AgentRouter: Heterogeneous Model Routing for Cost-Optimal Multi-Step Agentic Workflows

Rudrendu Kumar Paul [1] **Sourav Nandy** [2]

Abstract

Enterprise agentic systems that route every trajectory step to a frontier model waste 60–80% of their inference budget on subtasks that smaller models handle equally well. Existing routing solutions optimize single-turn query assignment but ignore a property unique to agentic workflows: subtask complexity varies widely within a single trajectory. A planning step may require frontier-class reasoning while a subsequent formatting step needs only a 7B model. We formalize *step-level model routing* as a sequential assignment problem over agent trajectories and propose AgentRouter, a lightweight classifier (12M parameters, $<$5ms overhead per step on an A100 GPU) that maps each trajectory step to one of four model tiers using five features extractable at routing time. Trained on 50,000 annotated agent trajectory steps spanning planning, coding, research, and data analysis tasks, AgentRouter achieves 72% cost reduction relative to frontier-only baselines, retaining 97.3% of frontier-only quality (less than 3% degradation in end-to-end task completion); per-step routing accuracy reaches 91% on minimal-complexity steps and 85% on efficient-tier steps, with 76–82% on the harder mid-range and frontier tiers. On the same benchmarks, RouteLLM and FrugalGPT (applied per-step) achieve only 31% and 44% cost reduction respectively, because their single-turn training signal misses trajectory-level quality dependencies.

[1]Boston University, Boston, MA, USA [2]University of Texas at Austin, Austin, TX, USA. Correspondence to: Rudrendu Kumar Paul <rudrendupaul2022@gmail.com>, Sourav Nandy <sourav.nandy@gmail.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

1. Introduction

Consider a document analysis agent executing a seven-step workflow. Step 1 decomposes the user query into subtasks (multi-hop reasoning, chain-of-thought planning). Step 2 retrieves relevant document chunks (straightforward embedding lookup). Step 3 synthesizes findings across chunks (cross-document reasoning over long context). Steps 4–5 extract structured fields and format them as JSON. Steps 6–7 generate a summary and validate output format. Routing all seven steps to GPT-4-class models costs $0.42 per execution. Steps 2, 4, 5, and 7 produce identical quality when served by a 7B model at $0.03 per step, dropping execution cost to $0.18 (a 57% reduction on this single workflow) with no measurable quality loss.

This pattern repeats across every production agent system we have operated. In enterprise deployments spanning document processing, compliance review, and customer interaction, 55–70% of agent trajectory steps require no frontier-model capability. The steps that do (initial planning, complex reasoning, ambiguous input handling) are a minority of the token budget, yet they determine end-to-end quality.

Existing routing approaches were designed for single-turn queries. RouteLLM (Ong et al., 2024) selects between a strong and weak model per query using preference data. FrugalGPT (Chen et al., 2024) cascades from cheapest to most expensive model until a confidence threshold is met. Both treat each call as independent, missing the fact that step complexity varies within a single trajectory and that routing decisions at step t affect context quality at step $t{+}1$. Applying either method per-step effectively converts each agent step into an independent single-turn query; the routing decision for step $t{+}1$ receives no signal about what model executed step t or what quality the output achieved. AgentRouter's trajectory-aware features (specifically f_{ctx} and the dependency-aware training labels) fill this gap. Recent agentic cost optimizers (Su et al., 2026; Zhang et al., 2025; Qian et al., 2025) operate at the query or trajectory level rather than per-step, and use heavyweight mechanisms (VAEs, RL-trained LLMs) with non-trivial overhead.

We make three contributions:

1. We formalize **step-level model routing** for agent trajec-

tories, defining the assignment problem over heterogeneous model tiers with per-step quality constraints and end-to-end quality guarantees (Section 3).
2. We propose **AgentRouter**, a 12M-parameter classifier that routes each step to one of four model tiers using five features extractable at routing time, adding $<$5ms latency per step (measured on an A100 GPU; see Section 3).
3. We evaluate on a **benchmark of 2,400 agent trajectories** across four task domains, demonstrating 72% cost reduction with $<$3% quality degradation, outperforming adapted single-turn routers and agent-level optimizers (Section 4).

2. Related Work

Single-turn model routing. RouteLLM (Ong et al., 2024) pioneered learning-based routing between strong and weak LLMs using preference data. FrugalGPT (Chen et al., 2024) introduced LLM cascades that try cheaper models first. PILOT (Panda et al., 2025) frames routing as a contextual bandit with budget modeled as a multi-choice knapsack. Pulishetty et al. (2025) use cross-attention to jointly model query and model embeddings. Two recent surveys (Moslem & Kelleher, 2025; Varangot-Reille et al., 2025) provide taxonomies of routing paradigms, distinguishing pre-generation and post-generation approaches. All of these operate on independent queries. When applied to agent trajectories by routing each step independently, they miss inter-step quality dependencies: a cheap model producing a subtly wrong intermediate result forces the frontier model at the next step to reason over corrupted context, degrading end-to-end quality.

Agent-level cost optimization. DAAO (Su et al., 2026) uses a VAE to estimate query difficulty and allocates operators and LLMs accordingly, surpassing prior systems by 11.21% in accuracy while using only 64% of their inference cost. Budget-Aware Agentic Routing (Zhang et al., 2025) selects between a cheap and expensive model at each agent step using boundary-guided policy optimization. xRouter (Qian et al., 2025) trains an RL-based tool-calling router with cost-aware rewards. CoRL (Jin et al., 2025) uses a centralized controller LLM to coordinate expert models under budget constraints. These methods reduce cost but use heavyweight routing mechanisms: DAAO adds a VAE forward pass per query, xRouter and CoRL use LLMs as routers (adding the very cost they aim to reduce), and Zhang et al. (2025) are limited to binary model choice rather than multi-tier assignment.

Model selection for compound systems. Chen et al. (2025) optimize per-module model assignment in compound AI systems, achieving 5–70% accuracy gains. Sharma & Mehta (2025) find that small language models with guided decoding close the capability gap with frontier models at 10–100$\times$ lower cost for tool-use tasks. Salim et al. (2026) quantify token distribution in agentic workflows, revealing that 59.4% of tokens go to Code Review stages specifically, supporting our premise that step complexity varies enough to make per-step routing valuable. Concurrent work on Aragog (Team, 2025) performs per-stage model routing in agentic RAG workflows; AgentRouter differs by operating at the individual trajectory step level with a lightweight classifier rather than at the pipeline stage level.

3. Method: AgentRouter

3.1. Problem formulation

An agent trajectory $\tau = (s_1, s_2, \ldots, s_T)$ consists of T steps, where each step s_t represents a single LLM call with defined input context, instruction, and expected output. Let $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$ be a set of four model tiers ordered by capability and cost:

- **Tier 1** (Frontier): GPT-4-class, Claude Opus-class. \$15–30 per million output tokens.
- **Tier 2** (Mid-range): GPT-4o-mini-class, Claude Sonnet-class. \$3–8 per million output tokens.
- **Tier 3** (Efficient): Llama-3-70B-class, Mixtral-class. \$0.50–2 per million output tokens.
- **Tier 4** (Minimal): 7B–13B models. \$0.05–0.20 per million output tokens.

For each step s_t, let $q(s_t, m)$ denote the quality of output when step s_t is served by model m, and let $c(s_t, m)$ denote the cost. The step-level routing problem is:

$$\min_{\pi} \sum_{t=1}^{T} c(s_t, \pi(s_t)) \quad \text{s.t.} \quad Q(\tau, \pi) \geq Q_{\min} \qquad (1)$$

where $\pi : s_t \to \mathcal{M}$ is the routing policy and $Q(\tau, \pi)$ is the end-to-end trajectory quality under policy π. The constraint $Q_{\min}$ is a user-specified minimum quality threshold (we use 97% of the frontier-only baseline throughout experiments).

The difficulty is that $Q(\tau, \pi)$ is not decomposable into per-step terms. Routing step t to a weaker model can degrade the context available at step $t{+}1$, creating cascading quality loss. We handle this through a step-dependency-aware training procedure described below.

3.2. Feature extraction

At routing time (before the LLM call), AgentRouter extracts five features from the step specification:

1. **Inferred task type** $f_{\text{type}} \in \{\text{PLAN, REASON, RETRIEVE, GENERATE, FORMAT, VERIFY}\}$: classified from the step instruction using keyword matching and a small BERT-based tagger (2M parameters).

2. **Reasoning depth** $f_{\text{depth}} \in [0, 1]$: estimated from instruction complexity (number of conditional clauses, presence of multi-hop indicators like "compare," "synthesize across," "evaluate whether").
3. **Tool-use requirements** $f_{\text{tool}} \in \{0, 1, 2+\}$: number of tool calls specified or implied in the step.
4. **Output format constraints** $f_{\text{format}} \in \{\text{FREE}, \text{STRUCTURED}, \text{CODE}, \text{JSON}\}$: derived from output schema if specified, otherwise from instruction keywords.
5. **Context window utilization** $f_{\text{ctx}} \in [0, 1]$: ratio of input tokens to the target model's context limit, computed from the accumulated trajectory context.

Features 1–4 are extractable from the step specification alone. Feature 5 requires knowing the accumulated context size, available from the orchestrator at routing time.

3.3. Classifier architecture and training

AgentRouter is a feedforward classifier with two hidden layers (256 and 128 units), ReLU activations, and a 4-class softmax output corresponding to the four model tiers. The predicted tier is $\hat{k}_t = \arg\max_{k \in \{1,2,3,4\}} \hat{p}_k$, where $\hat{p}_k$ is the softmax probability for tier k. Input features are embedded: categorical features (f_{type}, f_{tool}, f_{format}) through learned embeddings, continuous features (f_{depth}, f_{ctx}) through linear projection. Total parameter count: 12M (dominated by the task-type tagger).

Training data. We construct a dataset of 50,000 step-level annotations from 6,250 agent trajectories across four domains: document analysis (1,800 trajectories), code generation (1,600), research synthesis (1,500), and data analysis (1,350). For each trajectory, we execute all steps with all four model tiers and record per-step quality scores (task-specific automated metrics: F1 for document extraction, pass@1 for code, factual accuracy for research, numeric correctness for data analysis). Each step receives a *minimum adequate tier* label: the cheapest tier whose quality on that step is within 2% of the frontier tier's quality.

Dependency-aware labeling. To account for inter-step dependencies, we do not label steps in isolation. Instead, we label step t assuming steps 1 through $t-1$ were routed to their minimum adequate tiers. This captures the realistic setting where a downstream step receives context generated by a mix of model tiers, not uniformly frontier-quality context.

Training procedure. We train with cross-entropy loss on the 4-class tier labels. We add a cost-sensitive regularization term that penalizes over-provisioning (routing to a higher tier than needed) less heavily than under-provisioning (routing to a tier that degrades quality):

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \sum_t \max(0, k_{\text{pred}}(s_t) - k_{\text{label}}(s_t))^2 \quad (2)$$

where k_{label} and k_{pred} are the tier indices (1–4) and $\lambda = 0.3$. This asymmetric penalty ensures AgentRouter errs toward more capable models when uncertain, preserving quality at the expense of slightly higher cost.

Algorithm 1 AgentRouter Step-Level Routing

Require: Agent trajectory specification $\tau = (s_1, \ldots, s_T)$, model pool $\mathcal{M}$, quality threshold $Q_{\min}$
Ensure: Model assignment $\pi(s_t)$ for each step
1: **for** $t = 1$ to T **do**
2: Extract features $(f_{\text{type}}, f_{\text{depth}}, f_{\text{tool}}, f_{\text{format}}, f_{\text{ctx}})$ from s_t
3: $\hat{k}_t \leftarrow \text{AgentRouter}(f_{\text{type}}, f_{\text{depth}}, f_{\text{tool}}, f_{\text{format}}, f_{\text{ctx}})$
4: $\pi(s_t) \leftarrow m_{\hat{k}_t}$
5: Execute step s_t with model $\pi(s_t)$
6: Update trajectory context with output of s_t
7: **end for**
8: $Q_{\text{actual}} \leftarrow \text{EvaluateTrajectory}(\tau, \pi)$
9: **if** $Q_{\text{actual}} < Q_{\min}$ **then**
10: Re-execute failed steps with m_1 (frontier fallback)
11: **end if**

The trajectory context update (line 6) incorporates the actual output length, quality score, and accumulated content from step t into the feature vector for step $t{+}1$ (specifically, f_{ctx} and any trajectory-level signals). This distinguishes AgentRouter from independent query routing: each routing decision uses evidence from prior steps, so the classifier operates on trajectory-aware context, with each routing decision informed by the accumulated outputs preceding it. The fallback mechanism (lines 8–10) re-executes quality-critical steps with frontier models when end-to-end quality falls below $Q_{\min}$. In evaluation, Q_{actual} is computed using the held-out ground-truth labels (F1, pass@1, factual accuracy). In production deployment, a lightweight LLM-as-a-judge proxy replaces the ground-truth evaluator; the judge's token cost and latency are included in the overhead reported in Section 4. In practice, fallback triggers on fewer than 4% of trajectories.

4. Experiments

4.1. Setup

Benchmark. We evaluate on 2,400 held-out trajectories (600 per domain) distinct from the training set. Each trajectory contains 4–12 steps. Total: 18,720 individual step-level routing decisions.

Model pool. Tier 1: GPT-4o ($15/M output tokens). Tier 2: GPT-4o-mini ($2.40/M output tokens). Tier 3: Llama-3-70B-Instruct via together.ai ($0.88/M output tokens). Tier 4: Llama-3-8B-Instruct via together.ai ($0.18/M output tokens). Prices as of February 2026.

Baselines. (1) **Frontier-only**: all steps routed to GPT-4o. (2) **RouteLLM** (Ong et al., 2024): applied per-step with

Table 1. Cost and quality comparison across baselines. Quality is end-to-end task completion relative to frontier-only (100%). Cost reduction is relative to frontier-only. Overhead is per-step routing latency.

Method	Quality	Cost Red.	Overhead
Frontier-only	100.0%	–	0 ms
RouteLLM (per-step)	94.2%	31.4%	3 ms
FrugalGPT cascade	96.8%	44.1%	48 ms
DAAO	98.1%	36.2%	22 ms
Budget-Aware	95.6%	52.7%	8 ms
AgentRouter	**97.3%**	**71.8%**	**4.7 ms**

its matrix factorization router, trained on Chatbot Arena preferences, selecting between GPT-4o and Llama-3-8B. (3) **FrugalGPT cascade** (Chen et al., 2024): adapted to four tiers, querying from cheapest to most expensive until a confidence threshold is met. (4) **DAAO** (Su et al., 2026): difficulty-aware orchestration applied at the trajectory level. (5) **Budget-Aware Routing** (Zhang et al., 2025): binary routing (GPT-4o vs. Llama-3-8B) at each step with boundary-guided training.

Metrics. End-to-end task completion rate (domain-specific: F1, pass@1, factual accuracy, numeric correctness, averaged). Cost per trajectory in USD. Cost reduction percentage relative to frontier-only. Routing overhead (milliseconds per step).[1]

4.2. Main results

Table 1 presents the main comparison. AgentRouter achieves 71.8% cost reduction while maintaining 97.3% of frontier-only quality. Budget-Aware Routing reaches 52.7% cost reduction but drops quality to 95.6% because its binary choice (frontier vs. 8B) cannot exploit mid-range tiers. FrugalGPT preserves quality (96.8%) but achieves only 44.1% cost reduction, with 48ms routing overhead from sequential model probing. RouteLLM performs worst in cost reduction (31.4%) because its preference-based router, trained on single-turn conversations, misroutes agent steps whose complexity depends on accumulated trajectory context. DAAO preserves the highest quality (98.1%) but achieves only 36.2% cost reduction: its VAE assigns a single difficulty score to the entire trajectory, provisioning all steps for the hardest one.

4.3. Per-tier routing accuracy

Table 2 breaks down routing accuracy by tier. The distribution confirms our premise: only 14.2% of steps genuinely require frontier models, yet those steps consume 48.3% of the remaining cost. Tier 4 steps (35.3% of all steps) are the easiest to classify, with 91.2% accuracy. Tier 2 steps are hardest (76.4%), because the boundary between "needs mid-range reasoning" and "could be handled by a 70B model" is often ambiguous. The asymmetric loss (Equation (2)) causes most Tier 2 errors to be over-provisions to Tier 1 rather than under-provisions to Tier 3, protecting quality at modest cost increase.

[1]Implementation details and routing configurations are provided in supplementary materials.

Table 2. Routing accuracy and cost contribution by tier. Ground truth is the minimum adequate tier determined by exhaustive evaluation.

Tier	Steps (%)	Accuracy	Cost share
Tier 1 (Frontier)	14.2%	82.1%	48.3%
Tier 2 (Mid-range)	21.6%	76.4%	24.1%
Tier 3 (Efficient)	28.9%	84.7%	16.8%
Tier 4 (Minimal)	35.3%	91.2%	10.8%
Overall	100%	84.6%	100%

4.4. Ablation and domain analysis

Feature ablation (replacing each feature with its marginal distribution) reveals task type as the dominant signal (-12.3 accuracy points when removed, cost reduction drops to 54.2%), followed by reasoning depth (-7.1), context utilization (-4.8), tool use (-3.2), and output format (-2.1). Cost reduction varies by domain: document analysis (76.4%), data analysis (74.1%), research synthesis (69.2%), code generation (63.8%). Code generation benefits least because 26.3% of its steps require Tier 1 reasoning vs. 8.7% for document analysis.

5. Discussion and Limitations

Distributional shift. AgentRouter trains on trajectories from LangGraph and CrewAI. Testing on AutoGen (unseen framework) shows 6–9% accuracy degradation, though cost reduction still reaches 58%. Cross-framework generalization remains an open problem.

Model tier evolution. The four-tier structure reflects pricing and capability gaps as of early 2026. Retraining is needed when the model pool changes, though the lightweight architecture keeps this inexpensive ($<$2 GPU-hours on a single A100).

Trajectory scope. AgentRouter assumes the step sequence is known or estimable at routing time: either specified upfront in a trajectory template or inferred from a planner before execution begins. Fully online agentic scenarios where future steps are generated dynamically based on prior outputs, with no advance visibility into remaining steps, fall outside the current scope. Extending AgentRouter to such settings would require online re-planning and incremental trajectory estimation.

Routing overhead and tier transitions. Beyond per-step latency, practitioners should account for tier-switching frequency. Across our benchmark trajectories, the median number of model-tier transitions per trajectory is 3.2 (95th percentile: 7 transitions for 8-step trajectories). Inference providers that charge per-call initialization overhead beyond per-token cost may see this switching contribute meaningfully to total latency.

Cold start and latency. The task-type tagger requires step instructions to be available before routing. For dynamically generated instructions, we route the instruction-generation call to Tier 2 by default. The post-hoc fallback (Algorithm 1, lines 8–10) catches quality violations but adds latency; mid-trajectory quality prediction is the subject of ongoing work.

6. Conclusion

We formalized step-level model routing for agent trajectories and proposed AgentRouter, a lightweight 12M-parameter classifier that assigns each step in a multi-step agent workflow to the cheapest adequate model tier. On a benchmark spanning four task domains, AgentRouter achieves 72% cost reduction with less than 3% quality degradation, outperforming both single-turn routers adapted to agentic settings and agent-level cost optimizers. The central finding is that step complexity within a single trajectory varies enough to make fine-grained routing viable, and that a small classifier trained on dependency-aware step labels captures this variation with negligible overhead. More broadly, AgentRouter implements resource-adaptive inference at the trajectory level, dynamically matching computational cost to task complexity at each agent step rather than applying uniform model selection across the workflow.

Impact Statement

Reducing the cost of frontier AI by 72% through intelligent per-step routing makes capable agent systems accessible to organizations and researchers who cannot sustain frontier-only inference budgets. The four-tier model preserves quality guarantees through a per-trajectory quality threshold and a fallback mechanism, so cost reduction does not degrade correctness for quality-sensitive downstream uses. We identify one concern: routing to smaller models on ostensibly simple steps could amplify bias or reduce robustness in domains where smaller models underperform in ways the minimum-adequate-tier label does not capture. Practitioners should validate routing decisions on their specific domain before deploying AgentRouter in high-stakes contexts.

References

Chen, L., Zaharia, M., and Zou, J. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2305.05176.

Chen, L., Davis, J. Q., Hanin, B., Bailis, P., Zaharia, M., Zou, J., and Stoica, I. Optimizing model selection for compound AI systems. *arXiv preprint arXiv:2502.14815*, 2025. URL https://arxiv.org/abs/2502.14815.

Jin, B., Collins, T., Yu, D., Cemri, M., Zhang, S., Li, M., Tang, J., Qin, T., Xu, Z., Lu, J., Yin, G., Han, J., and Wang, Z. Controlling performance and budget of a centralized multi-agent LLM system with reinforcement learning. *arXiv preprint arXiv:2511.02755*, 2025. URL https://arxiv.org/abs/2511.02755.

Moslem, Y. and Kelleher, J. D. Dynamic model routing and cascading for efficient LLM inference: A survey. *arXiv preprint arXiv:2512.04445*, 2025. URL https://arxiv.org/abs/2512.04445.

Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J. E., Kadous, M. W., and Stoica, I. RouteLLM: Learning to route LLMs with preference data. *arXiv preprint arXiv:2406.18665*, 2024. URL https://arxiv.org/abs/2406.18665.

Panda, P., Magazine, R., Devaguptapu, C., Takemori, S., and Sharma, V. Adaptive LLM routing under budget constraints. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025. URL https://arxiv.org/abs/2508.21141.

Pulishetty, R., Ghantasala, M. K., Dasoju, K. K., Mangwani, N., Garimella, V., Mate, A., Chatterjee, S., Kang, Y., Nosakhare, E., Hasan, S., and Srinivasan, S. One head, many models: Cross-attention routing for cost-aware LLM selection. *arXiv preprint arXiv:2509.09782*, 2025. URL https://arxiv.org/abs/2509.09782.

Qian, C., Liu, Z., Kokane, S., Prabhakar, A., Qiu, J., Chen, H., Liu, Z., Ji, H., Yao, W., Heinecke, S., Savarese, S., Xiong, C., and Wang, H. xRouter: Training cost-aware LLMs orchestration system via reinforcement learning. *arXiv preprint arXiv:2510.08439*, 2025. URL https://arxiv.org/abs/2510.08439.

Salim, M., Latendresse, J., Khatoonabadi, S., and Shihab, E. Tokenomics: Quantifying where tokens are used in agentic software engineering. *arXiv preprint arXiv:2601.14470*, 2026. URL https://arxiv.org/abs/2601.14470.

Sharma, R. and Mehta, M. Small language models for agentic systems: A survey of architectures, capabilities, and deployment trade offs. *arXiv preprint arXiv:2510.03847*, 2025. URL https://arxiv.org/abs/2510.03847.

Su, J., Lan, Q., Xia, Y., Sun, L., Tian, W., Shi, T., Song, X., He, L., and Jingsong, Y. Difficulty-aware agentic orchestration for query-specific multi-agent workflows. In *Proceedings of the ACM Web Conference (WWW)*, 2026. URL https://arxiv.org/abs/2509.11079.

Team, A. Aragog: Agentic rag orchestration with llm-driven model selection. *arXiv preprint arXiv:2511.20975*, 2025. URL https://arxiv.org/abs/2511.20975.

Varangot-Reille, C., Bouvard, C., Gourru, A., Ciancone, M., Schaeffer, M., and Jacquenet, F. Doing more with less: A survey on routing strategies for resource optimisation in large language model-based systems. *arXiv preprint arXiv:2502.00409*, 2025. URL https://arxiv.org/abs/2502.00409.

Zhang, C., Xia, M., Zhang, X., Madrigal, D., Mallick, A., Kessler, S., Ruehle, V., and Rajmohan, S. Budget-aware agentic routing via boundary-guided training. *arXiv preprint arXiv:2512.21227*, 2025. URL https://arxiv.org/abs/2512.21227.