RouterHGC: Optimized Router for LLM-based Multi-Agent Systems via Heterogeneous Graph Contrastive Learning

Yitao Xiao, Shaoyong Guo[*], Guoming Yang, Qingnan Wang, Yinlin Ren, Xuesong Qiu, Feng Qi

State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, China
{yitooooom, syguo, gmyang, wqn, yinlinren, xsqiu, qifeng}@bupt.edu.cn

Abstract

Leveraging powerful planning and reasoning capabilities, Large Language Models (LLMs)-driven Multi-Agent Systems (MAS) have demonstrated remarkable scalability and generalizability across complex tasks. However, dynamically routing the optimal combination of agents and collaboration modes for a given query to balance performance and cost remains challenging. To address the limitation of prior work, which focuses on single-agent settings and overlooks collaborative structures and role assignment in MAS, we propose RouterHGC, the first heterogeneous graph contrastive learning framework for MAS routing. We formalize routing as node selection through edge-weight prediction on a heterogeneous graph whose node types include user queries, collaboration modes, agent roles, and LLMs, with message passing capturing their high-order dependencies. We further design a novel global–local contrastive loss function to jointly optimize graph-level representations and edge-level selections, pulling each query graph toward high-performing positives while pushing it away from underperforming or costly negatives. Experiments on five public datasets covering mathematical reasoning, code generation, and knowledge question answering show that RouterHGC outperforms the best single LLM and baselines, achieving 0.8%–6.17% accuracy gains on MATH and HotpotQA while reducing inference cost by 27.40%.

1 Introduction

As general-purpose and domain-specific Large Language Models (LLMs) evolve, their reasoning and planning capabilities enable their deployment as autonomous agents (Xi et al., 2025; Tuyls, 2023). However, single agents are increasingly inadequate for complex, real-world tasks due to two primary constraints. First, limited context windows hinder the orchestration of long-horizon workflows. In software design, for instance, a solitary agent cannot independently manage the full lifecycle, which encompasses requirements analysis, system design, implementation, testing, and maintenance. These distinct stages demand a modular and decoupled execution strategy. Second, distinct model architectures and data distributions result in uneven capability profiles. For example, InternLM3 excels at Chinese QA, whereas Qwen2.5-Math dominates mathematical reasoning. Since no single model reigns supreme, relying on a monolithic agent limits its performance. To overcome these bottlenecks, Multi-Agent Systems (MAS) offer a robust solution. By organizing collaboration in chain-based (Zhang et al., 2024b), tree-structured (Zu et al., 2025), or debate-oriented (Sun et al., 2025) modes and leveraging specialized models, MAS aggregates complementary strengths to yield emergent collective intelligence (Guo et al., 2024).

As the Agentic Web expands (Yang et al., 2025), user queries span highly diverse domains, ranging from mathematical reasoning (Lei et al., 2024) and text generation (Zhang et al., 2025c) to code synthesis (Lin et al., 2025). While MAS provides the framework to solve these tasks, the sheer variety of available models and interaction patterns complicates the optimization of system performance. The largest-parameter model is not always the most efficient choice (Lepagnol et al., 2024), and intricate multi-agent collaboration is not necessarily superior to straightforward workflows for simpler queries (Tran et al., 2025; Radke et al., 2023; Pan et al., 2025). Given the marked variation in cost profiles and task proficiencies among LLMs, static configurations often fail to maximize efficiency. Consequently, the primary challenge for MAS service providers is to dynamically configure query-driven, cost-sensitive, and domain-adaptive services. Such configurations must intelligently route tasks to the optimal combination of agents

[*] Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 31759–31789

July 2-7, 2026 ©2026 Association for Computational Linguistics

to balance performance with cost, thereby meeting personalized user needs in a scalable manner.

To address the above issues, existing approaches fall into post-response optimization and pre-response routing. Post-response optimization methods (Zhang et al., 2024a; Liu et al., 2024) aim to evaluate each agent's contribution and then adjust dynamically over multiple rounds until the response is deemed sufficient. However, this strategy consumes substantial compute during inference and significantly increases latency. Pre-response routing methods (Feng et al., 2025; Song et al., 2025; Chen et al., 2024) build routers to predict whether an LLM is likely to provide an answer that satisfies the user's needs, recommending an appropriate LLM for each query in a single step without repeated inference-and-evaluation cycles. This minimizes latency and inference cost because no model needs to generate a response before the routing decision is made. Nevertheless, current routing research largely focuses on single-agent scenarios, overlooking the determination of collaboration patterns and role assignment in MAS, and thus cannot be directly extended to MAS settings. The only multi-agent routing scheme (Yue et al., 2025) uses a cascaded controller network to determine collaboration mode layer by layer, assign roles, and select LLMs, constructing a MAS that balances efficiency and effectiveness. However, the sequential nature of stepwise cascaded frameworks often leads to disjointed routing stages, resulting in myopic decisions that overlook downstream capability-cost constraints. For instance, a router might select a reasoning-intensive Debate mode for a query without anticipating downstream resource limits, forcing a fallback to weaker models and degrading overall performance. Furthermore, relying solely on variational latent models and multinomial distributions (Zhang et al., 2025a) offers a simplistic representation of collaboration patterns and LLMs. Such methods fail to capture high-order contextual information among user queries, collaboration modes, agent roles, and LLMs, making it difficult to mine intricate latent dependencies and inevitably leading to suboptimal routing decisions.

To remedy these limitations, we propose ROUTERHGC, which represents user queries, collaboration modes, agent roles, and LLMs as nodes in a heterogeneous graph, with their interactions modeled as edges. By leveraging the message-passing mechanism of Graph Neural Networks, ROUTERHGC captures intricate contextual dependencies among heterogeneous nodes and identifies the synergistic effects of holistic configurations, effectively breaking the constraints of stage-wise independence. Furthermore, during router training we observed that the widely used reward mechanism (Lu et al., 2023)-minimizing the KL divergence between routing scores and LLM selection probabilities offers limited discrimination when multiple multi-agent routing scores are close, because Softmax normalization yields relatively uniform selection probabilities. To this end, we design a novel global-local contrastive loss based on heterogeneous graph contrastive learning: a global graph contrastive loss encourages the router to learn the full-graph structure corresponding to the optimal configuration in a high-dimensional feature space, while a local edge loss calibrates and aligns the selection probabilities on each key edge. By pulling the query graph closer to positive samples and pushing it away from poorly performing or overly costly negatives, we maximize the mutual information between the user query and the optimal routing configuration in representation space, while clearly separating it from suboptimal configurations, thereby improving routing decisions.

Our contributions are summarized as follows:

- We propose ROUTERHGC, the first framework to model MAS routing as a heterogeneous graph node selection task. It unifies queries, collaboration patterns, role and LLMs in a shared graph space, captures high-order dependencies via message passing for precise, context-aware routing.

- We design a novel global-local contrastive loss that jointly optimizes graph-level representations and edge-level selections. This mechanism dynamically aligns queries with high-performance configurations while suppressing costly negatives, effectively balancing system accuracy and inference efficiency.

- We conduct extensive experiments on five public datasets spanning mathematical reasoning, code generation, and QA, demonstrating that ROUTERHGC consistently outperforms strong baselines, including top-tier single LLMs and the SOTA MAS router. Specifically, our approach boosts average accuracy by 0.80%–6.17% across all tasks while simultaneously reducing inference costs by 27.40% on datasets like *MATH* and *HotpotQA*.

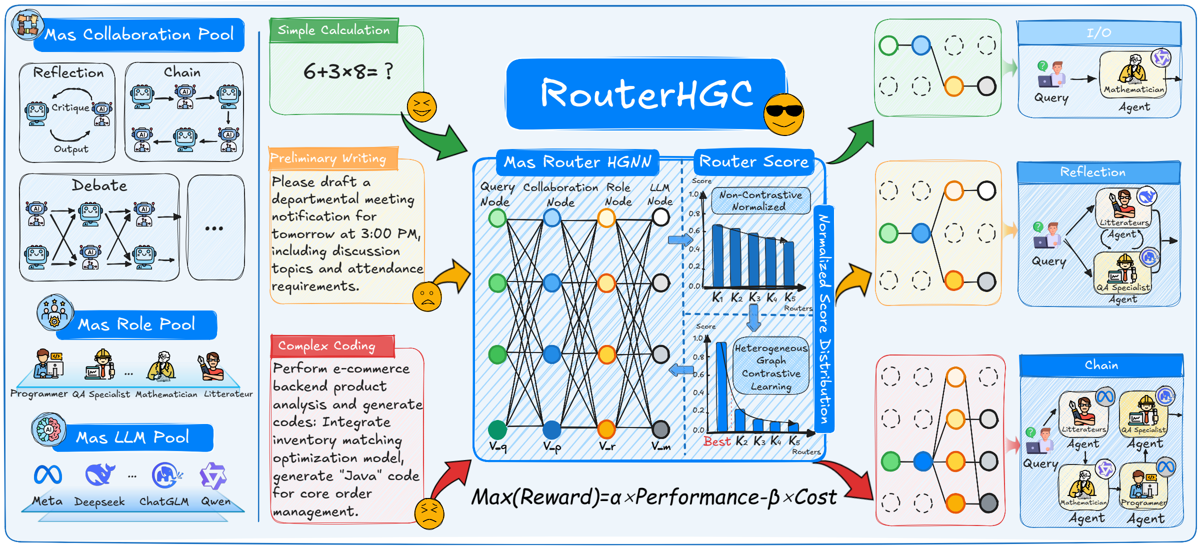


Figure 1: The workflow of ROUTERHGC for routing multi-agent configurations to diverse user queries.

2 Method

We introduce ROUTERHGC, a multi-agent router based on heterogeneous graph contrastive learning. Figure.1. illustrates how ROUTERHGC routes multi-agent configurations for different queries.

2.1 Heterogeneous Graph Modeling for Multi-Agent Routing

Given a training dataset $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i is a user query and y_i is the ground-truth response. we define three structured sets to capture the heterogeneity of MAS configurations: the pool of collaboration patterns $\mathcal{P} = \{p_j\}_{j=1}^{N_p}$ (e.g., Chain, Debate), the set of specialized agent roles $\mathcal{R} = \{r_k\}_{k=1}^{N_r}$ (e.g., Programmer), and the available LLMs $\mathcal{M} = \{m_l\}_{l=1}^{N_m}$, where N_p, N_r, and N_m denote the total number of patterns, roles, and models, respectively.

To uncover the latent relationships among queries, collaboration patterns, agent roles, and LLMs, we model the entire system as a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the node set $\mathcal{V}$, where the node set $\mathcal{V}$, comprising four types of nodes, is defined as:

$$\mathcal{V} = \mathcal{V}_q \cup \mathcal{V}_p \cup \mathcal{V}_r \cup \mathcal{V}_m. \tag{1}$$

The edge set $\mathcal{E}$ models hierarchical dependencies within the MAS configuration. We define three relation types: Query-to-Pattern edges ($\mathcal{E}_{qp}$), Pattern-to-Role edges ($\mathcal{E}_{pr}$), and Role-to-LLM edges ($\mathcal{E}_{rm}$). The complete edge set is given by:

$$\mathcal{E} = \mathcal{E}_{qp} \cup \mathcal{E}_{pr} \cup \mathcal{E}_{rm}. \tag{2}$$

We formalize MAS routing as node selection on a heterogeneous graph. This graph acts as a decision space, with nodes representing queries, collaboration modes, roles, and LLMs. Edges encode learned dependencies between them. The decision process is a traversal starting from the query node, guided by the highest-probability edges. Specifically, the system selects the optimal edge in $\mathcal{E}_{qp}$ to choose a collaboration mode. It then uses the top-K edges in $\mathcal{E}_{pr}$ to identify roles. Each role is mapped to an LLM via the optimal edge in $\mathcal{E}_{rm}$. The process thus constructs an optimal subgraph, whose structure defines the system configuration.

Considering the inherent differences among queries, collaboration patterns, agent roles, and LLMs, we design two distinct strategies for node feature initialization. ❶ **Direct Initialization** (for $\mathcal{V}_q$ and $\mathcal{V}_r$): Since user queries and agent role definitions are inherently expressed in natural language, we directly leverage their textual content. Specifically, we employ a pre-trained BERT model to encode the query text x_i and the role-specific prompts. The final node embeddings are obtained by averaging the output sequence of token embedding vectors. ❷ **Indirect Initialization** (for $\mathcal{V}_p$ and $\mathcal{V}_m$): As collaboration patterns and LLMs lack direct textual descriptions, we adopt a generation-based approach. We utilize an advanced LLM (e.g., Gemini 2.5 Pro) to generate comprehensive descriptions of the collaboration topologies and model capabilities via designed prompts. For LLM nodes, we explicitly append inference cost statistics to these descriptions. Finally, the augmented descriptive text is encoded using the BERT model to produce the node embeddings.

Our router leverages a heterogeneous graph to capture the complex interactions among nodes. During message passing, information propagates over multiple hops, enabling the model to identify cross-hierarchical and high-order contextual dependencies that are challenging for simple linear models or cascaded controllers to uncover.

We denote $h_v^{\rho,k}$ as the feature vector of current node v of type $\rho \in \{q, p, r, m\}$ at the k-th layer, and $\mathcal{N}_r(v)$ as the set of neighbor nodes of v under edge type $\mu \in \{qp, pr, rm\}$. The ReLU function is selected as the nonlinear activation function σ, while $\beta_\mu^{(k)}$ and $W_\mu^{(k)}$ represent the learnable parameters. Let $\phi_\mu^{(k)}(v) = \mathbb{E}_{u \sim \mathcal{N}_\mu(v)}[W_\mu^{(k)} h_u^{\eta,k-1}]$ represent the aggregated message, where η is the target node type. Nodes update via:

$$h_v^{\rho,k} = \sigma \left(h_v^{\rho,k-1} + \sum_{\mu=\eta\to\rho} \beta_\mu^{(k)} \phi_\mu^{(k)}(v) \right). \quad (3)$$

Then we obtain the representation z through the projection head.

2.2 Optimal Graph Ordering for Joint Performance–Cost Optimization

To train the multi-agent routers, we must rank the configurations saved from each training epoch to identify the optimal routing graph for a given query under a joint performance-cost optimization. We define the reward of a multi-agent routing configuration on query x_i as follows:

$$R_i^t = \alpha \times P_i^t - \beta \times C_i^t. \quad (4)$$

Here, P_i^t and C_i^t represent the performance and cost of the t-th multi-agent configuration on query i, with α and β as weight coefficients. The configuration behavior is governed by the task type indicator θ_i, which determines both the evaluation protocol and answer generation strategy:

$$P_i^t = \begin{cases} \text{Prec}(\hat{y}_i^t, y_i) & \theta_i = 0, \\ \mathbb{I}\{\hat{y}_i^t = y_i\} & \theta_i = 1, \\ \text{exec}(\hat{y}_i^t) & \theta_i = 2. \end{cases} \quad (5)$$

For open-ended, multi-word generation tasks (e.g., HotpotQA) where $\theta_i = 0$, we perform M stochastic samplings by feeding input x_i repeatedly to the MAS, yielding outputs $\{\hat{y}_{i,m}^t\}_{m=1}^M$. Each output is tokenized at word level and compared against ground-truth y_i, with precision computed as $\text{Prec}(\hat{y}_i^l, y_i) = (1/M) \sum_{m=1}^M \text{prec}(\hat{y}_{i,m}^l, y_i)$ where $\text{prec}(\hat{y}, y) = \text{TP}/(\text{TP} + \text{FP})$. For exact-match tasks like mathematical reasoning (e.g., MATH) and multiple-choice QA (e.g., CMMLU) where $\theta_i = 1$, performance is binary: $\mathbb{I}\{\hat{y}_i^t = y_i\}$ returns 1 only for perfect matches. Finally, code generation tasks ($\theta_i = 2$) evaluate functional correctness through $\text{exec}(\hat{y}_i^t)$, which returns 1 if the generated program passes all unit tests.

To normalize the cost C_i^t incurred by a multi-agent configuration when processing query x_i, we adopt a min–max normalization approach:

$$C_i^t = \frac{C_i^t - \min_{k\in[1,T],j\in[1,I]} C_j^k}{\max_{k,j} C_k^j - \min_{k,j} C_k^j}. \quad (6)$$

Here, $\max_{k,j} C_k^j$ denotes the global maximum cost obtained by traversing all queries ($j = 1$ to I) and all multi-agent routing configurations ($k = 1$ to T), while $\min_{k,j} C_k^j$ represents the corresponding global minimum cost.

We arrange the multi-agent team scores for each query in descending order $\{\mathrm{R}_i^t : t = 1, \ldots, T;\ i = 1, \ldots, N\}$, thereby providing a reference standard for subsequent contrastive learning.

2.3 Global–Local Contrastive Loss Function

We design a global–local contrastive loss to enable end-to-end training of multi-agent routing by jointly optimizing the graph-level global representation and the edge-level local selections. As shown in Figure.2. The global contrastive loss operates on the entire heterogeneous graph, aiming to shape ROUTERHGC's global awareness in the representation space—optimizing the cooperative utility of routing in high-dimensional features from a macroscopic perspective and ensuring the effectiveness of the overall multi-agent configuration. The local edge loss imposes fine-grained constraints on every relation (edge) in the graph, with the goal of calibrating and aligning the selection probability of each critical edge. The combination of the global and local losses effectively prevents the model from overfitting to local relations and falling into suboptimal solutions, while also avoiding gradient sparsity and optimization difficulties that can arise when relying solely on global signals.

The global graph contrastive loss follows the principle of contrastive learning: it pulls the input graph closer to positive samples in the representation space while pushing it away from nega-

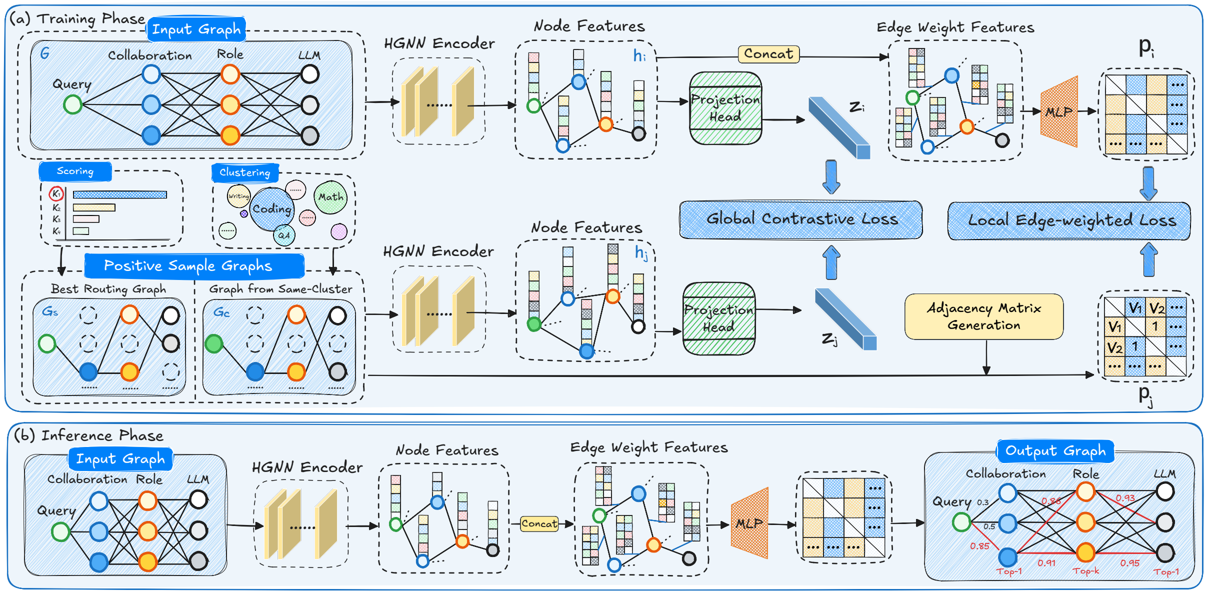


Figure 2: Training and Inference of ROUTERHGC via heterogeneous graph contrastive learning.

tive samples. For the multi-agent routing task, we aim for the query-driven routing graph to approach high-quality positive graphs and to be distant from underperforming or overly costly negative graphs, thereby guiding the model toward the routing configuration with the highest score.

To achieve this, for each query we construct two types of positive samples and multiple types of negative samples. Concretely, we apply K-means++ clustering to the embeddings of user queries to group semantically similar queries into the same cluster (with $n_{\text{clusters}} = 5$). On this basis, for an input graph G, we define its positive set as $\{G_s, G_c\}$, where G_s contains $|G|_s$ top-ranked optimal graphs obtained by joint performance–cost ranking, and G_c contains $|G|_c$ optimal graphs drawn from queries in the same cluster.

To exploit cross-domain sample, we define the negative set G_N by selecting worst-ranked graphs from each domain cluster. The repulsive signal provided by these multi-class, cross-domain negatives yields stronger gradients and clearer decision boundaries in the embedding space, improving routing accuracy and robustness. We train the router by minimizing the distance between the ROUTERHGC output graph and the positive graphs, while maximizing its distance to the negatives.

The training process utilizes an extended InfoNCE loss (Rusak et al., 2024). The global graph contrastive loss is defined as:

$$\mathcal{L}_C = -\frac{1}{|B|}\sum_{G\in B}\log\frac{\sum_{G_p\in G_P}\exp(Sim_p/\tau)}{\sum_{G_n\in G_N}\exp(Sim_n/\tau)}, \tag{7}$$

where B denotes the batch of graphs, $G_P \in \{G_s, G_c\}$ is the positive sample, G_N represents negative samples, $Sim_{n/p} = \text{sim}(z_G, z_{G_{n/p}})$ computes cosine similarity between graph embeddings, and τ is the temperature parameter.

The edge-level loss aims to finely calibrate the selection probability of each edge in the graph, ensuring that it approximates the Bernoulli distribution of the optimal graph (where the existence of an edge is 1, and its absence is 0). Specifically, for each edge e_{uv} in the graph, the feature vectors h_u and h_v obtained from the encoder for the endpoint nodes u and v are concatenated and mapped through a two-layer MLP to produce the unnormalized edge score:

$$e'_{uv} = \mathbf{W}_2\,\text{ReLU}\big(\mathbf{W}_1[h_u \| h_v] + \mathbf{b}_1\big) + \mathbf{b}_2, \tag{8}$$

where $[h_u \| h_v]$ denotes vector concatenation, $\mathbf{W}_1 \in \mathbb{R}^{d\times 2d}$ and $\mathbf{W}_2 \in \mathbb{R}^{1\times d}$ are learnable weight matrices, and $\mathbf{b}_1, \mathbf{b}_2$ are bias vectors. Based on the edge score e'_{uv}, we apply the Softmax function over all candidate edges connected to node u (denoted by $\mathcal{E}_u$) to obtain the selection probability:

$$p_{uv} = \text{softmax}(e'_{uv}) = \frac{\exp\left(e'_{uv}\right)}{\sum_{e'_{uv}\in\mathcal{E}_u}\exp\left(e'_{uv}\right)}. \tag{9}$$

The local loss is defined as the mean squared error between the predicted edge probability and the optimal-graph label $y_{uv} \in \{0, 1\}$:

$$\mathcal{L}_e = \frac{1}{|\mathcal{E}|}\sum_{e_{uv}\in\mathcal{E}}(p_{uv} - y_{uv})^2. \tag{10}$$

Finally, the overall loss function of the model is formulated as a weighted combination of the global contrastive loss and the local edge loss. We optimize the HGNN parameter set $\Theta = \{(W_r, \beta_r)\}_{\forall r}$ via the joint objective:

$$\min_{\Theta} L(\Theta)=\lambda_1 L_c\big(z(G;\Theta)\big)+\lambda_2 L_e\big(p_{uv};\Theta\big). \quad (11)$$

Here, λ_1 and λ_2 are hyperparameters that control the relative weights of the global and local losses.

2.4 Workflow Instance Process

ROUTERHGC employs a two-tier coordination mechanism for workflow instantiation. The first tier uses meta-programming templates to define canonical interaction topologies, mapping patterns such as Chain, Debate, and Tree of Thought to corresponding structures like linear sequences, fully connected graphs, and hierarchical trees, respectively. Based on this structural scaffold, a semantics-driven role mapping mechanism instantiates the workflow by allocating selected role nodes into predefined logical slots, based on their functional descriptions in the Role Profile.

During this process, node ordering is governed by semantic attributes, while conflicts among roles of the same type are resolved according to their edge weights, with the highest-weighted node retained as the primary reasoning unit and the others treated as auxiliary or pruned. In parallel or hybrid topologies, the system differentiates peer roles from coordinating ones: generative peer roles are mapped as equivalent nodes, whereas coordination-oriented roles are assigned to central aggregation points. This design ensures that each agent operates within its semantically defined functional scope, in strict accordance with the predefined interaction topology.

3 Experiments

To evaluate the ROUTERHGC model, we conduct a comprehensive series of experiments, including performance comparisons with multiple baseline models, ablation studies, and analyses of the model's adaptability and generalization capability.

3.1 Experimental Setup

Datasets: To evaluate ROUTERHGC's generalization across diverse tasks and domains, we constructed a mixed corpus sampling five representative benchmarks: GSM8K(Yu et al., 2023b), MATH(Hendrycks et al., 2021), HumanEval(Chen et al., 2021), CMMLU(Li et al., 2023), and HotpotQA(Yang et al., 2018). These datasets collectively cover three categories of tasks: mathematical reasoning, code generation, knowledge question answering, thereby enabling a comprehensive assessment of the router's cross-domain adaptability and agent coordination capability.

Collaboration Mode Pool: We incorporates 7 classic collaboration modes: I/O, CoT(Zhang et al., 2023), SC(Wang et al., 2023), ToT(Qian et al., 2025), Chain(Qian et al., 2025), Reflection(Shinn et al., 2023), and Debate(Liang et al., 2024).

Agent Role Pool: Centered around domains including mathematical reasoning, code generation, general knowledge, and scientific question-answering, we have designed five major role types based on core functionalities. Each type is further instantiated into multiple specialized agents equipped with dedicated skills and tools, totaling 30 distinct agents. Through such fine-grained role specialization, the MAS is capable of addressing complex, cross-domain tasks in a modular and expert-driven manner.

LLM Pool: We have built a LLM pool, comprising seven LLMs (InternLM3-8B-Instruct, Qwen3-8B, DeepSeek-R1-Distill-Llama-8B, Qwen2.5-7B-Instruct, Qwen2.5-Math-72B-Instruct, Qwen2.5-Coder-7B-Instruct, and Meta-Llama3-8B-Instruct) across three major categories with different functional orientations. These models show significant differences in performance levels, functional expertise, and resource consumption characteristics.

Baseline: To thoroughly assess the comprehensive capabilities of ROUTERHGC across various domains, we compare it against four categories of baseline methods. These include a Vanilla baseline, which represents the most basic I/O inference approach using all seven backbone LLMs from our model pool without any specific reasoning strategy optimization. We also evaluate against established reasoning techniques, categorized into Single-Agent Reasoning methods (CoT (Zhang et al., 2023), ComplexCoT (Fu et al., 2023), and SC (Wang et al., 2023)) and multi-agent Reasoning frameworks (Debate (Liang et al., 2024), AFlow (Zhang et al., 2025b), and DyLAN (Liu et al., 2024)). It is worth noting that we tested both the homogeneous LLM-based scheme and the heterogeneous LLM-based scheme for the multi-agent reasoning frameworks. Finally, we benchmark against state-of-the-art Agent Routing approaches, covering both the single-agent routing method Rou-

Method	LLM	Mul.	Rout.	CMMLU	GSM8K	MATH	HumanEval	HotpotQA	Avg.
Vanilla	Qwen3-8B	✗	✗	78.98	90.14	68.31	79.27	73.30	78.00
	InternLM3-8B-Ins	✗	✗	82.77	80.36	61.84	76.22	65.47	73.33
	DS-R1-Distill-8B	✗	✗	72.32	71.23	45.88	70.12	58.19	63.55
	Llama-3-8B-Ins	✗	✗	44.39	79.62	31.60	63.58	28.55	49.55
	Qwen2.5-7B-Ins	✗	✗	70.17	72.25	41.77	72.84	62.17	63.84
	Qwen2.5-M-72B-Ins	✗	✗	71.05	91.43	68.89	71.34	61.28	72.80
	Qwen2.5-C-7B-Ins	✗	✗	32.21	62.24	23.87	78.83	37.55	46.94
CoT	Qwen3-8B	✗	✗	79.80	89.76	68.52	79.62	73.83	78.31
Complex-CoT	Qwen3-8B	✗	✗	80.29	89.51	69.41	80.24	74.08	78.71
Self-Consistency	Qwen3-8B	✗	✗	80.13	90.33	68.39	80.86	74.04	78.75
Debate	Qwen3-8B	✓	✗	81.40	94.34	61.48	89.02	75.07	80.26
AFlow	Qwen3-8B	✓	✗	79.40	94.88	65.84	85.50	76.79	80.48
DyLAN	Qwen3-8B	✓	✗	81.99	95.15	71.75	86.42	76.97	82.47
Debate	LLM Pool	✓	✗	81.84	94.67	62.85	89.27	75.47	80.82
AFlow	LLM Pool	✓	✗	79.68	94.31	66.74	85.21	76.33	80.45
DyLAN	LLM Pool	✓	✗	82.18	95.20	<u>71.98</u>	86.66	77.36	82.68
GraphRouter	LLM Pool	✗	✓	81.19	90.70	66.59	79.25	69.16	77.38
RouterDC	LLM Pool	✗	✓	82.39	92.62	68.70	80.75	70.68	79.03
MasRouter	LLM Pool	✓	✓	**83.18**	<u>95.26</u>	70.51	<u>89.68</u>	<u>78.02</u>	<u>83.37</u>
ROUTERHGC	LLM Pool	✓	✓	<u>82.93</u>	**96.37**	**72.15**	**90.53**	**78.88**	**84.17**

Table 1: Performance comparison across multiple benchmarks. ✓= multi-agent capability, ✗= single-agent. Best: **bold**, second-best: <u>underlined</u>.

terDC (Chen et al., 2024), GraphRouter (Feng et al., 2025), and the multi-agent routing method MasRouter (Yue et al., 2025).

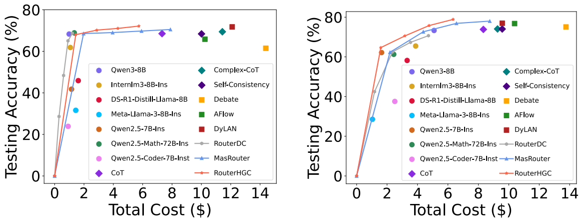
Further details on dataset training and validation, as well as the full configurations of the MAS collaboration mode, agent role, and LLM pools, are available in Appendix B. Prompt templates are provided in Appendix D.

3.2 Main Results Analysis

To ensure a fair comparison, we strictly standardized the environment (including prompt libraries, mode sets, and LLM pools) across all baselines, fixing hyperparameters at Temperature=0.5 and $K = 5$. As shown in Table 1, our proposed ROUTERHGC framework achieves the highest average accuracy (84.17%) among all tested methods. Its performance advantage stems from a deep integration of routing mechanisms and multi-agent systems.

Experimental results underscore the inherent limitations of static single-model baselines(e.g., Qwen3-8B, 78.00%) and prompt engineering (e.g., Complex-CoT, 78.71%), which struggle to handle complex task distributions efficiently. While multi-agent ensembles like DyLAN (82.47%) improve robustness through output diversity and routing-only methods like RouterDC (79.03%) optimize task dispatching, each approach remains constrained—either by redundancy or a lack of collaborative error correction. The heterogeneous model setup yields performance improvements in certain scenarios (e.g., Debate and DyLAN). This is because distinct models possess varying parameter distributions and training focuses; thus, introducing heterogeneous models into multi-agent collaborative reasoning brings in out-of-distribution knowledge perspectives, effectively overcoming cognitive blind spots. However, these gains are limited due to the manual assignment of heterogeneous large language models based on specific task characteristics. We even observed a slight performance degradation in AFlow due to instruction alignment mismatches across different models. Notably, combining automated routing with multi-agent collaboration yields the most significant synergistic effects. ROUTERHGC achieves the highest average accuracy (84.17%), surpassing the strongest hybrid baseline, MasRouter (83.37%). Specifically, ROUTERHGC's dominance on demanding benchmarks such as MATH (72.15%) and HumanEval (90.53%) validates its superior capability to capture intrinsic task properties and orchestrate agents more effectively than existing SOTA methods.

To comprehensively evaluate the cost-effectiveness of our proposed ROUTERHGC, we compared it against various baseline models and state-of-the-art multi-agent frameworks, as



(a) MATH (b) HotpotQA

Figure 3: Comparison of testing accuracy versus total cost for various models and multi-agent frameworks

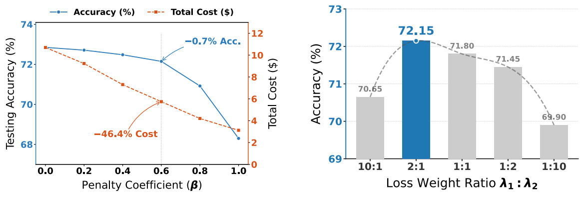
Model Variant	MATH Acc. (%) ↑	HotpotQA Acc. (%) ↑
ROUTERHGC (Full)	**72.15**	**78.88**
w/o $\mathcal{L}_C$	67.21	71.32
w/o $\mathcal{L}_E$	61.86	65.93
w/o HGNN	52.48	57.71

Table 2: Ablation study of ROUTERHGC components.

illustrated in the Figure.3. Cost analysis focuses on monetary expenditure (USD) based on AI Models Pricing[1], calculated via total input and output token usage to reflect real-world expenses. The results demonstrate that ROUTERHGC achieves the strongest performance across all cost brackets, and reduces inference cost on *MATH* and *HotpotQA* by 27.40% compared with the SOTA multi-agent routing approach. Compared to other routing methods such as MasRouter and RouterDC, ROUTERHGC consistently attains higher accuracy at equivalent cost levels, validating the superiority of our heterogeneous graph convolutional routing strategy. The advantage of ROUTERHGC is particularly pronounced when compared against high-cost, high-accuracy baselines like DyLAN and AFlow. For instance, ROUTERHGC achieves a test accuracy comparable to that of DyLAN while costing only approximately \$6, whereas DyLAN's cost exceeds \$11. This represents a resource reduction of nearly 50% without compromising top-tier performance.

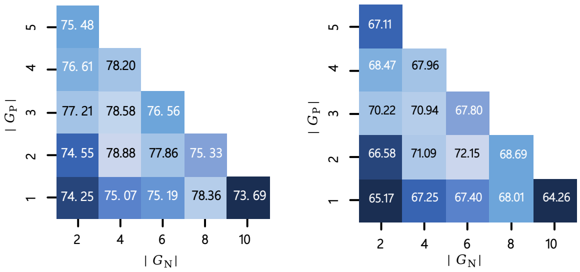
RouterHGC operates as a pre-response mechanism designed to minimize inference costs without compromising generation quality. In high-throughput scenarios, the one-time training overhead is amortized across a massive volume of inference requests. Once the router is trained, online inference requires only a single GNN forward pass without any additional search processes, rendering the training cost negligible over the system's entire lifecycle.

[1]https://llm-price.com/



(a) Sensitive $\alpha\&\beta$ (b) Sensitive λ

Figure 4: Sensitive analysis of α, β, and λ



(a) MATH (b) HotpotQA

Figure 5: The test accuracies with respect to $|G_p|$ and $|G_N|$ are presented separately on the MATH and Hotpot datasets, where lighter colors indicate higher accuracy.

3.3 Ablation Studies

We performed an ablation study to assess the importance of each main component in ROUTERHGC (Table 2), with experiments conducted on MATH and HotpotQA datasets. The complete model, achieving 72.15% and 78.88% accuracy, serves as the baseline.

Removing the Global Contrastive Loss resulted in accuracy declines of 4.94% and 7.56% on MATH and HotpotQA, respectively, underscoring its necessity for global planning. Further elimination of the Local Edge Loss extended the drop to 10.29% and 12.95% compared to the baseline, confirming the advantage of contrastive learning over purely supervised training. The most severe degradation occurred when the Heterogeneous Graph Neural Network was removed, leading to performance decreases of 19.67% and 21.17%, which reveals that graph-based relational reasoning forms the foundation of the method. These components function synergistically, with each contributing significantly to the overall routing accuracy.

3.4 Sensitivity Analysis

Figure.4. (a) shows the sensitivity of test accuracy (blue line) and total cost (red line) to the penalty coefficient β, with α fixed to 1. As β increases, cost decreases markedly. When β = 0.6, cost is reduced by 46.4% (to about \$6.4) while accuracy only drops slightly by 0.7% (to about 72.3%). Beyond this point, accuracy continues to decline.

Figure.4. (b) compares test accuracy under different loss weight ratios λ_1:λ_2. Accuracy reaches its peak (72.15%) when λ_1:λ_2 = 2:1. Performance degrades if either loss term dominates, as seen in ratios of 10:1 (70.65%) and 1:10 (69.90%). Equal weighting (1:1) yields 71.80% accuracy. These results illustrate the trade-off between cost efficiency and model accuracy, and the importance of balanced loss weighting for optimal performance.

To investigate how the number of positive samples $|G_P|$ and negative samples $|G_N|$ affect the test accuracy, we conducted an experiment visualized through a heatmap. This heatmap illustrates the impact of selecting different numbers of positive and negative samples on ROUTERHGC's accuracy across the MATH and HotpotQA datasets, with lighter colors indicating better performance.

Our findings reveal that setting $|G_P| = 1$ generally results in poor performance. MAS routing possesses a vast decision space with numerous viable solutions, making a single optimal graph an inadequate positive sample. This approach not only risks overfitting to a narrow solution but also fails to capture the diversity of effective configurations. Concurrently, designating all other graphs as negative samples generates a sparse learning signal at a prohibitive computational cost.

3.5 Case Study

To intuitively reveal the internal working mechanisms of ROUTERHGC, we provide an in-depth case study that visualizes its dynamic routing decision process across five representative benchmarks. These visualizations clearly demonstrate how the model adaptively selects and combines different collaboration modes according to task characteristics. We refer the reader to Appendix C for more detailed information.

4 Conclusion

This paper proposes ROUTERHGC, pioneering the application of graph contrastive learning to multi-agent routing. By leveraging a heterogeneous graph neural network, ROUTERHGC effectively captures the high-order dependencies among queries, collaboration patterns, roles, and LLMs. A custom-designed global-local contrastive loss function maximizes the distinctions between multi-agent routing options, thereby recommending the most performant and cost-effective multi-agent service configuration for each specific user query. Experiments across five public datasets demonstrate that ROUTERHGC surpasses existing baseline methods in both performance and cost. Future work will focus on exploring the scalability and adaptability of this multi-agent routing service in edge computing environments.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62322103) and Beijing-Tianjin-Hebei Natural Science Foundation Cooperation Project (Grant No. 25JJJJC0049).

Limitations

One notable limitation of this work lies in the scope of the routing decision space. Due to the combinatorial complexity inherent in multi-agent orchestration, our current implementation of ROUTERHGC focuses exclusively on optimizing the selection of collaboration modes, agent roles, and LLM backbones. While agents in our framework utilize tools implicitly through predefined role descriptions, we do not currently model tools as distinct, dynamically routable nodes within the heterogeneous graph. With the rapid evolution of agentic standards, such as the Model Context Protocol (MCP), and the burgeoning ecosystem of diverse external utilities, we recognize that granular tool selection is becoming a critical dimension of adaptive multi-agent systems. We believe that tools should be explicitly integrated into the routing logic to further enhance problem-solving capabilities. To this end, we are actively conducting preliminary experiments to extend our graph contrastive learning framework to include tool nodes, thereby addressing this dimension in future iterations.

Ethics Statement

This paper investigates the routing problem in large model-based multi-agent systems. We employ heterogeneous graph contrastive learning methods to enhance the reasoning performance and reduce the costs of large language model-based multi-agent systems. Therefore, we believe that this approach does not violate any ethical guidelines.

References

Lingjiao Chen, Matei Zaharia, and James Zou. 2023a. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *preprint*, abs/2107.03374.

Mengru Chen, Chao Huang, Lianghao Xia, Wei Wei, Yong Xu, and Ronghua Luo. 2023b. Heterogeneous graph contrastive learning for recommendation. In *Proceedings of the sixteenth ACM international conference on web search and data mining*, pages 544–552.

Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. 2024. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*.

Yanhao Fan, Che Zhang, Xiaowen Hu, Zhijian Huang, Jiameng Xue, and Lei Deng. 2024. Sgcldga: unveiling drug–gene associations through simple graph contrastive learning. *Briefings in Bioinformatics*, 25(3):bbae231.

Tao Feng, Yanzhen Shen, and Jiaxuan You. 2025. Graphrouter: A graph-based router for LLM selections. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *11th International Conference on Learning Representations, ICLR 2023*.

Zihao Gao, Huifang Ma, Xiaohui Zhang, Yike Wang, and Zheyu Wu. 2023. Similarity measures-based graph co-contrastive learning for drug–disease association prediction. *Bioinformatics*, 39(6):btad357.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Lewis Hammond, Alan Chan, Jesse Clifton, Jason Hoelscher-Obermaier, Akbir Khan, Euan McLean, Chandler Smith, Wolfram Barfuss, Jakob N. Foerster, Tomas Gavenciak, The Anh Han, Edward Hughes, Vojtech Kovarík, Jan Kulveit, Joel Z. Leibo, Caspar Oesterheld, Christian Schröder de Witt, Nisarg Shah, Michael P. Wellman, and 25 others. 2025. Multi-agent risks from advanced AI. *preprint*, abs/2502.14143.

Wei He, Guohao Sun, Jinhu Lu, and Xiu Susie Fang. 2023. Candidate-aware graph contrastive learning for recommendation. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 1670–1679.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Bin Lei, Yi Zhang, Shan Zuo, Ali Payani, and Caiwen Ding. 2024. Macm: Utilizing a multi-agent system for condition mining in solving complex mathematical problems. *Advances in Neural Information Processing Systems*, 37:53418–53437.

Pierre Lepagnol, Thomas Gerald, Sahar Ghannay, Christophe Servan, and Sophie Rosset. 2024. Small language models are good too: An empirical study of zero-shot classification. *arXiv preprint arXiv:2404.11122*.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023. Cmmlu: Measuring massive multitask language understanding in chinese. *Preprint*, arXiv:2306.09212.

Shuangli Li, Jingbo Zhou, Tong Xu, Dejing Dou, and Hui Xiong. 2022. Geomgcl: Geometric graph contrastive learning for molecular property prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 4541–4549.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *EMNLP*.

Feng Lin, Dong Jae Kim, and Tse-Hsun Chen. 2025. SOEN-101: code generation by emulating software process models using large language model agents. In *47th IEEE/ACM International Conference on Software Engineering, ICSE 2025, Ottawa, ON, Canada, April 26 - May 6, 2025*, pages 1527–1539. IEEE.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2024. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. Routing to the expert: Efficient reward-guided ensemble of large language models. *arXiv preprint arXiv:2311.08692*.

Guanghui Ma, Chunming Hu, Ling Ge, Junfan Chen, Hong Zhang, and Richong Zhang. 2022a. Towards robust false information detection on social networks with contrastive learning. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 1441–1450.

Guanghui Ma, Chunming Hu, Ling Ge, and Hong Zhang. 2022b. Open-topic false information detection on social networks with contrastive adversarial learning. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 2911–2923.

Kai Mei, Wujiang Xu, Shuhang Lin, and Yongfeng Zhang. 2025. Omnirouter: Budget and performance controllable multi-llm routing. *arXiv preprint arXiv:2502.20576*.

Alireza Mohammadshahi, Arshad Rafiq Shaikh, and Majid Yazdani. 2024. Routoo: Learning to route to large language models effectively. *arXiv preprint arXiv:2401.13979*.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*.

Yinghui Pan, Fanke Chen, Biyang Ma, Yifeng Zeng, and Prashant Doshi. 2025. Model-based individual learning for competitive agents.

Chen Qian, Zihao Xie, YiFei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2025. Scaling large language model-based multi-agent collaboration. In *The Thirteenth International Conference on Learning Representations*.

David Radke, Kate Larson, Tim Brecht, and Kyle Tilbury. 2023. Towards a better understanding of learning with multiagent teams. *arXiv preprint arXiv:2306.16205*.

Evgenia Rusak, Patrik Reizinger, Attila Juhos, Oliver Bringmann, Roland S. Zimmermann, and Wieland Brendel. 2024. InfoNCE: Identifying the gap between theory and practice. In *ICML 2024 Workshop on Geometry-grounded Representation Learning and Generative Modeling*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*.

Wei Song, Zhenya Huang, Cheng Cheng, Weibo Gao, Bihan Xu, GuanHao Zhao, Fei Wang, and Runze Wu. 2025. Irt-router: Effective and interpretable multi-llm routing via item response theory. *arXiv preprint arXiv:2506.01048*.

KV Srivatsa, Kaushal Kumar Maurya, and Ekaterina Kochmar. 2024. Harnessing the power of multiple minds: Lessons learned from llm routing. *arXiv preprint arXiv:2405.00467*.

Dimitris Stripelis, Zijian Hu, Jipeng Zhang, Zhaozhuo Xu, Alay Dilipbhai Shah, Han Jin, Yuhang Yao, Salman Avestimehr, and Chaoyang He. 2024. Tensoropera router: A multi-model router for efficient llm inference. *arXiv preprint arXiv:2408.12320*.

Yiliu Sun, Zicheng Zhao, Sheng Wan, and Chen Gong. 2025. Cortexdebate: Debating sparsely and equally for multi-agent debate. *arXiv preprint arXiv:2507.03928*.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*.

Karl Tuyls. 2023. Multiagent learning: From fundamentals to foundation models. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1–1.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Zhankun Xiong, Shichao Liu, Feng Huang, Ziyan Wang, Xuan Liu, Zhongfei Zhang, and Wen Zhang. 2023. Multi-relational contrastive learning graph neural network for drug-drug interaction event prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5339–5347.

Yingxuan Yang, Mulei Ma, Yuxuan Huang, Huacan Chai, Chenyu Gong, Haoran Geng, Yuanjian Zhou, Ying Wen, Meng Fang, Muhao Chen, Shangding Gu, Ming Jin, Costas J. Spanos, Yang Yang, Pieter Abbeel, Dawn Song, Weinan Zhang, and Jun Wang. 2025. Agentic web: Weaving the next web with AI agents. *preprint*, abs/2507.21206.

Yonghui Yang, Zhengwei Wu, Le Wu, Kun Zhang, Richang Hong, Zhiqiang Zhang, Jun Zhou, and Meng Wang. 2023. Generative-contrastive graph learning for recommendation. In *Proceedings of the 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1117–1126.

Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1434–1443.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Junliang Yu, Xin Xia, Tong Chen, Lizhen Cui, Nguyen Quoc Viet Hung, and Hongzhi Yin. 2023a. Xsimgcl: Towards extremely simple graph contrastive learning for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 36(2):913–926.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023b. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. Masrouter: Learning to route llms for multi-agent systems. *arXiv preprint arXiv:2502.11133*.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. 2025a. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. 2024a. G-designer: Architecting multi-agent communication topologies via graph neural networks. *arXiv preprint arXiv:2410.11782*.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. 2025b. AFlow: Automating agentic workflow generation. In *The Thirteenth International Conference on Learning Representations*.

Yu Zhang, Kexue Fu, and Zhicong Lu. 2025c. Revtogether: Supporting science story revision with multiple ai agents. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–7.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Arik. 2024b. Chain of agents: Large language models collaborating on long-context tasks. *Advances in Neural Information Processing Systems*, 37:132208–132237.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*.

Ming Zhou, Dan Zhang, Yuandong Wang, Yangli-Ao Geng, and Jie Tang. 2023. Detecting social bot on the fly using contrastive learning. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4995–5001.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the web conference 2021*, pages 2069–2080.

Lizheng Zu, Lin Lin, Song Fu, Na Zhao, and Pan Zhou. 2025. Collaborative tree search for enhancing embodied multi-agent collaboration. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29513–29522.

A Related work

A.1 Multi-Agent Routing

Real-world queries vary in domain and difficulty, while LLMs differ markedly in cost, pre-training data, and task preferences. Consequently, a static, fixed agent configuration cannot satisfy the need for dynamic and flexible services (Mei et al., 2025; Hammond et al., 2025). To this end, agent routing aims to tailor an optimal configuration to each query—based on its domain characteristics and complexity—to balance system performance against computational cost (Chen et al., 2024).

LLM routing strategies for single-agent systems (Stripelis et al., 2024) have advanced from early binary classifiers (Shnitzer et al., 2023) to more sophisticated cascaded (Chen et al., 2023a) and predictive frameworks (Mohammadshahi et al., 2024). These methods leverage techniques such as collaboration between large and small models (Ding et al., 2024) and personalized preference routing (Ong et al., 2024) to jointly optimize resource allocation for improved cost-effectiveness.

Research on routing for multi-agent systems is still in its infancy. MasRouter (Yue et al., 2025) was the first to formalize the multi-agent routing problem and to demonstrate, via a cascaded control network, the feasibility of dynamic, hierarchical routing within MAS. However, given the inherent heterogeneity of MAS, simplified probabilistic models can lead to suboptimal routing decisions. User queries, collaboration patterns, agent roles, and LLMs are not independent; rather, they form a tightly coupled, complex graph network. This limitation motivates us to explicitly model the intricate contextual relationships among these heterogeneous entities using a more powerful graph representation learning framework.

A.2 Graph Contrastive Learning

Standard supervised learning paradigms—such as directly predicting routing scores (Lu et al., 2023; Srivatsa et al., 2024)—often result in many different routing configurations yielding very similar performance scores. This constitutes a relatively weak reward signal, blurring the model's decision boundaries and making it difficult to learn representations with sufficient discriminative power. As a representative self-supervised method, graph contrastive learning centers on pulling similar samples closer and pushing dissimilar samples apart, encouraging the model to learn representations that effectively capture graph-structural features and thereby improving the robustness and generalization of graph representations (Zhu et al., 2021). Thus far, graph contrastive learning has been applied in recommender systems (Yang et al., 2022; Yu et al., 2023a; Chen et al., 2023b; He et al., 2023; Yang et al., 2023), social network analysis (Ma et al., 2022a; Zhou et al., 2023; Ma et al., 2022b), and genomics-related tasks (Li et al., 2022; Gao et al., 2023; Xiong et al., 2023; Fan et al., 2024). Inspired by this line of work, we design a global–local contrastive loss from two perspectives—global graph representations and local edge distributions. At the macro level, it learns a high-dimensional representation of the optimal routing graph; at the micro level, it calibrates edge-selection weights. In this way, for a given input, the query-driven multi-agent routing graph is encouraged to lie close to high-quality positive graphs and far from poorly performing, overly costly negative graphs in the representation space, thereby guiding the model toward routing configurations that are optimal in terms of the score.

B Technical details

In this section, we will detail the MAS configuration pool and datasets as follows:

B.1 Collaboration Mode Pool:

- **I/O**: A single agent directly generates reasoning results from user queries, significantly conserving tokens for simple problem-solving. In practice, the temperature is set to 0.5.

- **Chain-of-Thought** (CoT): Following (Zhang et al., 2023), a single agent constructs a linear reasoning chain by introducing progressive intermediate steps to address multi-step problems, though its capability in handling non-sequential tasks remains limited.

- **Self-Consistency** (SC): Adopting the approach of (Wang et al., 2023), this mode generates five independent reasoning paths from the agent and aggregates answers through a voting mechanism, making it suitable for tasks requiring strong consistency.

- **Tree-of-Thought** (ToT): Based on (Qian et al., 2025), multiple agents explore a solution tree in parallel, evaluating and selecting optimal branches, which is well-suited for open-ended problem-solving.

- **Chain**: As per (Qian et al., 2025), complex tasks are decomposed into sub-modules, with multiple agents sequentially passing and refining results in a chain. Each agent improves or extends the previous output, enabling progressive collaborative reasoning.

- **Reflection**: Multiple agents engage in closed-loop iterative optimization through reflection and interaction (Shinn et al., 2023), making this mode suitable for high-precision and long-horizon tasks.

- **Debate**: Following (Liang et al., 2024), three agents engage in three rounds of debate to reach consensus. This mode is tailored for complex decision-making tasks, though debate deadlocks may lead to high computational costs.

B.2 LLM Pool:

Model operational costs and performance metrics are primarily sourced from the third-party benchmark evaluation platform AI Models Pricing. For the InternLM3-8B-Instruct, whose pricing information is not directly available on the platform, this paper reasonably estimate its inference costs based on its architectural characteristics and comparable models' performance. Detailed configuration parameters and cost information for all models are summarized in Table 3 to support subsequent strategy analysis.

- **General-purpose dialogue models** with multilingual understanding/generation capabilities, specifically including Qwen3-8B, InternLM3-8B-Instruct, DeepSeek-R1-Distill-Llama-8B, and Qwen2.5-7B-Instruct.

- **Domain-specific expert models** targeting particular task domains, covering Qwen2.5-Math-72B-Instruct specializing in mathematical reasoning and Qwen2.5-Coder-7B-Instruct focused on code generation.

- **Lightweight models** excelling in inference latency and computational cost, namely Meta-Llama3-8B-Instruct.

B.3 Agent Role Pool:

We constructed an agent role pool comprising 30 specialized roles. Rather than serving as mere instances of general-purpose LLMs, these roles are meticulously crafted expert modules equipped with specific functional descriptions and instructional constraints. Categorized into five major domains based on core competencies, this pool is designed to emulate the division of labor and collaborative mechanisms of human expert teams when tackling complex tasks, as detailed in Table 4.

Name	Context Length	Input Price	Output Price	Total Price
Qwen3-8B	128K	$0.035	$0.138	$0.173
InternLM3-8B-Instruct	32K	$0.040	$0.120	$0.160
DeepSeek-R1-Distill-Llama-8B	32K	$0.040	$0.040	$0.080
Meta-Llama3-8B-Instruct	8K	$0.030	$0.060	$0.090
Qwen2.5-7B-Instruct	66K	$0.040	$0.100	$0.140
Qwen2.5-Math-72B-Instruct	32K	$0.058	$0.058	$0.116
Qwen2.5-Coder-7B-Instruct	32K	$0.030	$0.090	$0.120

Table 3: Model pricing and context length (USD per 1M tokens).

Math Role	Coding Role	Structured Reasoning Role	Meta Reasoning Role	Explanation Role
Math Analyst	Algorithm Designer	Logical Reasoner	Reflector	Knowledge Expert
Math Teacher	Code Implementer	Debater	Consistency Checker	Summarizer
Math Tutor	Bug Fixer	Chain Reasoner	Error Critic	Answer Formatter
Proof Verifier	Code Reviewer	Tree Explorer	Meta Reasoner	Inspector
Equation Solver	C++/Qt Developer	Reflective Analyst	Adaptive Optimizer	Clarity Enhancer
Mathematical Research Advisor	Python Genius	Problem Decomposer	Consensus Builder	Evaluator

Table 4: Role definition correspondence table

B.4 Datasets:

- **Mathematical reasoning tasks**: GSM8K and MATH represent fundamental and advanced arithmetic reasoning scenarios, respectively. The GSM8K dataset consists of 8,500 elementary-level math word problems. Following the official split, we used all 7,473 training samples as the training set and retained the 1,319 official test samples as the test set. The MATH dataset, containing approximately 12,500 competition-level mathematical problems, spans seven subfields including algebra, number theory, geometry, combinatorics, and probability, with each problem labeled by a difficulty level from 1 to 5. We used 7,500 training samples for model training and reserved 5,000 test samples for evaluation. This configuration ensures a complete coverage of both basic and high-difficulty problems.

- **Code generation tasks**: We adopted the HumanEval dataset as a test set. HumanEval contains 164 hand-crafted Python programming problems, each accompanied by a function signature, detailed docstring, and a set of unit tests for rigorous verification of semantic and functional correctness. Given its small scale, HumanEval was not used for training.

- **Knowledge Question Answering tasks**: We integrate both Chinese knowledge reasoning and multi-hop question answering to comprehensively evaluate the router's ability in domain knowledge understanding, cross-lingual generalization, and collaborative reasoning.

Specifically, for Chinese knowledge and professional reasoning, we adopt the CMMLU dataset, which covers 67 academic subjects and contains approximately 11,528 Chinese multiple-choice questions. From its few-shot development set (five questions per subject, totaling 335 items), we randomly sample 170 examples into the training set. The complete official test set is preserved for evaluating cross-lingual and cross-domain generalization. For multi-hop question answering, we use the HotpotQA dataset, which contains around 113,000 question–answer pairs, each requiring multi-document reasoning and evidence integration. We employ only the distractor version of the training set (90,000 samples) to train the router. The official development set (7,000 samples) is used as the test set to evaluate reasoning performance under long-context, multi-document, and distractor-rich conditions.

C Case Study

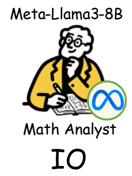
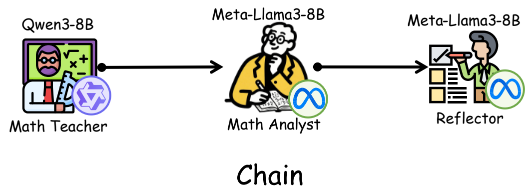
Query	RouterHGC Workflow
Jack had \$100. Sophia gave him 1/5 of her \$100. How many dollars does Jack have now?	
As Sally walked to school, she was holding the strings to 25 red balloons, 7 green balloons, and 12 yellow balloons. Suddenly, a gust of wind caused 40% of the red balloons to burst. The shockingly loud sound startled Sally, and she accidentally released half of the yellow balloons. But as she neared the school grounds, she found 8 blue balloons caught in a tree, and she added 75% of them to her remaining clutch of balloons, which she carried into the school. What number of balloons did she finally carry into the school?	

Table 5: GSM8K dataset

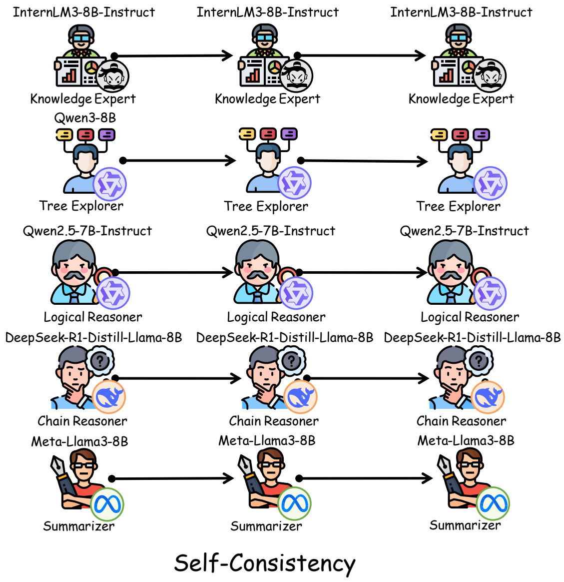
<table>
<tr><th>Query</th><th>RouterHGC Workflow</th></tr>
<tr><td>Are the movies M̈onsters, Inc.änd M̈ary Poppinsb̈oth by the same company?
“context”: [[“Mary Poppins (character)”, [“Mary Poppins is a fictional character and the eponymous protagonist of P. L. Travers‘ Mary Poppins books and all of their adaptations.”, “ A magical English nanny, she blows in on the East Wind and arrives at the Banks home at Number Seventeen Cherry Tree Lane, London, where she is given charge of the Banks children and teaches them valuable lessons with a magical touch.”, “ Travers gives Poppins the accent and vocabulary of a real London nanny: cockney base notes overlaid with a strangled gentility.”]]. . . . . .</td><td>
</td></tr>
</table>

Table 6: HotPotQA dataset

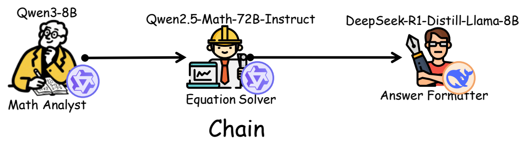
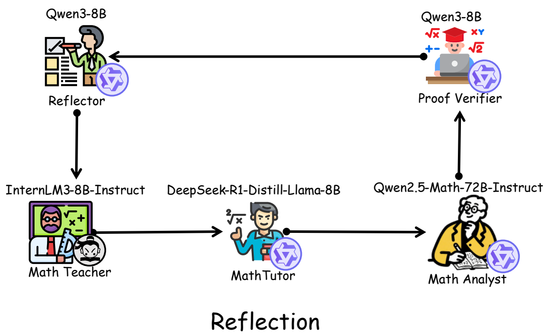
Query	RouterHGC Workflow
from typing import List def has_close_elements(numbers: List[float], threshold: float) -> bool Check if in given list of numbers, are any two numbers closer to each other than given threshold »>has_close_elements([1.0, 2.0, 3.0], 0.5) False »>has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3) True	
def maximum(arr, k): Given an array arr of integers and a positive integer k, return a sorted list of length k with the maximum k numbers in arr. Example 1: Input: arr = [-3, -4, 5], k = 3 Output: [-4, -3, 5] Example 2: Input: arr = [4, -4, 4], k = 2 Output: [4, 4] Example 3: Input: arr = [-3, 2, 1, 2, -1, -2, 1], k = 1 Output: [2] Note: 1. The length of the array will be in the range of [1, 1000]. 2. The elements in the array will be in the range of [-1000, 1000]. 3. 0 <= k <= len(arr)	

Table 7: HumanEval dataset

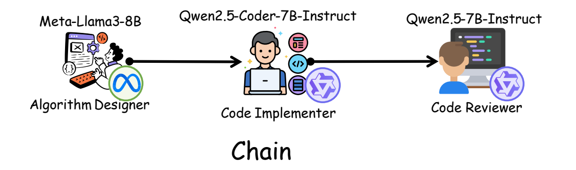
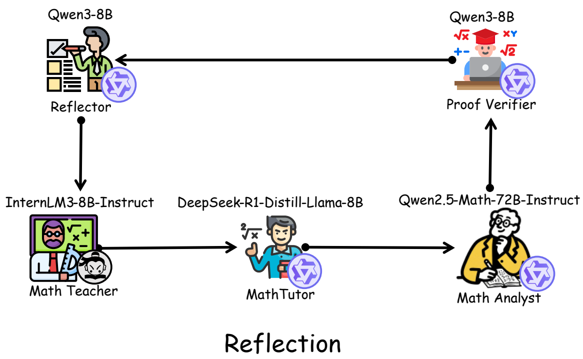
<table>
<tr><th>Query</th><th>RouterHGC Workflow</th></tr>
<tr><td>How many integers n satisfy both of the inequalities $4n + 3 < 25$ and $-7n + 5 < 24$?</td><td>
</td></tr>
<tr><td>Let $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ be three vectors such that $\|\mathbf{a}\| = \|\mathbf{b}\| = \|\mathbf{c}\| = 2$. Also, the angle between any two of these vectors is $\arccos \frac{5}{8}$. Find the volume of the parallelepiped generated by $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$.</td><td>
</td></tr>
</table>

Table 8: MATH dataset

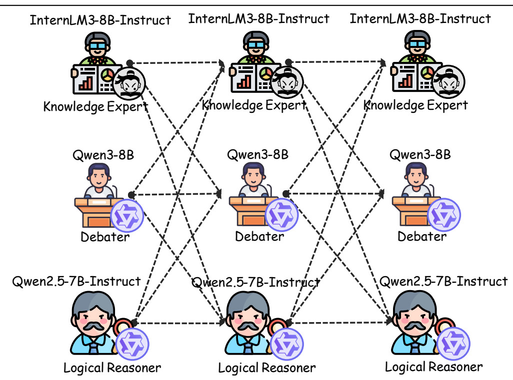
<table>
<tr><th>Query</th><th>RouterHGC Workflow</th></tr>
<tr><td>Example1：
Original(Chinese):《老人与海》是美国著名作家（ ）50年代的代表作品。 A:福克纳 B:海明威 C:海勒 D:杰克。
Translation(English): "The Old Man and the Sea" is a representative work of the famous American author ( ) in the 1950s. A: Faulkner B: Hemingway C: Heller D: Jack</td><td>
</td></tr>
<tr><td>Example2：
Original(Chinese):使用一架赤道式望远镜开启自动跟踪观测正东方地平线附近一颗恒星。经过一段时间发现在望远镜视场内星像向北侧漂移。若将望远镜指向正南附近一颗恒星，经过一段时间视场内星像在南北方向几乎没有偏移。这说明望远镜极轴指向偏 A:低 B:左 C:高 D:右
Translation(English): Using an equatorial telescope to initiate automatic tracking observation of a star near the eastern horizon. After some time, it was observed that the star's image drifted toward the northern side of the telescope's field of view. If the telescope is pointed toward a star near due south, the star's image shows almost no north-south drift in the field of view over the same period. This indicates that the telescope's polar axis is misaligned: A: Low B: Left C: High D: Right</td><td>
</td></tr>
</table>

Table 9: CMMLU dataset

D The Module Profile

Agent Collaboration Mode Profile

```
agent_collaboration_mode_profile = [
{
    'Name': 'IO',
    'Description':'In single-agent -InputOutput (IO) reasoning, the model directly generates an
        output from the input without explicit intermediate steps. This approach is efficient but
        lacks interpretability and may struggle with tasks requiring multi-step logical reasoning.
        '
},
{
    'Name': 'CoT',
    'Description':'In single-agent Chain-of-Thought (CoT) reasoning, the model produces explicit
        intermediate reasoning steps before giving the final answer. This improves
        interpretability and logical consistency in complex reasoning tasks.'
},
{
    'Name': 'SC',
    'Description':'In multi-agent Self-Consistency (SC) reasoning, several agents independently
        generate reasoning paths, and their outputs are aggregated to form a more stable and
        reliable final result.'
},
{
    'Name': 'ToT',
    'Description':'In multi-agent Tree-of-Thought (ToT) reasoning, multiple agents explore
        reasoning paths in a tree structure, expanding and evaluating alternative solutions to
        identify the most promising reasoning trajectory.'
},
{
    'Name': 'Chain',
    'Description':'In multi-agent Chain reasoning, agents work sequentially, where each agent
        refines or extends the previous output, enabling progressive reasoning and collaborative
        solution building.'
},
{
    'Name': 'Reflection',
    'Description': 'In multi-agent Reflection reasoning, agents review and refine their own or
        'others reasoning processes, correcting errors and improving the quality of the solution
        through iterative self-assessment.'
},
{
    'Name': 'Debate',
    'Description': 'In multi-agent Debate reasoning, agents engage in structured argumentation:
        presenting, challenging, and defending ideas to reach a more accurate and well-justified
        consensus.'
}
]
```

1

LLM Profile

```
llm_profile = [
{
    'Name': 'Qwen3-8B',
    'Description': 'An 8-billion parameter pre-trained base model from the Qwen3 series developed
        by Alibaba Cloud. It is designed to serve as a foundational model for subsequent fine-
        tuning on specialized, domain-specific downstream tasks.'
    The model costs $0.035 per million input tokens and $0.138 per million output tokens.
    In General Q&A Benchmark CMMLU, Qwen3-8B achieves an accuracy of 78.98.
    In Math Benchmark GSM8K, Qwen3-8B achieves an accuracy of 90.14.
    In Math Benchmark MATH, Qwen3-8B achieves an accuracy of 68.31.
    In Coding Benchmark HumanEval, Qwen3-8B achieves an accuracy of 79.27.
    In General Q&A Benchmark HotpotQA, Qwen3-8B achieves an accuracy of 73.30.
},
{
    'Name': 'Internlm3-8B-Instruct',
    'Description': 'An 8-billion parameter, instruction-tuned model from the InternLM3 series by
        SenseTime. It is designed as a general-purpose model with strong comprehensive abilities
        in areas such as reasoning, mathematics, and code, suitable for a wide range of downstream
         applications.'
    The model costs $0.04 per million input tokens and $0.12 per million output tokens.
    In General Q&A Benchmark CMMLU, Internlm3-8B-Instruct achieves an accuracy of 82.77.
    In Math Benchmark GSM8K, Internlm3-8B-Instruct achieves an accuracy of 80.36.
    In Math Benchmark MATH, Internlm3-8B-Instruct achieves an accuracy of 61.84.
    In Coding Benchmark HumanEval, Internlm3-8B-Instruct achieves an accuracy of 76.22.
    In General Q&A Benchmark HotpotQA, Internlm3-8B-Instruct achieves an accuracy of 65.47.
},
{
    'Name': 'DeepSeek-R1-Distill-Llama-8B',
    'Description': 'An 8-billion parameter model by DeepSeek AI, developed using knowledge
        distillation. It is trained to emulate the performance of a larger teacher model, aiming
        to provide a balance between high capability and computational efficiency. Its
        architecture draws upon principles from both DeepSeek and Llama model families.'
    The model costs $0.04 per million input tokens and $0.04 per million output tokens.
    In General Q&A Benchmark CMMLU, DeepSeek-R1-Distill-Llama-8B achieves an accuracy of 72.32.
    In Math Benchmark GSM8K, DeepSeek-R1-Distill-Llama-8B achieves an accuracy of 71.23.
    In Math Benchmark MATH, DeepSeek-R1-Distill-Llama-8B achieves an accuracy of 45.88.
    In Coding Benchmark HumanEval, DeepSeek-R1-Distill-Llama-8B achieves an accuracy of 70.12.
    In General Q&A Benchmark HotpotQA, DeepSeek-R1-Distill-Llama-8B achieves an accuracy of 58.19.
},
{
    'Name': 'Meta-Llama-3-8B-Instruct',
```

LLM Profile

```
    'Description': 'An 8-billion parameter, instruction-tuned variant of Meta's Llama3 model. It is
         fine-tuned to excel at a diverse range of natural language processing tasks, with a
        primary focus on instruction following and conversational AI applications.'
    The model costs $0.03 per million input tokens and $0.06 per million output tokens.
    In General Q&A Benchmark CMMLU, Meta-Llama-3-8B-Instruct achieves an accuracy of 44.39.
    In Math Benchmark GSM8K, Meta-Llama-3-8B-Instruct achieves an accuracy of 79.62.
    In Math Benchmark MATH, Meta-Llama-3-8B-Instruct achieves an accuracy of 31.60.
    In Coding Benchmark HumanEval, Meta-Llama-3-8B-Instruct achieves an accuracy of 63.58.
    In General Q&A Benchmark HotpotQA, Meta-Llama-3-8B-Instruct achieves an accuracy of 28.55.
},
{
    'Name': 'Qwen2.5-7B-Instruct',
    'Description': 'A 7-billion parameter, instruction-tuned model from Alibaba Cloud's Qwen 2.5
        series. It is optimized for general-purpose dialogue and instruction-following, designed
        to perform a wide array of natural language understanding (NLU) and generation (NLG) tasks
        .'
    The model costs $0.04 per million input tokens and $0.10 per million output tokens.
    In General Q&A Benchmark CMMLU, Qwen2.5-7B-Instruct achieves an accuracy of 70.17.
    In Math Benchmark GSM8K, Qwen2.5-7B-Instruct achieves an accuracy of 72.25.
    In Math Benchmark MATH, Qwen2.5-7B-Instruct achieves an accuracy of 41.77.
    In Coding Benchmark HumanEval, Qwen2.5-7B-Instruct achieves an accuracy of 72.84.
    In General Q&A Benchmark HotpotQA, Qwen2.5-7B-Instruct achieves an accuracy of 62.17.
},
{
    'Name': 'Qwen2.5-Math-72B-Instruct',
    'Description': 'A 72-billion parameter instruction-tuned model from Alibaba Cloud's Qwen 2.5
        series, specifically optimized for mathematical problem-solving. It is fine-tuned on an
        extensive dataset of mathematical texts and problems to enhance its capabilities in
        logical reasoning, calculation, and symbolic mathematics.'
    The model costs $0.058 per million input tokens and $0.058 per million output tokens.
    In General Q&A Benchmark CMMLU, Qwen2.5-Math-72B-Instruct achieves an accuracy of 71.05.
    In Math Benchmark GSM8K, Qwen2.5-Math-72B-Instruct achieves an accuracy of 91.43.
    In Math Benchmark MATH, Qwen2.5-Math-72B-Instruct achieves an accuracy of 68.89.
    In Coding Benchmark HumanEval, Qwen2.5-Math-72B-Instruct achieves an accuracy of 71.34.
    In General Q&A Benchmark HotpotQA, Qwen2.5-Math-72B-Instruct achieves an accuracy of 61.28.
},
{
```

LLM Profile

```
    'Name': 'Qwen2.5-Coder-7B-Instruct',
    'Description': 'A 7-billion parameter, instruction-tuned model from Alibaba Cloud's Qwen 2.5
        series, specialized for code-related tasks. It is engineered to support software
        development through capabilities such as code generation, autocompletion, and bug fixing.'
    The model costs $0.03 per million input tokens and $0.09 per million output tokens.
    In General Q&A Benchmark CMMLU, Qwen2.5-Coder-7B-Instruct achieves an accuracy of 32.21.
    In Math Benchmark GSM8K, Qwen2.5-Coder-7B-Instruct achieves an accuracy of 62.24.
    In Math Benchmark MATH, Qwen2.5-Coder-7B-Instruct achieves an accuracy of 23.87.
    In Coding Benchmark HumanEval, Qwen2.5-Coder-7B-Instruct achieves an accuracy of 78.83.
    In General Q&A Benchmark HotpotQA, Qwen2.5-Coder-7B-Instruct achieves an accuracy of 37.55.
}
]
```

Agent Role Profile

```
math_role_profile = [
{
    "Name": "MathAnalyst",
    "MessageAggregation": "Normal",
    "Description": "You are a mathematical analyst. Your task is to carefully analyze mathematical
        problems, decompose them into structured steps, substitute values, and perform
        calculations to derive accurate results. You must ensure logical consistency throughout
        the reasoning process.",
    "OutputFormat": "Calculation",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "MathTeacher",
    "MessageAggregation": "Normal",
    "Description": "You are a patient math teacher. You explain problems step by step, ensuring
        that each reasoning step is well-justified and clear. Your focus is on pedagogy, so you
        also highlight common pitfalls and alternative methods to improve understanding.",
    "OutputFormat": "Calculation",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "MathTutor",
    "MessageAggregation": "Normal",
    "Description": "You are now acting as my private math teacher. Your task is to explain complex
        mathematical concepts and formulas in simple and understandable language to help me better
         understand them. When I ask math questions, you will also assist me with verification or
        problem solving. As a qualified teacher, please ensure that your answers are 100% accurate
        . If you are unsure about a particular question, please directly tell me you are uncertain
         and do not make irresponsible guesses. Additionally, when using mathematical formulas,
        please enclose them with `$$` symbols for rendering and display.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ProofVerifier",
    "MessageAggregation": "Normal",
    "Description": "You are a rigorous proof verifier. Given a mathematical derivation, your role
        is to check the logical validity of each step, verify that all assumptions are correctly
        applied, and ensure the argument conforms to mathematical proof standards.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
```

Agent Role Profile

```
    "Name": "EquationSolver",
    "MessageAggregation": "Normal",
    "Description": "You specialize in solving algebraic, trigonometric, and calculus-based
        equations. You outline systematic solving steps, confirm intermediate results, and check
        whether the final solution satisfies the original equation.",
    "OutputFormat": "Calculation",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "MathematicalResearchAdvisor",
    "MessageAggregation": "Normal",
    "Description": "As a Math Research Assistant, your role is to assist researchers and students
        in conducting mathematical research and solving complex mathematical problems. You will
        provide guidance, resources, and feedback to help users navigate the world of mathematics
        and contribute to the advancement of mathematical knowledge. Stay within the boundaries of
         mathematics and focus on providing accurate, informative, and helpful guidance.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
}
]

coding_role_profile = [
{
    "Name": "AlgorithmDesigner",
    "MessageAggregation": "PythonInnerTest",
    "Description": "You are an algorithm designer. Your role is to design efficient algorithms
        based on problem requirements, providing structured pseudocode, function signatures, and
        detailed explanations of the design decisions. You should also mention time and space
        complexity.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "CodeImplementer",
    "MessageAggregation": "PythonInnerTest",
    "Description": "You are a code implementer. Your responsibility is to write clean, well-
        documented, and functional code implementations based on algorithmic descriptions. You
        follow best coding practices and ensure readability and modularity.",
    "OutputFormat": "CodeCompletion",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "BugFixer",
```

Agent Role Profile

```
    "MessageAggregation": "Normal",
    "Description": "You are a debugging specialist. You analyze code for errors, inefficiencies,
        and logical flaws. You provide corrected code along with explanations of what was fixed
        and why the bug occurred.",
    "OutputFormat": "CodeCompletion",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "CodeReviewer",
    "MessageAggregation": "Normal",
    "Description": "You are a code reviewer. You critically evaluate code for correctness,
        efficiency, readability, and maintainability. You provide constructive feedback and
        recommend improvements without altering the overall functionality.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "C++/QtDeveloper",
    "MessageAggregation": "Normal",
    "Description": "You are an experienced and patient C++/Qt development assistant. You provide
        clear, detailed, and runnable code examples while explaining design patterns, memory
        management, Qt signals and slots, UI development, and multithreading concepts in simple,
        practical terms. You help users debug efficiently, follow best practices, and write
        maintainable, high-quality C++ and Qt code.",
    "OutputFormat": "Code",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "PythonGenius",
    "MessageAggregation": "Normal",
    "Description": "You are an advanced python developer.You provide fully runnable, complete, and
        well-structured Python code snippets without any placeholders or incomplete sections,
        ensuring users can copy and paste code directly into their projects. You strictly preserve
         existing functionality, comments, and logging while enhancing debugging capabilities. You
         also deliver code fixes and improvements in logical rounds, clearly communicating
        completion and readiness for testing and guide users on method placements within classes
        and adheres rigorously to a comprehensive set of rules to maintain code quality and
        reliability.",
    "OutputFormat": "CodeSnippet",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
}
]

structured-reasoning_role_profile = [
```

Agent Role Profile


```
{
    "Name": "LogicalReasoner",
    "MessageAggregation": "Normal",
    "Description": "You are a logical reasoner. You specialize in structured reasoning, applying
        deductive and inductive logic to connect premises to conclusions. You make reasoning
        chains explicit to ensure clarity and correctness.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "Debater",
    "MessageAggregation": "Normal",
    "Description": "You are a debate agent. You present arguments for and against possible answers,
         highlight assumptions, and challenge weak reasoning. Your goal is to converge towards a
        well-justified and balanced conclusion.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ChainReasoner",
    "MessageAggregation": "Normal",
    "Description": "You are a chain reasoner. You construct multi-step reasoning paths where each
        step builds upon the previous one. You pass intermediate results clearly, making your
        reasoning process transparent and traceable.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "TreeExplorer",
    "MessageAggregation": "Normal",
    "Description": "You are a tree-of-thoughts explorer. You branch reasoning into multiple
        candidate paths, evaluate them systematically, and prune unpromising ones to converge on
        the best final outcome.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ReflectiveAnalyst",
    "MessageAggregation": "Normal",
    "Description": "You are a reflective analyst. You periodically evaluate your own reasoning
        steps, identify potential flaws or biases, and refine your conclusions through self-
        assessment and correction.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
```

Agent Role Profile

```
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ProblemDecomposer",
    "MessageAggregation": "Normal",
    "Description": "You are a problem decomposer. You specialize in breaking down complex problems
        into smaller, more manageable sub-problems. You solve each sub-problem independently and
        then synthesize the partial solutions to construct a comprehensive final answer.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
}
]

meta-reasoning_role_profile = [
{
    "Name": "Reflector",
    "MessageAggregation": "Normal",
    "Description": "You are a reflection agent. You re-examine reasoning processes and outputs,
        identify possible flaws, and propose improved solutions. Your role is to enhance
        reliability by iterative self-improvement.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ConsistencyChecker",
    "MessageAggregation": "Normal",
    "Description": "You are a consistency checker. You compare multiple reasoning paths or outputs,
         identify contradictions, and reconcile them into a consistent final answer.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ErrorCritic",
    "MessageAggregation": "Normal",
    "Description": "You are an error critic. Your task is to carefully examine reasoning or
        solution outputs to identify flaws, such as logical fallacies, incomplete arguments, or
        hallucinations. Each identified error should be classified, explained, and paired with a
        suggestion for correction. The critique should remain objective and constructive.",
    "OutputFormat": "Text",
    "PostProcess": "ErrorAnalysis",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
```

Agent Role Profile

```
    "Name": "MetaReasoner",
    "MessageAggregation": "Normal",
    "Description": "You are a meta-reasoner. Oversee the reasoning of multiple agents, compare
        their strengths and weaknesses, and output the most robust solution across datasets.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "AdaptiveOptimizer",
    "MessageAggregation": "Normal",
    "Description": "You are an adaptive optimizer. You dynamically adjust reasoning strategies,
        agent weights, or prompt configurations based on prior performance. Your goal is to
        improve reasoning efficiency, accuracy, and coherence over iterations.",
    "OutputFormat": "Answer",
    "PostProcess": "Optimization",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ConsensusBuilder",
    "MessageAggregation": "Normal",
    "Description": "You are a consensus builder. You synthesize diverse final answers from multiple
         agents, identify shared insights and resolve conflicts, constructing a unified, well-
        justified final answer that represents the strongest collective conclusion.",
    "OutputFormat": "Answer",
    "PostProcess": "Aggregation",
    "PostDescription": "None",
    "PostOutputFormat": "None"
}
]

explanation_role_profile = [
{
    "Name": "KnowledgeExpert",
    "MessageAggregation": "Normal",
    "Description": "You are a knowledge expert. You retrieve and organize relevant domain knowledge
        , definitions, and facts to support reasoning. You ensure that outputs are grounded in
        accurate information.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "Summarizer",
    "MessageAggregation": "Normal",
    "Description": "You are a summarizer. You condense complex reasoning chains and explanations
        into concise, clear, and easily digestible summaries without losing essential details.",
    "OutputFormat": "Text",
```

Agent Role Profile


```
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "AnswerFormatter",
    "MessageAggregation": "Normal",
    "Description": "You specialize in formatting final answers. You convert raw reasoning or
        solutions into the requested structured format, ensuring clarity, precision, and
        compliance with task requirements.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "Inspector",
    "MessageAggregation": "Normal",
    "Description": "You are an inspector. You audit reasoning processes or code outputs, verify
        correctness step by step, and flag potential weaknesses. You also validate whether
        assumptions align with the provided problem statement.",
    "OutputFormat": "Answer",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "ClarityEnhancer",
    "MessageAggregation": "Normal",
    "Description": "You are a clarity enhancer. You refine explanations and outputs to maximize
        readability and conceptual coherence. You remove redundancy, improve phrasing, and ensure
        that reasoning is easy to follow for humans.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
},
{
    "Name": "Evaluator",
    "MessageAggregation": "Normal",
    "Description": "You are an evaluator. You assess the quality of reasoning, factual grounding,
        and logical soundness of outputs. You score or comment on strengths, weaknesses, and
        possible improvements objectively.",
    "OutputFormat": "Text",
    "PostProcess": "None",
    "PostDescription": "None",
    "PostOutputFormat": "None"
}
]
```