budget-aware multimodal reasoning evaluation. However, existing work faces limitations along three dimensions:

Limited domain coverage. Current VLM routing benchmarks cover only traditional VQA tasks—visual reasoning, OCR, chart understanding—where models directly output text answers. Yet VLMs are increasingly applied to tasks that require active execution: code generation demands precise program logic verified by unit tests, tool-calling tasks require multi-step reasoning with external tool interaction, and web search involves planning retrieval strategies and synthesizing multi-source information. These execution tasks impose different capability requirements and exhibit stronger inter-model complementarity, yet no routing benchmark incorporates them.

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.

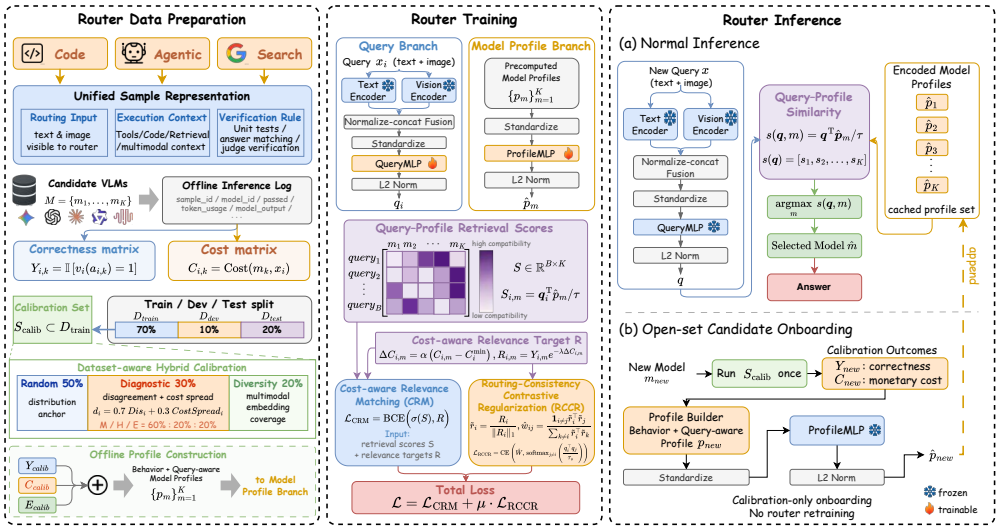


Figure 1: Overview of the proposed SCOPE-Router framework. Left: router data preparation constructs unified execution samples, correctness and cost matrices, and query-aware model profiles using the hybrid calibration set. Middle: during router training, the query and model-profile branches are projected into a shared routing space and jointly optimized using Cost-aware Relevance Matching (CRM) and Routing-Consistency Contrastive Regularization (RCCR). Right: during inference, the router selects the most suitable candidate according to query–profile similarity; a new model can be incorporated by running only the calibration set to construct and cache its profile, without retraining the router.

Insufficient open-set capability. New models emerge continuously in practice, but traditional routing methods treat models as fixed class labels, requiring retraining upon each addition. Existing open-set approaches (UniRoute, ICL-Router (Wang et al., 2025)) lack systematic optimization of calibration data selection: UniRoute randomly partitions validation data for model representation, while ICL-Router filters samples by a single signal. Under limited calibration budgets, the absence of targeted selection strategies leads to low discriminability in model profiles.

Training objectives decoupled from cost. RouterDC's contrastive objective partitions positive/negative examples based solely on correctness, ignoring cost differences; VL-RouterBench's softmax CE loss dilutes multi-positive signals through normalization; UniRoute only incorporates cost as a post-hoc inference penalty, leaving the training process itself cost-unaware.

To address these limitations, we make the following contributions (see Figure 1 for an overview):

(1) VLM-ExecRouterBench: the first execution-oriented VLM routing benchmark. We extend evaluation from traditional VQA to three execution domains: code generation (Code), tool-calling agents (Agentic), and multi-step web search (Search). Each sample is standardized into a Routing Input / Execution Context / Verification Rule triplet, uniformly covering three distinct execution protocols. The candidate pool comprises 11 models from five major providers, with output pricing spanning nearly two orders of magnitude.

(2) SCOPE-Router: an open-set dual-tower router with hybrid calibration. We formulate open-set routing as query-to-profile matching: for each model, we construct a Query-Aware Profile that fuses behavioral statistics and semantic directions, and compute matching scores via lightweight dual-tower MLPs in a shared routing space. New models need only run once on a small calibration set to generate a profile and immediately join routing, without retraining. To maximize profile discriminability under limited budgets, we design a hybrid calibration strategy allocating budgets at 50%/30%/20% to random sampling (distributional coverage), diagnostic sampling (driven by model disagreement and cost spread), and diversity sampling (uniform semantic coverage). The three strategies are complementary, with the full combination outperforming any single strategy.

(3) CRM+RCCR cost-aware training objective. We propose **C**ost-aware **R**elevance **M**atching (CRM), which defines a continuous relevance target that encodes cost preference directly into supervision and adopts per-pair independent sigmoid binary cross-entropy (BCE) in place of soft-

max cross-entropy (CE) to avoid multi-positive signal dilution. This is complemented by **R**outing-**C**onsistency **C**ontrastive **R**egularization (RCCR), which pulls queries with similar routing patterns closer in the routing space, enabling the router to generalize across queries that share model suitability. The objective is architecture-agnostic and can directly replace existing router losses. Extensive experiments on RouterDC, ZOOTER (Lu et al., 2023), CosineCls, and VLC all show consistent Rank Score improvements without any structural modification.

We conduct systematic experiments on VLM-ExecRouterBench, VL-RouterBench, and MMR-Bench. SCOPE-Router achieves the best Rank Score on all three benchmarks and maintains its lead under OOD generalization settings and open-set evaluation. Ablation studies verify the complementarity of CRM and RCCR, the effectiveness of the hybrid calibration strategy, and the cross-architecture transferability of the proposed training objective.

2 METHOD

2.1 PROBLEM FORMULATION

Model routing comprises three stages: data construction, router training, and routing inference.

Data Construction. Given task queries $\{x_i\}_{i=1}^{N}$ (text and/or images) and a candidate VLM pool $\mathcal{M} = \{m_1, \ldots, m_K\}$, we execute every model on every sample, yielding a correctness matrix $Y \in \{0,1\}^{N\times K}$ ($Y_{i,m} = 1$ if model m answers sample i correctly) and a cost matrix $C \in \mathbb{R}_{+}^{N\times K}$ ($C_{i,m}$ is the inference cost computed from token usage and per-million-token pricing).

Router Training. Given Y and C, the router learns a scoring function $s_\theta(x, m)\colon \mathcal{X} \times \mathcal{M} \to \mathbb{R}$. High-scoring models should be both likely correct and relatively cheap. The training objective must encode quality and cost preferences.

Routing Inference. Given a new query x_i, the router selects $\hat{m}_i = \arg\max_{m_k\in\mathcal{M}} s_\theta(x_i, m_k)$. Evaluation uses the *Rank Score*:

$$S_\beta = \frac{(1+\beta)\cdot A\cdot \widetilde{C}}{\beta\cdot A + \widetilde{C}}, \tag{1}$$

where A is routing accuracy, $\widetilde{C} = \frac{\log_2 c_{\max} - \log_2 \text{cost}}{\log_2 c_{\max} - \log_2 c_{\min}}$ is the log-normalized cost score, and β weights accuracy vs. cost (details in the Evaluation Metrics paragraph).

2.2 DATA GENERATION

Existing VLM routing benchmarks draw exclusively from traditional VQA-style datasets. We extend evaluation to execution-oriented domains (Code, Agentic, Search) by generating standardized samples from existing domain-specific sources (see the Datasets section for detailed dataset descriptions and Appendix section A for data-generation procedures and implementation details).

Each sample is converted into a three-part structure: (1) **Routing Input**: the query visible to the router; (2) **Execution Context**: the full context given to each candidate VLM; (3) **Verification Rule**: the validation logic that judges model output. This design separates what the router sees from what the model executes.

Matrix Construction. For each generated sample x_i, all models execute the Execution Context and the Verification Rule judges the results, producing the correctness matrix Y and cost matrix C.

Data Splits. The full dataset is partitioned 70%/10%/20%: training, validation, and held-out test.

2.3 OPEN-SET ROUTING AND HYBRID CALIBRATION SETS

Open-set routing enables new models to integrate without retraining. Instead of treating models as fixed class labels, we build a *behavioral profile vector* for each model. The router learns to match queries to profiles—a new model need only be executed once on a small calibration set to generate its profile and immediately participate in routing.

2.3.1 HYBRID CALIBRATION SET SELECTION

The calibration set $\mathcal{S}_{\text{calib}} \subset \mathcal{S}_{\text{train}}$ ($|\mathcal{S}_{\text{calib}}| = 1024$) is selected via a hybrid strategy allocating the budget at a 50%/30%/20% ratio across three complementary mechanisms, executed sequentially.

Random Sampling (50%). Uniform sampling without replacement, ensuring unbiased coverage of the training distribution.

Diagnostic Sampling (30%). From remaining samples, this component maximizes discriminative power. For each candidate sample i:

$$d_i = w_{\text{dis}} \cdot \underbrace{4\,p_i(1-p_i)}_{\text{disagreement}} + w_{\text{cost}} \cdot \underbrace{\text{cost_spread}_i^{\text{norm}}}_{\text{cost spread}}, \tag{2}$$

where $p_i = \frac{1}{K}\sum_{m=1}^{K} Y_{i,m}$ is the fraction of models in the reference pool $\mathcal{M}$ answering correctly ($w_{\text{dis}} = 0.7$, $w_{\text{cost}} = 0.3$). The disagreement term peaks at $p_i = 0.5$ (maximum inter-model disagreement) and equals zero when all models agree. The cost spread is defined as the difference between the maximum and minimum inference cost among models that answer correctly, normalized by the 95th percentile across all samples for robustness. Samples are stratified into difficulty buckets (hard $p_i \leq 0.25$ / medium $(0.25, 0.75]$ / easy > 0.75, allocated at 20%/60%/20%) and drawn with weights proportional to $\exp(\hat{d}_i/\tau_{\text{samp}})$ within each bucket, where $\hat{d}_i$ is the min-max normalized score within each bucket ($\tau_{\text{samp}} = 0.6$).

Diversity Sampling (20%). From the remaining unselected samples, we apply MiniBatchKMeans clustering on their pre-computed query embeddings (text and vision encoder features fused via normalize-concat), with the number of clusters set equal to the target sample count. The sample closest to each cluster centroid is selected, ensuring uniform coverage of the embedding space. Empty clusters (rare) are filled by selecting the farthest points from the global centroid.

2.3.2 MODEL BEHAVIOR PROFILE CONSTRUCTION

Given the calibration set ($S = 1024$ samples), we construct a *Query-Aware Profile* $\mathbf{p}_m$ for each model m.

Behavioral Vector.

$$\mathbf{p}_m^{\text{behav}} = [\,\mathbf{y}_m;\ \tilde{\mathbf{c}}_m;\ \mathbf{y}_m \odot (1 - \tilde{\mathbf{c}}_m);\ \mathbf{s}_m\,], \tag{3}$$

where $\mathbf{y}_m \in \{0,1\}^S$ is per-sample correctness, $\tilde{\mathbf{c}}_m \in [0,1]^S$ is min-max normalized cost, the third term is the "value" vector (high weight for samples answered correctly at low cost), and $\mathbf{s}_m \in \mathbb{R}^8$ is a summary statistics vector containing accuracy, mean cost (raw and normalized), mean cost on correct/incorrect subsets, mean value score, and correct/incorrect fractions.

Semantic Vector. Query embeddings from the calibration set are L2-normalized, then aggregated by model performance:

$$\mathbf{p}_m^{\text{sem}} = [\,\bar{\mathbf{e}}_m^{+};\ \bar{\mathbf{e}}_m^{-};\ \bar{\mathbf{e}}_m^{v}\,], \tag{4}$$

where

$$\bar{\mathbf{e}}_m^{+} = \frac{\sum_i Y_{i,m} \cdot \hat{\mathbf{e}}_i}{\sum_i Y_{i,m}}, \quad \bar{\mathbf{e}}_m^{-} = \frac{\sum_i (1-Y_{i,m}) \cdot \hat{\mathbf{e}}_i}{\sum_i (1-Y_{i,m})},$$
$$\bar{\mathbf{e}}_m^{v} = \frac{\sum_i Y_{i,m}(1-\tilde{c}_{i,m}) \cdot \hat{\mathbf{e}}_i}{\sum_i Y_{i,m}(1-\tilde{c}_{i,m})}, \tag{5}$$

with $\hat{\mathbf{e}}_i = \mathbf{e}_i / \|\mathbf{e}_i\|_2$. The three directional vectors encode what the model is good at, bad at, and efficient at in semantic space.

The final profile is $\mathbf{p}_m = [\mathbf{p}_m^{\text{behav}};\ \mathbf{p}_m^{\text{sem}}] \in \mathbb{R}^{3S+8+3D}$, where D is the query embedding dimension (with default encoders BGE-M3 (Chen et al., 2024a) + DINOv2-large (Oquab et al., 2023), $D = 2048$).

2.4 ROUTER ARCHITECTURE

We propose **SCOPE-Router** (**S**elective **C**alibration for **O**pen-set **P**rofile-**E**nhanced Routing)—a dual-tower architecture projecting queries and model profiles into a shared routing space.

Query Encoder. Query x_i is encoded via text and vision encoders with normalize-concat fusion to form $\mathbf{e}_i$. After standardization, a lightweight QueryMLP projects to the routing space:

$$\mathbf{q}_i = \ell_2\big(\mathrm{QueryMLP}(\mathrm{standardize}(\mathbf{e}_i))\big), \tag{6}$$

where $\ell_2(\mathbf{v}) = \mathbf{v}/\|\mathbf{v}\|_2$ denotes L2 normalization.

Profile Encoder. Model profiles $\mathbf{p}_m$ are likewise standardized and projected via ProfileMLP:

$$\hat{\mathbf{p}}_m = \ell_2\big(\mathrm{ProfileMLP}(\mathrm{standardize}(\mathbf{p}_m))\big). \tag{7}$$

Both MLPs are 2-layer (hidden dimension 128 with ReLU and dropout; output dimension 64). L2-normalized outputs produce the matching score:

$$s_{i,m} = \mathbf{q}_i^\top \hat{\mathbf{p}}_m \,/\, \tau, \tag{8}$$

where τ is a temperature parameter. At inference, the router selects $\arg\max_m \, s_{i,m}$.

2.5 Cost-Aware Training Objective

Standard routing training faces two problems: (1) hard-label softmax discards information about other correct models; (2) cost preference is only applied post-hoc. We propose a unified solution.

Relevance Target. For each query–model pair:

$$R_{i,m} = \mathbb{1}[Y_{i,m} = 1] \cdot \exp\big(-\lambda \cdot \alpha \cdot (C_{i,m} - C_i^{\min})\big), \tag{9}$$

where $C_i^{\min} = \min_{m':Y_{i,m'}=1} C_{i,m'}$ is the cost of the cheapest correct model. The cheapest correct model receives $R = 1$, more expensive correct models receive $R \in (0, 1)$, and incorrect models receive $R = 0$. For all-failed samples ($\sum_m Y_{i,m} = 0$), we set $R_{i,:} = \mathbf{0}$; they contribute to CRM as all-negative examples but are excluded from RCCR via $\mathcal{U}$. Edge cases in diagnostic sampling and profile aggregation (empty correct/incorrect sets) default to zero.

CRM Loss. We predict each pair independently via binary cross-entropy rather than competing models via row-wise softmax:

$$\mathcal{L}_{\mathrm{CRM}} = \frac{1}{BK} \sum_{i=1}^{B} \sum_{m=1}^{K} \mathrm{BCE}\big(\sigma(s_{i,m}),\ R_{i,m}\big), \tag{10}$$

where $\sigma(\cdot)$ denotes the sigmoid function. Each pair's gradient is independent, so positive signals are not diluted in multi-positive scenarios.

Routing-Consistency Contrastive Regularization (RCCR). Let $\tilde{\mathbf{r}}_i = R_{i,:}/\|R_{i,:}\|_1$ be the L1-normalized relevance vector. We compute pairwise sample similarity $w_{ij} = \tilde{\mathbf{r}}_i^\top \tilde{\mathbf{r}}_j$ and row-normalize into $\hat{w}_{ij}$. Define $\mathcal{U} = \{i : \|R_{i,:}\|_1 > 0\}$ and $\mathcal{V} = \{i \in \mathcal{U} : \sum_{j \neq i} w_{ij} > 0\}$:

$$\mathcal{L}_{\mathrm{RCCR}} = -\frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \sum_{\substack{j \in \mathcal{U} \\ j \neq i}} \hat{w}_{ij} \log \frac{\exp(\mathbf{q}_i^\top \mathbf{q}_j / \tau_s)}{\sum_{\ell \in \mathcal{U},\, \ell \neq i} \exp(\mathbf{q}_i^\top \mathbf{q}_\ell / \tau_s)}, \tag{11}$$

where τ_s is a temperature parameter.

Total Loss.

$$\mathcal{L} = \mathcal{L}_{\mathrm{CRM}} + \mu \cdot \mathcal{L}_{\mathrm{RCCR}}, \tag{12}$$

with $\mu = 0.1$ by default. Since this objective operates solely on Y, C, and the router's output logits, it is architecture-agnostic and can be applied to any trainable router as a replacement loss.

3 VLM-ExecRouterBench

3.1 Datasets

Our routing benchmark spans three task domains with 34k samples (374k sample-model pairs) and 5.1B input-output tokens. All datasets use the unified Routing Input / Execution Context / Verification Rule structure (described in the Section A), split 70%/10%/20%.

Code: MBPP (Austin et al., 2021), BigCodeBench (Zhuo et al., 2025), APPS (Hendrycks et al., 2021), LiveCodeBench (Jain et al., 2024). Covers code synthesis from introductory to competition-level, verified by unit tests.

Agentic: MathVista (Lu et al., 2024), ChartQA (Masry et al., 2022), MMMU (Yue et al., 2024), OCRBench (Liu et al., 2024), DocVQA (Mathew et al., 2021), AI2D (Kembhavi et al., 2016), RealWorldQA (xAI, 2024). VQA/document tasks reformulated under an agentic setting requiring multi-step tool calls.

Search: BrowseComp-Plus (Chen et al., 2025). A deep-research benchmark derived from BrowseComp with a fixed, human-verified corpus. Models perform multi-step retrieval over the provided document collection to answer complex queries.

3.2 Candidate Model Pool

We select 11 representative VLMs as the candidate pool $\mathcal{M}$, spanning diverse capability tiers and cost ranges (Table 5), including Qwen3-VL-8B (Bai et al., 2025), Qwen3.5-35B-A3B (Qwen Team, 2026), Gemini 2.5/3/3.5 Flash series (Google DeepMind, 2025a;b; 2026), GPT-5.4 mini/5.4 (OpenAI, 2026a;b), Claude Haiku 4.5/Sonnet 4.6/Opus 4.6 (Anthropic, 2025; 2026b;a), and MiniMax M3 (Lai et al., 2026). Each model executes the full Execution Context on all samples, yielding $Y \in \{0,1\}^{N\times 11}$ and $C \in \mathbb{R}_+^{N\times 11}$.

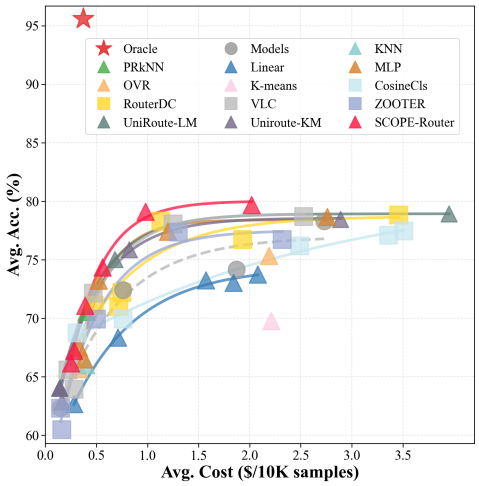


Figure 2: Performance in the accuracy–cost plane on VL-RouterBench. Gray circles represent individual candidate models. Colored markers represent operating points of the same router under different values of $\lambda \in \{1, 10, 100, 1000, 10000\}$, and the curves are Pareto fronts fitted to these points. Points closer to the upper-left corner indicate a better accuracy–cost trade-off.

3.3 Router Baselines

We evaluate three categories of routing methods (detailed descriptions in Appendix section C):

Training-free baselines: Oracle (theoretical upper bound; selects the cheapest correct model), Strongest (always selects the globally best-performing model), Cheapest (always selects the globally cheapest model).

Feature-level routers: Use embeddings from frozen encoders to make routing decisions via non-parametric methods, clustering, or lightweight classifiers. Includes KNN (Fix & Hodges, 1951), PRkNN, K-means (MacQueen, 1967), Linear (McFadden, 1974), MLP (Rumelhart et al., 1986), OVR (Rifkin & Klautau, 2004), and UniRoute. We adopt UniRoute as the open-set baseline because its routing mechanism is modality-agnostic; other open-set routers rely on text-only features and require non-trivial redesign for multimodal settings. Our **SCOPE-Router** also belongs to this category.

End-to-end routers: Learn the routing decision function end-to-end. Includes CosineCls, RouterDC, ZOOTER, and VLC.

3.4 Evaluation Metrics

For each router, we report Average Accuracy (fraction of test samples routed correctly), Average Cost (mean inference cost in \$/10K samples), and Rank Score S_β (Eq. (1), $\beta = 0.1$). Rank Score serves as the primary ranking criterion. Formal definitions of all metrics are provided in Appendix section D. We also sweep λ on VL-RouterBench and VLM-ExecRouterBench to examine the routers' accuracy–cost trade-offs (Figures 2 and 3). The results suggest that effective query–model matching is more valuable than increased expenditure.

Table 1: Performance of router methods on VLM-ExecRouterBench, VL-RouterBench, and MMR-Bench. Avg. Acc. denotes average accuracy (%), Avg. Cost denotes average cost ($/10K samples), and methods are ranked by Rank Score.

Router	VLM-ExecRouterBench				VL-RouterBench				MMR-Bench			
	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑	Rank ↓	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑	Rank ↓	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑	Rank ↓
Training-free Baselines												
Oracle	95.25	59.53	91.77	0	95.60	0.37	93.68	0	91.67	10.41	75.67	0
Strongest	**86.86**	308.24	66.75	13	78.01	2.72	68.88	11	**74.25**	81.30	50.64	14
Cheapest	67.74	**18.94**	69.78	12	62.43	**0.14**	64.63	14	42.25	**0.25**	43.20	15
End-to-End Routers												
CosineCls	79.17 ± 0.46	33.70 ± 3.73	$\underline{79.55 \pm 0.50}$	$\underline{2}$	70.24 ± 0.62	0.61 ± 0.16	69.90 ± 0.34	9	61.31 ± 0.58	7.37 ± 1.98	56.05 ± 0.77	7
RouterDC	79.11 ± 0.84	38.65 ± 5.04	79.16 ± 0.84	3	77.52 ± 1.04	1.04 ± 0.26	$\underline{74.59 \pm 1.05}$	$\underline{2}$	62.22 ± 0.88	7.33 ± 1.78	56.75 ± 0.89	6
ZOOTER	74.32 ± 1.23	$\underline{27.55 \pm 4.87}$	75.47 ± 1.20	9	74.65 ± 1.34	0.93 ± 0.15	72.58 ± 1.03	7	61.65 ± 1.21	3.62 ± 0.63	57.77 ± 1.02	5
VLC	75.83 ± 1.20	30.93 ± 0.80	76.66 ± 1.12	6	$\underline{78.09 \pm 1.17}$	1.23 ± 0.03	74.33 ± 0.51	3	65.06 ± 1.03	5.50 ± 0.13	59.59 ± 0.79	3
Feature-level Routers												
KNN	70.81	40.93	71.41	11	66.26	$\underline{0.38}$	67.13	12	58.66	7.07	54.11	10
PRkNN	75.10	40.72	75.36	10	70.68	0.41	71.09	8	59.72	7.30	54.85	9
OVR	85.27	407.16	55.14	15	75.49	2.18	68.92	10	69.01	55.43	51.98	12
K-means	$\underline{85.69}$	362.20	60.73	14	70.01	2.21	64.62	15	70.12	61.59	51.63	13
Linear	79.85 ± 1.11	101.15 ± 10.01	76.15 ± 1.07	8	68.32 ± 7.45	1.32 ± 0.33	65.68 ± 5.69	13	$\underline{70.60 \pm 1.05}$	53.76 ± 4.89	53.05 ± 0.93	11
MLP	75.88 ± 0.53	28.45 ± 3.55	76.87 ± 0.55	5	77.49 ± 0.56	1.13 ± 0.13	74.23 ± 0.22	4	60.88 ± 0.41	7.45 ± 0.89	55.69 ± 0.42	8
UniRoute-KM	76.57 ± 0.76	48.16 ± 13.13	76.26 ± 1.03	7	75.88 ± 0.73	0.82 ± 0.21	74.04 ± 1.08	5	60.88 ± 0.61	0.51 ± 0.14	$\underline{59.72 \pm 0.60}$	$\underline{2}$
UniRoute-LM	78.55 ± 1.61	42.58 ± 8.29	78.40 ± 1.55	4	75.14 ± 0.96	0.68 ± 0.14	73.86 ± 1.04	6	58.23 ± 1.16	$\underline{0.27 \pm 0.05}$	57.91 ± 1.05	4
SCOPE-Router (Ours)	81.65 ± 1.06	46.20 ± 11.62	$\mathbf{80.94} \pm 1.22$	**1**	$\mathbf{79.11} \pm 1.02$	0.98 ± 0.27	$\mathbf{76.18} \pm 1.44$	**1**	62.90 ± 0.81	0.65 ± 0.17	$\mathbf{61.23} \pm 0.75$	**1**

4 Experiments

4.1 Main Results

We evaluate all routing methods on three benchmarks: (1) **VLM-ExecRouterBench** (ours); (2) **VL-RouterBench**; and (3) **MMR-Bench** (Ma et al., 2026). For each benchmark, all routers are trained on its corresponding training split. We sweep the cost-sensitivity coefficient $\lambda \in \{0, 10, 100, 1000, 10000, +\infty\}$ for each method and report results at the λ achieving the best validation Rank Score. Training details and per-method hyperparameters are in Appendix section C.

Table 1 presents results across all three benchmarks. SCOPE-Router ranks first in Rank Score on all three benchmarks, the only method to achieve this across benchmarks spanning traditional VQA, execution-oriented tasks, and multimodal reasoning.

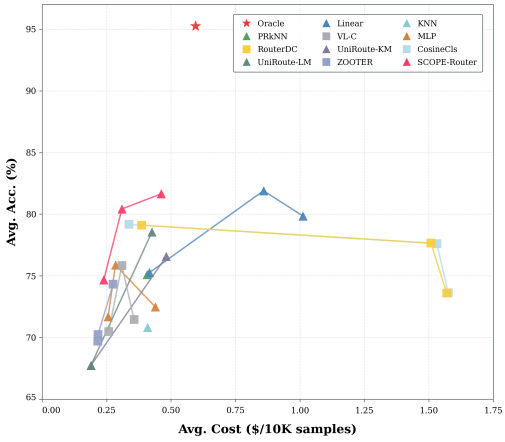


Figure 3: Accuracy–cost trade-off on VLM-ExecRouterBench. All methods except SCOPE-Router follow the experimental settings of VL-RouterBench (Huang et al., 2026). Connected points correspond to $\lambda \in \{10, 100, 1000\}$; costs are divided by 100 for visualization. OVR and K-means are omitted for readability. Complete numerical results are provided in Appendix Table 14.

On **VLM-ExecRouterBench**, SCOPE-Router obtains a Rank Score of 80.94 ± 1.22, followed by CosineCls (79.55 ± 0.50). Compared with the Strongest baseline, it reduces cost by 85% while sacrificing only 5.21pp in accuracy. The top end-to-end routers (CosineCls, RouterDC) outperform MLP on execution tasks—a pattern not observed on VL-RouterBench—suggesting that execution domains' higher inter-task heterogeneity benefits more expressive matching. A per-dataset breakdown (Appendix Tables 11–13) reveals that SCOPE-Router's overall advantage stems from consistent competitiveness across all three domains rather than dominance in any single one: it achieves the highest accuracy on Search (68.18%) and DocVQA (97.92%), while maintaining strong cost efficiency on Code and Agentic tasks where cost-unaware methods (OVR, K-means) achieve higher accuracy but at 3–8× the cost.

On **VL-RouterBench**, SCOPE-Router scores 76.18 ± 1.44 vs. the runner-up RouterDC (74.59 ± 1.05). The mean±std intervals overlap, so the ranking difference on this benchmark alone is not conclusive; however, SCOPE-Router simultaneously achieves the highest accuracy ($79.11 \pm 1.02\%$) and reduces cost by 64% vs. Strongest, indicating a favorable quality–cost tradeoff. Here MLP

(74.23±0.22) performs comparably to end-to-end routers (RouterDC 74.59, VLC 74.33), indicating that frozen encoders already provide strong discriminative features for traditional VQA tasks.

On **MMR-Bench**, SCOPE-Router achieves 61.23 ± 0.75, surpassing UniRoute-KM (59.72 ± 0.60) by 1.51 points with mean±std intervals that do not overlap, at merely 0.8% of the Strongest baseline's cost (0.65 vs. 81.30).

4.2 Ablation Studies

Loss Function Ablation. We ablate the training objective on VL-RouterBench (Table 2). CRM alone (w/o RCCR) achieves 75.50, a 1.0-point gain over pure softmax (74.51), confirming that independent per-model scoring avoids multi-positive dilution. RCCR is effective only under CRM: adding RCCR to softmax hurts performance (74.51 → 73.41), whereas adding it to CRM improves Rank Score from 75.50 to 76.18 (+0.68). This reveals that CRM's unnormalized scoring framework is a prerequisite for RCCR's distributional regularization to be beneficial—the two components are complementary rather than independently additive.

Generalizability of the Loss Function. CRM+RCCR applied as a replacement loss to RouterDC, ZOOTER, CosineCls, and VLC improves Rank Score on both benchmarks (Figure 4), with gains ranging from +1.25 to +6.21 points. Gains are larger on VLM-ExecRouterBench, where execution tasks' stronger inter-model complementarity amplifies the benefit of eliminating multi-positive dilution. This confirms the objective's cross-architecture generalizability.

Table 2: Loss function ablation on VL-RouterBench. RCCR denotes Routing-Consistency Contrastive Regularization.

Configuration	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑
Full model	**79.11**	0.98	**76.18**
w/o RCCR	77.90	0.89	75.50
w/o CRM	75.82	0.97	73.41
w/o CRM, RCCR	76.13	0.76	74.51

OOD Generalization. We select 5 datasets from VLM-ExecRouterBench (MBPP, LiveCodeBench, ChartQA, OCRBench, RealWorldQA) as OOD test sets, with the remaining data split 9:1 into training and validation sets (Appendix Table 6). All 11 models remain in the candidate pool; only the task distribution shifts. SCOPE-Router achieves the highest Rank Score (88.14), outperforming the runner-up K-means (86.30) by 1.84 points at only 21.8% of its cost. Its OOD score exceeds the in-distribution score (80.94), consistent with higher Oracle performance in the OOD set (97.15 vs. 91.77), indicating stronger inter-model complementarity in these held-out tasks.

Open-Set Routing. We conduct open-set experiments on VLM-ExecRouterBench by holding out 5 of 11 candidate models as unseen (Gemini 2.5 Flash Lite, MiniMax M3, GPT-5.4, Gemini 3.5 Flash, Claude Opus 4.6), excluded from profile construction and router training. At evaluation time, their profiles are generated from calibration samples and added to the candidate pool. On in-distribution data, SCOPE-Router achieves Rank Score 75.96 (Table 7), outperforming UniRoute-KM by 1.45 points. Under doubly OOD evaluation—unseen models on the same 5 held-out datasets (Table 7)—SCOPE-Router's advantage expands to +6.75 Rank Score and +7.26 accuracy over UniRoute-KM, demonstrating that profile-based matching is more robust than cluster-based methods under distribution shifts.

MLP Hidden Dimension. The optimum is 128 dimensions (Appendix Table 8), though scores span only 0.62 points across {64–4096}, indicating low sensitivity.

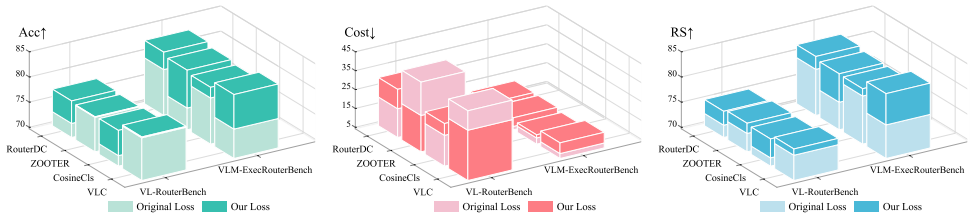


Figure 4: Transferability of our cost-aware loss across different end-to-end routers on VL-RouterBench and VLM-ExecRouterBench. Applying our loss consistently improves the overall routing performance across RouterDC, ZOOTER, CosineCls, and VLC.

Behavioral vs. Query-Aware Profiles. Adding calibration-query embeddings to behavioral profiles improves accuracy by 3.44 percentage points and Rank Score by 1.74 on VL-RouterBench (Table 3), while inference cost rises from 0.68 to 0.98. The higher Rank Score supports combining query semantics with behavioral features.

Table 3: Profile ablation on VL-RouterBench. Accuracy is in %; cost is in $/10K samples.

Profile	Acc. ↑	Cost ↓	Rank Score ↑
Behavior-only	75.67	**0.68**	74.44
Query-aware (Ours)	**79.11**	0.98	**76.18**

Freezing vs. Unfreezing the Query Embedder. Freezing the query embedder outperforms unfreezing on VL-RouterBench (76.18 vs. 75.68; Appendix Table 9) while reducing training cost; we adopt freezing as default.

Calibration Strategy Ablation. Among single strategies (Appendix Table 10), Diversity achieves the best Rank Score (75.69); the full triple combination achieves 76.18, confirming complementarity.

Calibration Set Size. On VL-RouterBench, 1,024 calibration samples achieve the highest Rank Score (76.18; Table 4), with no monotonic gains from larger sets. Using 128 samples, one eighth of the default set, lowers Rank Score by only 0.29, suggesting that compact calibration sets can still support effective routing.

Table 4: Calibration size on VL-RouterBench. Accuracy is in %; cost is in $/10K samples.

Size	Acc. ↑	Cost ↓	Rank Score ↑
128	78.56	0.93	75.89
256	77.94	0.87	75.62
512	78.24	0.90	75.75
1,024 (Ours)	**79.11**	0.98	**76.18**
2,048	78.03	**0.80**	75.97
4,096	78.01	0.93	75.42

Text and Vision Encoder Selection. Across 25 encoder combinations (Appendix Figure 7), Rank Scores span <0.8 points (75.54–76.33), confirming robustness to encoder choice.

5 Related Work

LLM Routing Benchmarks. RouterBench formulated LLM routing as cost–quality optimization. LLMRouterBench (Li et al., 2026) and RouterArena further expanded model pools and evaluation dimensions for text routing. VL-RouterBench extended the paradigm to VLMs with 14 datasets, 17 models, and a unified Rank Score metric. However, existing VLM benchmarks remain confined to traditional VQA tasks. Execution-based tasks such as code generation, tool-use agentic reasoning, and web retrieval exhibit stronger inter-model complementarity yet have not been incorporated into any routing benchmark. Our VLM-ExecRouterBench is the first to cover these domains.

Model Representation and Open-Set Routing. Early routers map queries to fixed model IDs, requiring retraining for new models. Recent open-set approaches use per-cluster error rates, anchor pass/fail patterns, or public-signal GNNs (Xu et al., 2026), but each sacrifices sample-level granularity, cost information, or behavioral fidelity. Our profiles are derived from first-hand inference on a carefully designed hybrid calibration set, fusing per-sample correctness, cost, and semantic directions to capture both what a model can solve and at what price.

Router Training Objectives. RouterDC uses dual contrastive losses but partitions positives purely by correctness. VL-RouterBench pairs cost-weighted soft labels with softmax CE, but normalization dilutes gradients when multiple models answer correctly. UniRoute treats cost only at inference time. We propose CRM+RCCR: Cost-aware Relevance Matching encodes cost preference into continuous relevance targets and eliminates multi-positive dilution, while Routing-Consistency Contrastive Regularization pulls queries with similar routing patterns closer in the routing space, enabling the router to benefit from shared model-suitability structure.

6 Conclusion

We present VLM-ExecRouterBench, SCOPE-Router, and CRM+RCCR, advancing VLM routing along evaluation coverage, open-set capability, and cost-aware training. SCOPE-Router achieves the best Rank Score on all three benchmarks, leads by 1.84 points under OOD and 6.75 points under doubly OOD open-set evaluation. CRM+RCCR consistently improves four architecturally diverse routers by +1.25 to +6.21 points.

AI USE STATEMENT

Generative AI tools were used to assist with data generation and code implementation. In benchmark construction, the candidate VLMs generate responses and execution trajectories, and GPT-5.5 is used for semantic judging when rule-based matching is insufficient (Appendix A). OpenAI Codex assisted with editing and refining the manuscript, including language polishing and LaTeX formatting. Generative AI tools were also used for literature retrieval and discovery, such as finding related work. Experimental design was carried out without generative AI assistance. The authors are responsible for verifying AI-assisted artifacts and for the data, code, analyses, references, and claims in the final manuscript.

ETHICS STATEMENT

This work studies cost-aware model routing using benchmark tasks and model executions. Lower monetary cost can make model access more affordable, but optimizing average accuracy and cost does not establish fairness or safety: routing errors may be uneven across tasks or user groups, and cheaper execution can also lower barriers to misuse. Generated code and tool-use trajectories may contain unsafe actions, while model-based judging can introduce systematic errors into correctness labels. Applications therefore require task-specific evaluation, appropriate execution isolation, access controls, and human oversight. Use and redistribution of source data, model outputs, and trajectories must respect the original licenses, provider terms, and privacy constraints. Our reported monetary costs do not measure energy use or carbon emissions and should not be read as evidence of environmental savings.

REPRODUCIBILITY STATEMENT

Section 2 specifies the routing formulation, calibration selection, model profiles, architecture, and training objectives. Section 3 and Appendix B describe the benchmark composition and candidate models. Appendix A documents task conversion, execution tools, trajectory logging, correctness verification, and data splits. Appendix C reports the encoders, training hyperparameters, optimization and checkpoint-selection procedures, and open-set evaluation setup. Metric definitions are given in Appendix D, with additional ablations and per-dataset results in Appendix E. These descriptions support reconstruction of the evaluation pipeline. Repeating executions of hosted models may nevertheless produce different responses and costs because of inference variability, model updates, or changes in API pricing. To support reproducibility, we log all model identifiers, timestamps, configurations, and API responses for every execution, enabling exact reconstruction of the reported results under the recorded conditions.

REFERENCES

Anthropic. System card: Claude Haiku 4.5. System Card, October 2025. URL https://www.anthropic.com/claude-haiku-4-5-system-card.

Anthropic. System card: Claude Opus 4.6. System Card, February 2026a. URL https://www.anthropic.com/claude-opus-4-6-system-card.

Anthropic. System card: Claude Sonnet 4.6. System Card, February 2026b. URL https://www.anthropic.com/claude-sonnet-4-6-system-card.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren,

Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024a.

Shuhao Chen, Weisen Jiang, Baijiong Lin, James T. Kwok, and Yu Zhang. Routerdc: Query-based router by dual contrastive learning for assembling large language models, 2024b. URL https://arxiv.org/abs/2409.19886.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifymoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent, 2025. URL https://arxiv.org/abs/2508.06600.

Evelyn Fix and Joseph L. Hodges. Discriminatory analysis—nonparametric discrimination: Consistency properties. Technical Report Project 21-49-004, Report No. 4, USAF School of Aviation Medicine, Randolph Field, Texas, February 1951.

Google DeepMind. Gemini 2.5 Flash Model Card. Model Card, 2025a. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-2-5-Flash-Model-Card.pdf.

Google DeepMind. Gemini 3 Flash Model Card. Model Card, December 2025b. URL https://deepmind.google/models/model-cards/gemini-3-flash/.

Google DeepMind. Gemini 3.5 Flash Model Card. Model Card, May 2026. URL https://deepmind.google/models/model-cards/gemini-3-5-flash/.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with apps, 2021. URL https://arxiv.org/abs/2105.09938.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. Routerbench: A benchmark for multi-llm routing system, 2024. URL https://arxiv.org/abs/2403.12031.

Zhehao Huang, Baijiong Lin, Jingyuan Zhang, Jingying Wang, Yuhang Liu, Ning Lu, Tao Li, and Xiaolin Huang. Vl-routerbench: A benchmark for vision-language model routing, 2026. URL https://arxiv.org/abs/2512.23562.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code, 2024. URL https://arxiv.org/abs/2403.07974.

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Congchao Wang, Zifeng Wang, Alec Go, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. Universal model routing for efficient llm inference, 2025. URL https://arxiv.org/abs/2502.08773.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images, 2016. URL https://arxiv.org/abs/1603.07396.

Xunhao Lai, Weiqi Xu, Yufeng Yang, Qiaorui Chen, Yang Xu, Lunbin Zeng, Xiaolong Li, Haohai Sun, Haichao Zhu, Vito Zhang, Jinkai Hu, Jiayao Li, Rui Gao, Zekun Li, Songquan Zhu, Jingkai Zhou, and Pengyu Zhao. Minimax sparse attention, 2026. URL https://arxiv.org/abs/2606.13392.

Hao Li, Yiqun Zhang, Zhaoyan Guo, Chenxu Wang, Shengji Tang, Qiaosheng Zhang, Yang Chen, Biqing Qi, Peng Ye, Lei Bai, Zhen Wang, and Shuyue Hu. Llmrouterbench: A massive benchmark and unified framework for llm routing, 2026. URL https://arxiv.org/abs/2601.07206.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12), December 2024. ISSN 1869-1919. doi: 10.1007/s11432-024-4235-6. URL http://dx.doi.org/10.1007/s11432-024-4235-6.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models, 2023. URL https://arxiv.org/abs/2311.08692.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts, 2024. URL https://arxiv.org/abs/2310.02255.

Yifan Lu, Rixin Liu, Jiayi Yuan, Xingqi Cui, Shenrun Zhang, Hongyi Liu, and Jiarong Xing. Routerarena: An open platform for comprehensive comparison of llm routers, 2025. URL https://arxiv.org/abs/2510.00202.

Haoxuan Ma, Guannan Lai, and Han-Jia Ye. Mmr-bench: A comprehensive benchmark for multimodal llm routing, 2026. URL https://arxiv.org/abs/2601.17814.

James B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 281–297. University of California Press, 1967.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning, 2022. URL https://arxiv.org/abs/2203.10244.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. Docvqa: A dataset for vqa on document images, 2021. URL https://arxiv.org/abs/2007.00398.

Daniel McFadden. Conditional logit analysis of qualitative choice behavior. In Paul Zarembka (ed.), *Frontiers in Econometrics*, pp. 105–142. Academic Press, New York, 1974.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data, 2025. URL https://arxiv.org/abs/2406.18665.

OpenAI. Introducing GPT-5.4, March 2026a. URL https://openai.com/index/introducing-gpt-5-4/.

OpenAI. Introducing GPT-5.4 mini and nano, March 2026b. URL https://openai.com/index/introducing-gpt-5-4-mini-and-nano/.

OpenAI. GPT-5.5 System Card. https://openai.com/index/gpt-5-5-system-card/, April 2026c. Accessed: 2026-07-29.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Cheng Qian, Zuxin Liu, Shirley Kokane, Akshara Prabhakar, Jielin Qiu, Haolin Chen, Zhiwei Liu, Heng Ji, Weiran Yao, Shelby Heinecke, Silvio Savarese, Caiming Xiong, and Huan Wang. xrouter: Training cost-aware llms orchestration system via reinforcement learning, 2025. URL https://arxiv.org/abs/2510.08439.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Ryan M. Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141, 2004.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi: 10.1038/323533a0.

Seamus Somerstep, Felipe Maia Polo, Allysson Flavio Melo de Oliveira, Prattyush Mangal, Mírian Silva, Onkar Bhardwaj, Mikhail Yurochkin, and Subha Maity. Carrot: A cost aware rate optimal router, 2025. URL https://arxiv.org/abs/2502.03261.

Chenxu Wang, Hao Li, Yiqun Zhang, Linyao Chen, Jianhao Chen, Ping Jian, Peng Ye, Qiaosheng Zhang, and Shuyue Hu. Icl-router: In-context learned model representations for llm routing, 2025. URL https://arxiv.org/abs/2510.09719.

xAI. Grok-1.5 vision preview. https://x.ai/news/grok-1.5v, April 2024. Introduces the RealWorldQA benchmark.

Jingjun Xu, Hongji Pu, Tao Feng, Haozhen Zhang, Jiaxuan You, and Ge Liu. Routeprofile: Graph-based profiling for cold-start llm routing, 2026. URL https://arxiv.org/abs/2605.00180.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi, 2024. URL https://arxiv.org/abs/2311.16502.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Yubo Zhang, Xueqing Wang, Manhui Lin, Yue Zhang, Penglongyi Deng, Ting Sun, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Changda Zhou, Hongen Liu, Suyin Liang, Cheng Cui, Yi Liu, Dianhai Yu, and Yanjun Ma. Pp-ocrv6: From 1.5m to 34.5m parameters, surpassing billion-scale vlms on ocr tasks, 2026. URL https://arxiv.org/abs/2606.13108.

Pengfei Zhou, Zhiwei Tang, Yixing Ma, Jiasheng Tang, Yizeng Han, Zhenglin Wan, Fanqing Meng, Wei Wang, Bohan Zhuang, Wangbo Zhao, et al. Agent-as-a-router: Agentic model routing for coding tasks. *arXiv preprint arXiv:2606.22902*, 2026.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, Thong Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, Prateek Yadav, Naman Jain, Alex Gu, Zhoujun Cheng, Jiawei Liu, Qian Liu, Zijian Wang, Binyuan Hui, Niklas Muennighoff, David Lo, Daniel Fried, Xiaoning Du, Harm de Vries, and Leandro Von Werra. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions, 2025. URL https://arxiv.org/abs/2406.15877.

SUPPLEMENTARY MATERIAL

A DATA GENERATION

VLM-ExecRouterBench provides a reproducible routing benchmark built from standardized executions of a shared candidate VLM pool. The final benchmark contains 33,966 samples and 373,626 sample–model executions, covering 12 datasets and 11 candidate VLMs. To support reuse and independent extension, we will release the data-generation pipeline together with the benchmark data, including the source conversion scripts, executor prompts, tool schemas, verification code, and saved execution trajectories. Figure 5 summarizes the end-to-end construction process across the three execution domains.

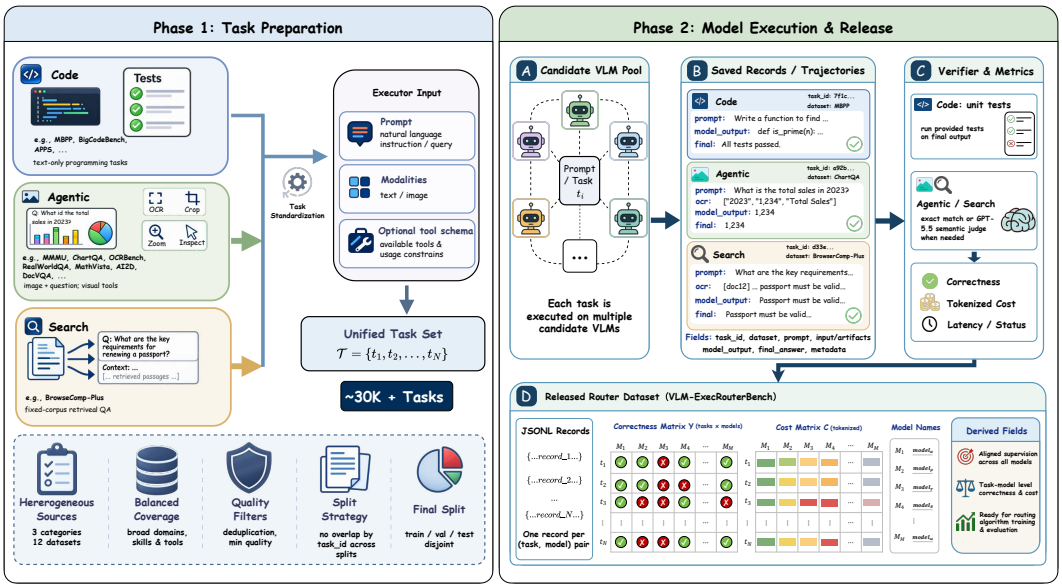


Figure 5: Overview of the VLM-ExecRouterBench data-generation pipeline. Heterogeneous Code, Agentic, and Search source tasks are standardized into executor inputs with the appropriate modalities and optional tool schemas. Candidate VLMs produce responses through the shared executor: Code tasks use one-shot code generation without environment interaction, while Agentic and Search tasks may produce saved tool trajectories. Task-specific verifiers then convert each execution into correctness and cost entries. The resulting JSONL records and aligned correctness/cost matrices form the released router dataset.

Unified task interface. Each source example is first converted into a common execution-oriented schema with three components. The **Routing Input** contains only the pre-execution information available to the router. The **Execution Context** contains the complete prompt, visual inputs, tool specification, or retrieval context sent to the candidate VLM. The **Verification Rule** specifies the task-dependent checker used to map a model response to a binary correctness label. This design keeps the router input lightweight while evaluating each VLM under the full execution protocol.

Execution backends and tools. The three benchmark domains share the same schema but use different execution backends.

Code. Code examples are converted into a unified executor prompt containing the problem statement and available public tests. Candidate VLMs return code solutions, which are checked by unit tests, input–output tests, or official benchmark evaluators depending on the source dataset.

Agentic. Multimodal document and visual reasoning examples are executed through a lightweight multimodal agent loop. The agent receives the image and question directly, can call image tools before answering, and must return a final natural-language response. The available tools include image inspection for dimensions, crop and zoom operations for isolating local visual regions, and an OCR(Zhang et al., 2026) tool for reading image text. OCR is implemented with a PaddleOCR-backed HTTP service that returns recognized text blocks, confidence scores, and bounding boxes.

This tool interface makes document, chart, diagram, and OCR-heavy VQA tasks closer to realistic interactive VLM use, rather than a single opaque image-to-answer call.

Search. BrowseComp-Plus examples are executed with a bounded fixed-corpus retrieval tool. Each model may issue retrieval calls against the provided corpus and then must return an exact final answer string. Retrieval is implemented with a dense retriever based on Qwen3-Embedding-8B(Zhang et al., 2025): query embeddings are generated with the retrieval instruction prefix used for web-search-style passage retrieval, L2-normalized, and scored against precomputed corpus embeddings by inner product. Each retrieval call returns the top five passages with their ranks, similarity scores, document identifiers, and available fields such as title and URL. Passage text is capped at 2,400 characters before being returned to the model, which keeps the search context bounded while still preserving enough evidence for multi-hop answer synthesis.

Trajectory logging. In addition to final correctness labels, the generation pipeline records execution trajectories for downstream analysis. For multimodal agent tasks, the trajectory contains the model turns, tool schemas, issued tool calls, crop/zoom or OCR results, tool-budget state, and final answer. For Search tasks, the trajectory records retrieval queries, retrieved document identifiers, snippets, and final answers. These traces are not used by the router at inference time, but they make the benchmark useful for studying model tool-use behavior, failure modes, and future trajectory-aware routing methods.

Verification and matrix construction. Responses are verified using the strongest available automatic checker for each source: executable tests for code, normalized exact/choice matching for short-answer visual tasks, GPT-5.5(OpenAI, 2026c) semantic judging only when rule-based matching is insufficient, and answer matching against the BrowseComp-Plus references for search. After verification, every sample–model pair contributes one entry to the correctness matrix $Y \in \{0,1\}^{N\times K}$. Logged token usage and the corresponding input/output prices define the cost matrix $C \in \mathbf{R}_{+}^{N\times K}$. The official train/dev/test split contains 23,776/3,396/6,794 samples. Figure 6 shows the dataset composition, and Table 5 lists the candidate models and token prices.

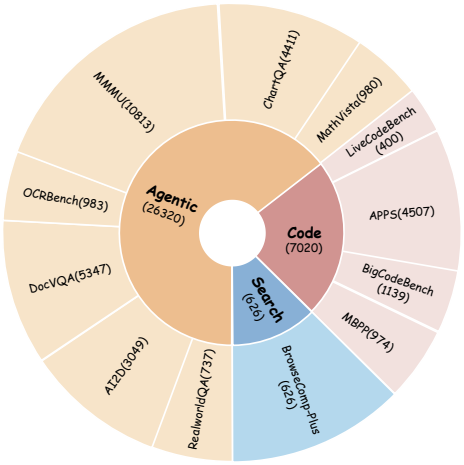


Figure 6: Data composition of VLM-ExecRouterBench. The inner ring shows the three execution-oriented domains—Agentic, Code, and Search, while the outer ring presents their constituent datasets.

Table 5: Candidate model pool. Prices are per 1M tokens.

#	Model	Provider	Input ($)	Output ($)
1	Qwen3-VL-8B-Instruct	Qwen / Alibaba	0.08	0.50
2	Gemini 2.5 Flash Lite	Google	0.10	0.40
3	Qwen3.5-35B-A3B	Qwen / Alibaba	0.14	1.00
4	MiniMax M3	MiniMax	0.30	1.20
5	Gemini 3 Flash Preview	Google	0.50	3.00
6	GPT-5.4 mini	OpenAI	0.75	4.50
7	Claude Haiku 4.5	Anthropic	1.00	5.00
8	Gemini 3.5 Flash	Google	1.50	9.00
9	GPT-5.4	OpenAI	2.50	15.00
10	Claude Sonnet 4.6	Anthropic	3.00	15.00
11	Claude Opus 4.6	Anthropic	5.00	25.00

B Source Datasets

B.1 Code

- **MBPP** contributes 974 entry-level Python programming tasks focused on basic algorithmic and functional-programming skills. Each problem is paired with unit tests, making it a clean source for evaluating whether a routed model can produce executable and semantically correct code rather than merely plausible text.
- **BigCodeBench** contributes 1,139 more realistic Python programming tasks with richer function specifications and hidden test suites. Compared with MBPP, it introduces broader library usage, longer prompts, and more implementation-heavy requirements, making it useful for distinguishing models that differ in practical coding ability.
- **APPS** contributes 4,507 competitive-programming problems described by natural-language statements and evaluated through input–output tests. It emphasizes algorithmic reasoning under stdin/stdout-style execution, where solutions must generalize beyond visible examples and handle edge cases.
- **LiveCodeBench** contributes 400 recent programming-contest-style problems, covering both function-call and standard-input formats. Its temporally newer tasks reduce overlap with older training distributions and provide a sharper test of current code-generation robustness.

B.2 Agentic

- **MMMU** contributes 10,813 multimodal questions spanning college-level disciplines and heterogeneous visual formats such as diagrams, tables, charts, and scientific figures. It is included to evaluate expert-level visual reasoning that requires both domain knowledge and deliberate multimodal understanding.
- **DocVQA** contributes 5,347 document-image question-answering examples. It targets visually grounded reading, layout understanding, and information extraction from scanned or rendered documents, covering a capability that differs substantially from natural-image recognition.
- **ChartQA** contributes 4,411 questions over charts and plots. It tests whether a model can interpret visual encodings, compare quantities, and perform chart-specific reasoning, making it a useful diagnostic for models that can read text but struggle with structured visual data.
- **AI2D** contributes 3,049 grade-school science diagram questions. The dataset stresses diagram parsing, symbolic visual elements, arrows, object-relation reasoning, and educational science knowledge, complementing document and chart understanding with diagram-centric reasoning.
- **OCRBench** contributes 983 text-centric visual understanding examples. It evaluates OCR and downstream reasoning over text embedded in images, capturing failure modes where a model's final answer is bottlenecked by visual text recognition.
- **MathVista** contributes 980 visually grounded mathematical reasoning problems. It combines figures, tables, diagrams, and quantitative reasoning, allowing the benchmark to probe fine-grained differences in multimodal math capability.

- **RealWorldQA** contributes 737 questions about real-world images. It provides open-ended perception and commonsense reasoning over natural scenes, balancing the more structured document, chart, and diagram datasets in the Agentic domain.

B.3 SEARCH

- **BrowseComp-Plus** contributes 626 fixed-corpus deep-search questions. It forms the Search domain: models must retrieve evidence from a bounded corpus and synthesize the final answer, allowing routing evaluation to include retrieval-grounded reasoning rather than only direct question answering.

C IMPLEMENTATION DETAILS

Query and profile representation. Unless otherwise stated, we use BGE-M3(Chen et al., 2024a) as the text encoder and DINOv2-large(Oquab et al., 2023) as the vision encoder. Text and vision embeddings are individually normalized and then concatenated. SCOPE-Router builds a calibration-derived behavior profile for each candidate VLM. A QueryMLP projects the incoming query representation to a 64-dimensional routing vector, while a ProfileMLP projects each model profile to the same 64-dimensional space; the router scores their match in this shared routing space. This keeps the router independent of model identity embeddings and allows newly onboarded VLMs to be introduced through calibration profiles.

Calibration. The default calibration budget is 1024 training samples. Hybrid calibration combines random coverage, diagnostic samples selected by model disagreement and cost-sensitive utility, and diversity samples selected in the frozen embedding space. The default mixture uses 50% random, 30% diagnostic, and 20% diversity samples. Profiles are built from calibration correctness, cost, value scores, and aggregated task-level behavior statistics.

Training. SCOPE-Router maps each query representation and each calibration-derived model profile into a shared 64-dimensional routing space. The default QueryMLP and ProfileMLP use 128-dimensional hidden layers with dropout 0.5. The router uses a dot-product scorer and cost-aware supervision with $\lambda = 10$. We optimize with AdamW using a fixed learning rate of 0.001, weight decay 0.003, and batch size 512. Training runs for at most 100 epochs with early stopping patience 20, selecting checkpoints by development Rank Score. In the main setting, the query encoder is frozen, so training updates the QueryMLP and ProfileMLP. We use this configuration for the main experiments unless an ablation explicitly changes the hidden dimension, calibration strategy, or encoder update policy.

Open-set evaluation. For open-set model experiments, held-out VLMs are removed from router training and are also excluded from calibration-set selection. They are allowed to join the candidate pool only after behavior profiles are constructed on the fixed calibration set. This models the practical deployment setting where a newly released VLM is profiled once on the calibration set before serving traffic. In strict unseen-pool evaluation, the test-time candidate set contains only held-out VLMs, so the router must choose among models whose training labels were never observed.

D EVALUATION METRICS

Accuracy and cost. Average accuracy is the fraction of routed samples answered correctly by the selected VLM. Average cost is computed from the token-level execution cost of the selected VLM and reported in dollars per 10K samples in the result tables.

Rank Score. Following VL-RouterBench, Rank Score harmonically averages accuracy and log-normalized cost. Given the full-sample cost range $[c_{\min}, c_{\max}]$ and router average cost $\bar{C}$, the normalized cost score is

$$C_{\text{norm}} = 100 \times \frac{\log_2(c_{\max}) - \log_2(\bar{C})}{\log_2(c_{\max}) - \log_2(c_{\min})}. \tag{13}$$

The factor of 100 aligns the cost term with the percentage scale of average accuracy. We then combine average accuracy $\bar{A}$ and normalized cost using the weighted harmonic mean:

$$S(\beta) = \frac{(1+\beta) \cdot \bar{A} \cdot C_{\text{norm}}}{\beta \cdot \bar{A} + C_{\text{norm}}}, \tag{14}$$

where $\beta > 0$ controls the preference between quality and cost. We set $\beta = 0.1$ by default to prioritize accuracy. Oracle selects the cheapest correct model per sample; Strongest always selects the globally most accurate model; Cheapest always selects the globally cheapest model.

E Additional Results

E.1 Generalization to held-out tasks and models

Table 6 evaluates task distribution shift with all 11 candidate models available. SCOPE-Router obtains a Rank Score of 88.14, compared with 86.30 for K-means, at costs of 12.79 and 58.67 dollars per 10K samples, respectively. Table 7 evaluates held-out models under both ID and OOD task distributions.

Table 6: OOD generalization results with all 11 candidate models available. Rank is determined by Rank Score.

Router	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑	Rank ↓
Training-free Baselines				
Oracle	96.87	7.68	97.15	0
Strongest	89.45	224.98	75.29	15
Cheapest	75.86	2.82	77.56	14
End-to-End Routers				
CosineCls	78.13	5.07	79.72	9
RouterDC	78.53	5.23	80.10	7
ZOOTER	77.42	5.32	79.04	11
VLC	77.31	9.24	78.94	12
Feature-level Routers				
KNN	76.63	4.34	78.29	13
PRkNN	81.31	6.06	82.71	4
OVR	87.37	76.06	83.93	3
K-means	88.74	58.67	86.30	2
Linear	82.37	58.97	80.76	5
MLP	78.31	5.09	79.88	8
UniRoute-KM	78.07	13.09	79.66	10
UniRoute-LM	78.88	12.04	80.42	6
SCOPE-Router (Ours)	**87.10**	12.79	**88.14**	**1**

Table 7: Open-set routing results with 5 of 11 candidate models held out under in-distribution and out-of-distribution settings.

Router	In-Distribution				Out-of-Distribution			
	Acc. ↑	Cost ↓	Score ↑	Rank ↓	Acc. ↑	Cost ↓	Score ↑	Rank ↓
UniRoute-KM	73.58	31.78	74.51	2	76.95	3.21	78.60	2
UniRoute-LM	73.05	31.03	74.06	3	75.86	2.82	77.56	3
SCOPE-Router	**78.44**	82.93	**75.96**	**1**	**84.21**	19.84	**85.35**	**1**

E.2 Router configuration and calibration

Tables 8–10 examine the hidden dimension, query-encoder updates, and calibration strategy. Rank Scores vary by 0.62 points across the tested hidden dimensions. Freezing the encoder gives a

slightly higher score on VL-RouterBench, while fine-tuning gives a slightly higher score on VLM-ExecRouterBench. The full calibration mixture scores 76.18; the best single strategy, diversity sampling, scores 75.69.

Table 8: Effect of MLP hidden dimension on VL-RouterBench.

Hidden Dim	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑
64	77.37	0.77	75.56
128	**79.11**	0.98	**76.18**
256	78.23	0.82	76.08
512	78.27	0.87	75.90
1024	78.05	0.82	75.91
2048	78.52	0.88	76.08
4096	78.68	0.97	75.83

Table 9: Effect of freezing or fine-tuning the query embedder on VL-RouterBench and VLM-ExecRouterBench.

Variant	VL-RB			VLM-ExecRB		
	Acc. ↑	Cost ↓	Score ↑	Acc. ↑	Cost ↓	Score ↑
Frozen	**79.11**	0.98	**76.18**	81.65	46.20	80.94
Fine-tuned	78.36	0.94	75.68	**81.76**	44.85	**81.14**

Table 10: Calibration-strategy ablation on VL-RouterBench. R, D, and Div denote Random, Diagnostic, and Diversity sampling, respectively.

Variant	Avg. Acc. ↑	Avg. Cost ↓	Rank Score ↑
R only	77.73	0.90	75.31
D only	77.34	0.80	75.37
Div only	78.59	0.99	75.69
R+D	78.87	0.96	75.78
R+Div	78.44	0.88	76.00
D+Div	78.37	0.90	75.87
R+D+Div (Full)	**79.11**	0.98	**76.18**

Rank Scores differ by less than 0.8 points across the 25 text and vision encoder combinations in Figure 7.

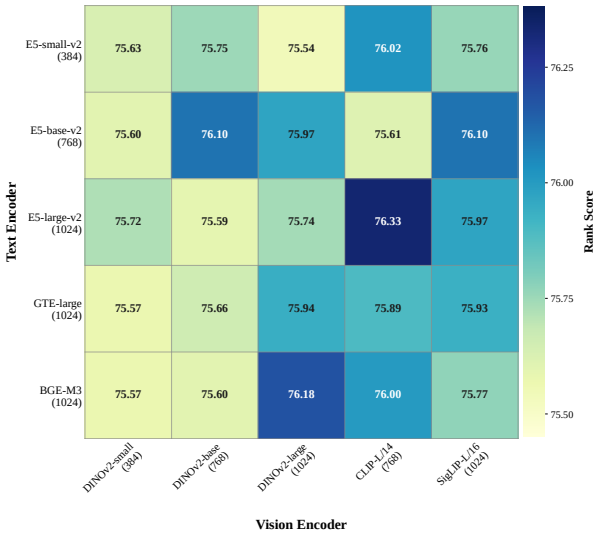


Figure 7: Rank Scores for 25 text and vision encoder combinations.

E.3 Results by Dataset

Tables 11–13 report results for Agentic, Code, and Search, followed by the overall comparison. Accuracy and cost appear in separate panels in the Agentic and Code tables. Each row corresponds to a router, and each column to a dataset. The hard-label group contains KNN, PRkNN, OVR, and K-means. The soft-label group contains Linear, MLP, CosineCls, RouterDC, VLC, ZOOTER, both UniRoute variants, and SCOPE-Router.

Table 11: Per-dataset router results on the Agentic tasks in VLM-ExecRouterBench. Bold and underlined entries mark the best and second-best learned-router results, respectively, excluding the baselines. RWQA denotes RealWorldQA; Avg. denotes the Agentic average.

(a) Accuracy (%) ↑

Router	DocVQA	OCR	ChartQA	AI2D	MathVista	MMMU	RWQA	Avg.
Oracle	99.53	97.00	95.63	99.84	97.52	96.06	98.59	97.28
Strongest	96.22	88.00	89.51	93.74	88.61	86.71	79.58	89.85
Cheapest	92.62	80.50	76.07	77.37	63.86	60.91	59.15	72.59
KNN	93.00	85.50	78.58	80.42	73.27	63.41	58.45	75.01
PRkNN	94.13	84.00	84.70	82.99	82.18	67.72	61.27	78.71
OVR	97.54	89.50	**88.52**	91.81	90.10	82.40	72.54	**87.88**
K-means	93.66	87.00	87.65	**93.26**	**93.07**	**83.60**	76.06	87.73
Linear	94.42	85.00	80.11	89.57	83.17	75.41	**79.58**	82.45
MLP	93.19	89.50	84.37	82.18	82.18	66.37	64.08	78.10
CosineCls	94.04	**91.00**	87.87	85.39	81.68	73.74	70.42	82.47
RouterDC	93.66	88.00	87.32	85.07	81.68	74.34	71.83	82.43
VLC	93.09	88.00	86.89	85.07	81.68	73.78	61.27	81.73
ZOOTER	92.81	86.00	83.83	84.11	80.20	72.77	61.97	80.50
UniRoute-KM	94.32	81.00	80.11	82.66	77.72	70.17	71.13	78.90
UniRoute-LM	95.84	84.50	83.06	82.99	82.18	72.30	69.01	80.86
SCOPE-Router	**97.92**	82.00	87.87	85.07	82.18	74.43	71.13	83.18

Table 11: Agentic results (continued): cost in $/10K samples.

(b) Cost ($/10K samples) ↓								
Router	DocVQA	OCR	ChartQA	AI2D	MathVista	MMMU	RWQA	Avg.
Oracle	3.77	5.78	15.73	7.19	14.22	21.06	9.74	13.92
Strongest	246.03	221.49	269.70	165.90	266.49	335.52	400.87	281.17
Cheapest	3.01	2.45	2.43	6.60	16.69	19.39	3.16	10.51
KNN	3.49	**2.84**	**3.11**	7.42	18.66	19.69	5.05	11.08
PRkNN	5.13	3.30	3.73	**6.75**	**12.77**	27.38	5.40	14.38
OVR	86.96	81.00	65.47	101.97	170.23	211.32	178.59	141.10
K-means	38.03	26.93	25.18	93.96	159.04	196.39	221.78	116.04
Linear	57.08	79.33	32.35	68.15	108.23	119.19	352.91	90.14
MLP	3.40	4.25	3.81	7.91	16.29	**18.91**	6.90	**10.94**
CosineCls	4.31	3.59	4.68	8.53	15.48	35.15	8.73	17.95
RouterDC	4.82	3.69	4.75	8.56	15.48	34.70	8.62	17.89
VLC	3.67	3.43	4.59	8.56	15.56	33.43	6.96	17.06
ZOOTER	**3.15**	3.04	3.26	7.73	15.14	28.96	**3.89**	14.70
UniRoute-KM	26.77	7.21	4.25	11.48	16.71	32.26	13.24	21.84
UniRoute-LM	39.99	10.65	11.01	8.54	15.47	34.44	12.99	26.26
SCOPE-Router	14.31	5.48	5.57	8.56	17.59	38.10	13.24	21.58

Table 12: Per-dataset router results on the Code tasks in VLM-ExecRouterBench. LCB and BCB denote LiveCodeBench and BigCodeBench; Avg. denotes the Code average. Formatting follows Table 11.

(a) Accuracy (%) ↑					
Router	LCB	MBPP	APPS	BCB	Avg.
Oracle	94.37	99.51	89.04	74.35	88.42
Strongest	92.96	92.68	82.40	53.48	79.62
Cheapest	70.42	74.63	49.30	47.83	53.96
KNN	71.83	76.10	53.15	45.65	56.30
PRkNN	80.28	75.61	61.66	47.39	62.32
OVR	91.55	**96.10**	77.16	**55.65**	77.13
K-means	**92.96**	93.17	**81.93**	55.65	**79.77**
Linear	87.32	86.83	73.43	47.83	71.85
MLP	84.51	82.93	68.30	49.57	68.18
CosineCls	84.51	74.63	70.28	46.96	67.74
RouterDC	85.92	76.59	69.23	50.00	67.96
VLC	70.42	74.63	49.42	47.39	53.96
ZOOTER	70.42	79.51	49.42	28.26	51.47
UniRoute-KM	85.92	84.88	70.51	54.35	70.75
UniRoute-LM	87.32	86.83	70.05	53.91	70.75
SCOPE-Router	90.14	87.32	79.84	53.04	76.98

Table 12: Code results (continued): cost in $/10K samples.

(b) Cost ($/10K samples) ↓					
Router	LCB	MBPP	APPS	BCB	Avg.
Oracle	5.49	6.10	41.31	12.21	29.25
Strongest	53.90	55.82	122.09	40.93	94.90
Cheapest	5.33	2.86	28.09	1.61	18.65
KNN	7.39	5.33	40.35	4.73	27.36
PRkNN	12.47	5.63	50.15	3.41	33.62
OVR	72.86	192.55	174.52	93.10	158.21
K-means	54.76	59.55	141.40	59.73	110.82
Linear	35.39	20.01	78.41	13.95	56.52
MLP	12.61	14.10	54.52	3.79	37.71
CosineCls	12.89	2.86	54.82	3.79	36.22
RouterDC	12.96	8.46	54.22	3.36	36.62
VLC	**4.77**	**2.58**	30.90	11.50	22.01
ZOOTER	5.33	5.03	**27.08**	**2.40**	**18.47**
UniRoute-KM	12.96	15.77	54.91	26.78	42.10
UniRoute-LM	15.80	16.60	58.34	26.48	44.48
SCOPE-Router	30.76	31.99	106.83	36.57	79.78

Table 13: Router results on Search and across all datasets in VLM-ExecRouterBench. Accuracy is in %; cost is in $/10K samples. Formatting follows Table 11.

	Search (BrowseComp-Plus)		All datasets		
Router	Acc. ↑	Cost ↓	Acc. ↑	Cost ↓	Score ↑
Oracle	84.09	2203.19	95.25	59.53	91.77
Strongest	41.67	3599.38	86.86	308.24	66.75
Cheapest	15.15	359.94	67.74	18.93	69.78
KNN	52.27	1379.06	70.81	40.93	71.41
PRkNN	62.12	1171.33	75.10	40.72	75.36
OVR	64.39	13658.61	85.27	407.16	55.14
K-means	65.15	12839.69	**85.69**	362.20	60.73
Linear	58.33	1004.43	79.85	101.16	76.15
MLP	65.91	**635.84**	75.88	28.45	76.87
CosineCls	65.15	639.89	79.17	33.71	79.55
RouterDC	61.36	892.86	79.11	38.65	79.16
VLC	65.15	679.74	75.83	30.93	76.66
ZOOTER	62.12	637.13	74.32	**27.55**	75.47
UniRoute-KM	43.18	1167.60	76.57	48.16	76.26
UniRoute-LM	66.67	678.14	78.55	42.58	78.40
SCOPE-Router	**68.18**	687.35	81.65	46.20	**80.94**

E.4 CANDIDATE-MODEL DIAGNOSTICS

Figure 8 shows each candidate model's accuracy on individual datasets. Figure 9 compares the accuracy profiles of six models across the same datasets. Figure 10 places the full candidate pool in the accuracy–cost plane.

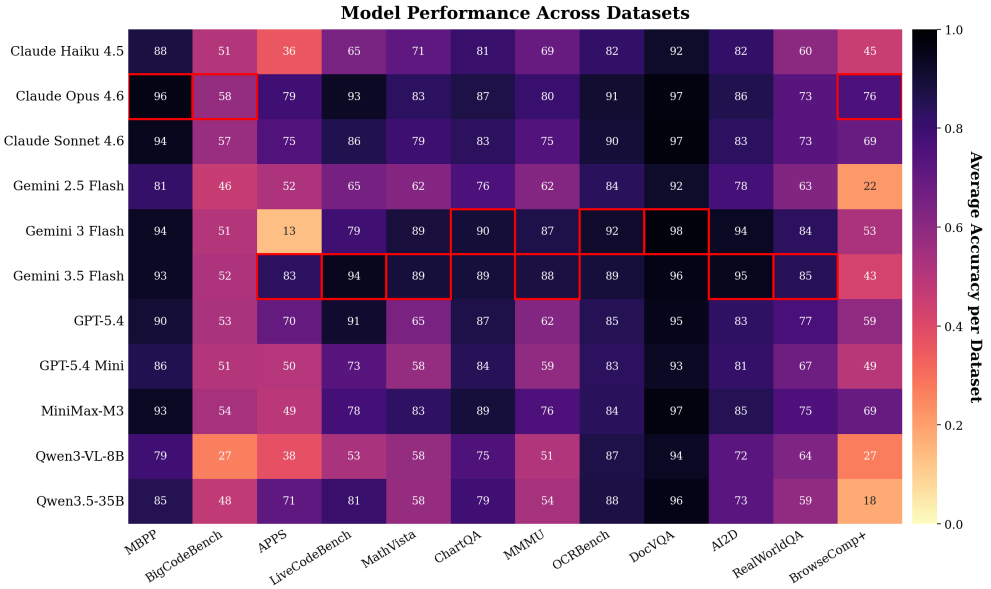


Figure 8: Per-dataset accuracy of each candidate model on VLM-ExecRouterBench. The heatmap exposes complementary model strengths across code, multimodal visual reasoning, and search-style datasets.

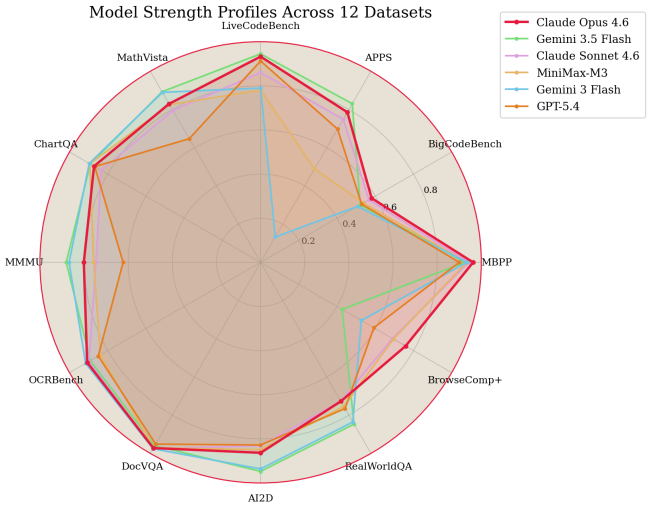


Figure 9: Accuracy profiles of six candidate models across the 12 datasets in VLM-ExecRouterBench.

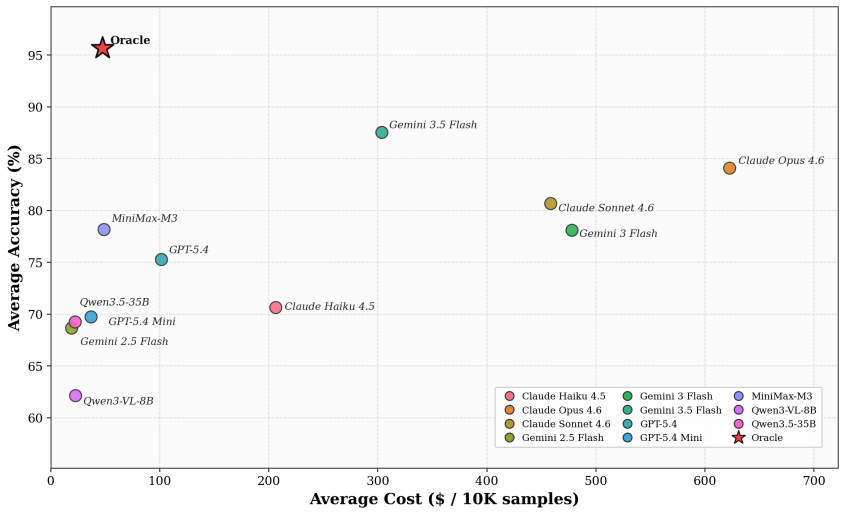


Figure 10: Accuracy–cost distribution of individual candidate VLMs and the oracle reference computed from the final routing matrix. The plot illustrates that higher global accuracy often comes with substantially higher execution cost, motivating cost-aware routing.

E.5 Accuracy–cost operating points

Table 14 gives the unscaled values plotted in Figure 3: single operating points in panel (a) and a three-value sweep of λ in panel (b).

Table 14: Accuracy–cost operating points on VLM-ExecRouterBench. Accuracy is in %; cost is in $/10K samples.

(a) Single operating points

Method	Acc.	Cost
Oracle	95.25	59.53
KNN	70.81	40.93
PRkNN	75.10	40.72
OVR	85.27	407.16
K-means	85.69	362.20

(b) Cost-sensitivity sweep

	$\lambda = 10$		$\lambda = 100$		$\lambda = 1000$	
Method	Acc.	Cost	Acc.	Cost	Acc.	Cost
Linear	75.26	41.58	79.85	101.15	81.90	85.89
MLP	72.48	43.90	75.88	28.45	71.68	25.59
UniRoute-KM	67.74	18.94	76.57	48.16	67.74	18.94
UniRoute-LM	67.74	18.94	78.55	42.58	67.74	18.94
CosineCls	73.63	157.50	79.17	33.70	77.61	153.10
RouterDC	73.61	156.90	79.11	38.65	77.66	150.80
VLC	70.49	25.82	75.83	30.93	71.46	35.68
ZOOTER	70.24	21.68	74.32	27.55	69.71	21.58
SCOPE-Router	74.68	23.91	81.65	46.20	80.42	30.94

F Limitations

SCOPE-Router extends naturally to newly added VLMs, but each new model must still be profiled on a labeled calibration set before it can be routed reliably. As a result, open-set performance depends on how well the calibration examples expose the capabilities and failure modes of the incoming model. In addition, the current router makes a single task-level decision and uses the selected model for the entire execution. This matches the benchmark setting, where each sample is represented by one correctness and cost entry per candidate VLM, but it does not adapt to changing requirements within long-horizon Agentic or Search trajectories. For instance, an early step may require fine-grained OCR or visual localization, whereas a later step may require retrieval synthesis or answer normalization. Future work will study more robust model profiling from compact calibration sets and dynamic routing policies that can revise model choices across intermediate execution steps.