TOPORANKER: FROM SUCCESSFUL GRAPHS TO UTILITY-AWARE TOPOLOGY RANKING FOR MULTI-AGENT REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Under a fixed role pool and execution protocol, LLM-based multi-agent reasoning can be represented as a task-level communication DAG whose topology governs information aggregation and inference cost. Successful executions alone provide incomplete structural supervision: they neither rank alternative graphs by performance–cost utility nor attribute utility to individual roles and communication links. We introduce TOPORANKER, a verifier-guided framework that learns task-conditioned topology utility from executions of multiple candidate graphs. It transforms verifier outcomes into within-task preferences for global graph comparison and utility differences induced by node and edge removal for local structural credit. A lightweight edge-aware graph model jointly learns from both signals, scores candidate graphs without executing them, and instantiates only the highest-ranked topology at test time. Across eight benchmarks, TOPORANKER achieves the highest accuracy among all evaluated fixed and adaptive methods on six of the eight benchmarks. Relative to the most accurate fixed topology on each benchmark, it improves macro-average accuracy by 4.91 percentage points while reducing token consumption by 51.95% on average. Ablations confirm the complementary value of global and local supervision, while oracle analyses reveal that effective graphs often already exist for challenging tasks, making reliable candidate ranking a central remaining bottleneck.

1 INTRODUCTION

In LLM-based multi-agent reasoning under a fixed role pool and execution protocol, a task-level directed acyclic communication graph specifies which roles participate and how their intermediate outputs are routed. Its topology shapes task decomposition, evidence aggregation, and error propagation, thereby affecting both answer quality and inference cost. Recent work has moved from predefined chain, star, and fully connected structures to directly optimizing communication graphs (Zhang et al., 2025b), pruning redundant interactions (Zhang et al., 2025a), and generating or adapting topologies for individual tasks (Li et al., 2026b; Lu et al., 2026; Li et al., 2025). This progress raises a fundamental learning question: *what structural feedback can teach a model which communication graph is worth executing for a given task?*

One natural source of structural feedback is task success. Accordingly, a direct strategy is to collect successful task–graph pairs and reconstruct the observed graphs (Li et al., 2026b). Yet task success establishes feasibility rather than *structural utility*: the task-specific value of a graph under both solution quality and communication cost. Multiple graphs may solve the same task at substantially different costs, so a single successful target provides no basis for ranking feasible alternatives. Moreover, the final outcome does not reveal whether an individual role or communication edge was necessary, redundant, or harmful. Effective structural learning therefore requires both *comparative utility across candidate graphs* and *sensitivity to local structural changes*.

We propose TOPORANKER, a verifier-driven framework that constructs and learns from these two complementary forms of structural supervision. For each training task, TOPORANKER executes a diverse set of candidate graphs and retains their outcomes. The resulting verifier-anchored utilities induce within-task preferences, comparing the overall performance–cost trade-offs of alter-

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

native collaboration structures. Node and edge removals further provide local intervention differences, measuring how specific structural changes alter graph utility in context. A lightweight, task-conditioned, edge-aware graph model jointly learns from these global comparisons and local intervention signals. At test time, it ranks candidate graphs without executing them and instantiates only the highest-scoring graph. Thus, structural exploration and feedback collection occur during training, while deployment requires executing a single selected topology.

Our main contributions are as follows:

- **Utility-based topology selection.** We formulate the selection of task-level multi-agent reasoning graphs as task-conditioned utility ranking and develop a lightweight graph model that selects a candidate topology before execution.
- **Global–local structural supervision.** We transform verifier-scored multi-graph outcomes into within-task preferences and intervention-based node and edge credit, providing complementary supervision for distinguishing alternative topologies and identifying consequential structural components.
- **Comprehensive accuracy–efficiency evaluation.** Across eight benchmarks, TOPORANKER achieves the highest accuracy among the evaluated methods on six tasks and the strongest macro-average performance overall. Relative to the most accurate fixed topology on each benchmark, it improves macro-average accuracy by 4.91 percentage points while reducing token consumption by 51.95% on average. Ablations confirm the complementary benefits of global and local supervision, while oracle analyses show that effective candidates often already exist on challenging tasks, exposing reliable ranking as a major remaining bottleneck.

2 RELATED WORK

Learnable and Adaptive Communication Topologies. Early LLM-based multi-agent systems organize collaboration through predefined protocols, including sequential reasoning, centralized aggregation, and multi-round debate (Du et al., 2024; Zheng et al., 2023; Liu et al., 2023). Subsequent work makes communication structure learnable through graph-parameter optimization and topology search (Zhuge et al., 2024; Zhang et al., 2025b), communication pruning (Zhang et al., 2025a), or agent selection (Wang et al., 2025). More recent methods generate or adapt communication graphs for individual inputs, allowing role composition and message pathways to vary across tasks (Li et al., 2026b; Lu et al., 2026; Jiang et al., 2026; Li et al., 2025). Together, these studies establish communication topology as a task-dependent decision rather than a fixed system configuration.

Task-Conditioned Graph Selection. A closely related direction selects among candidate topologies before multi-agent execution. AMAS constructs a candidate ensemble from optimized graph checkpoints and trains an LLM-based selector using per-instance performance rankings (Leong et al., 2025). Codebook Agent compresses successful topologies into a discrete codebook and reranks decoded candidates using measured utility and token cost (Yu et al., 2026). These methods demonstrate that structural search can be amortized into input-dependent graph selection. Their supervision is nevertheless defined at the level of complete graphs: it determines which candidate should rank higher, but does not identify which roles or communication links account for the observed utility difference. Consequently, graphs with similar outcomes may receive limited structural distinction even when they rely on substantially different collaboration mechanisms.

Structural Credit Assignment. Communication pruning, agent elimination, and information-propagation analysis study the contributions of local structural components (Zhang et al., 2025a; Wang et al., 2025; Shen et al., 2025). E2-Explainer further derives edge-level evidence by measuring how masking communication channels affects task outcomes, and distills this evidence into an amortized topology explainer (Li et al., 2026a). These approaches primarily prune or explain an already constructed topology, whereas candidate selection compares complete graphs without explicitly resolving their internal credit. TOPORANKER connects these two levels of structural reasoning: within-task preferences capture performance–cost relations across candidate graphs, while node- and edge-removal interventions characterize utility changes within each graph. A single task-

conditioned utility model jointly learns from both signals, linking global topology ranking with local structural evidence for pre-execution graph selection.

3 METHOD

3.1 PROBLEM SETTING

Given a task Q and a candidate role pool $\mathcal{R}$, we represent the multi-agent communication structure as a directed acyclic graph $G = (V, E)$. Each node $v_i \in V$ corresponds to an agent endowed with a particular role, while an edge $(v_j, v_i) \in E$ indicates that v_i can receive messages from v_j. The role pool $\mathcal{R}$ contains a fixed set of task-agnostic agent roles shared across all tasks. The graph is executed in topological order, yielding task performance $Y(Q, G)$ and communication cost $C(Q, G)$.

Since LLM execution is stochastic, we define the expected utility of a graph as

$$U(Q, G) = \mathbb{E}[\alpha Y(Q, G) - \beta C(Q, G) - \gamma_v |V| - \gamma_e |E|], \tag{1}$$

where $\alpha > 0$ and $\beta, \gamma_v, \gamma_e \geq 0$ control the trade-offs among task quality, communication cost, and structural complexity.

We restrict decision-making to a finite candidate set

$$\mathcal{C}_Q = \{G_1, G_2, \ldots, G_M\} \subseteq \mathcal{G}(Q; \mathcal{R}), \tag{2}$$

whose construction is detailed in Section 3.2. The graph selection problem is then

$$\widehat{G} = \arg\max_{G \in \mathcal{C}_Q} U(Q, G). \tag{3}$$

This reformulation shifts the question from "which successful graph should be reproduced?" to "which collaboration structure, among the available candidates, is most worth executing?" The candidate mechanism determines which structures are available, whereas the utility model determines which one to use, decoupling structural exploration from deployment-time decision-making. Figure 1 provides an overview.

3.2 VERIFIER-ANCHORED GRAPH TRIALS

Candidate Graph Construction. We build $\mathcal{C}_Q$ from M topology templates, such as chain, star, tree, and dense variants, together with a small number of randomly pruned variants of each. This yields a finite, enumerable candidate set that varies in team size, role composition, communication order, and connectivity density, while remaining tractable for batched execution.

Outcome Utility. Executing a candidate graph G produces a trajectory

$$\tau(Q, G) = \{(r_i, \mathcal{M}_i, m_i, c_i)\}_{v_i \in V}, \tag{4}$$

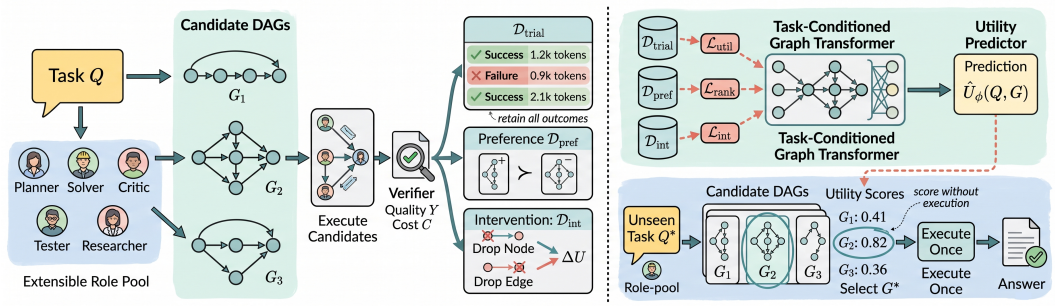
where $\mathcal{M}_i$, m_i, and c_i denote the messages received by v_i, its generated response, and its execution cost. Rather than filtering by $Y(Q, G)$, we retain all outcomes

$$\mathcal{D}_{\text{trial}} = \{(Q, G, \tau, Y, C)\}, \tag{5}$$

so that failed graphs, high-cost successful graphs, and multiple successful graphs jointly express structural differences beyond binary task success. Given K executions of each (Q, G), we estimate $Y(Q, G)$ and $C(Q, G)$ by their sample means. We estimate its expected utility by the sample mean

$$\widehat{U}_K(Q, G) = \frac{1}{K} \sum_{k=1}^{K} \left[\alpha Y^{(k)}(Q, G) - \beta C^{(k)}(Q, G) - \gamma_v |V| - \gamma_e |E|\right], \tag{6}$$

where $Y^{(k)}$ and $C^{(k)}$ denote the outcome and cost of the k-th execution. Throughout the paper, U denotes the theoretical expected utility in Eq. 1, while $\widehat{U}_K$ denotes its empirical estimate. All supervision targets are computed from $\widehat{U}_K$.



(a) Verifier-Grounded Structural Supervision (b) Utility Learning and Task-Adaptive Selection

Figure 1: **Overview of TOPORANKER. (a)** For each training task, TOPORANKER executes diverse candidate DAGs, evaluates their outcomes with a verifier, and constructs graph-level preferences together with node- and edge-removal interventions. **(b)** A task-conditioned graph utility model jointly learns from trial, preference, and intervention data. At test time, it scores candidate graphs before execution and runs only the graph with the highest predicted utility.

3.3 OUTCOME-INDUCED STRUCTURAL SUPERVISION

Fitting the absolute value of $U(Q, G)$ alone does not fully exploit the structure of the graph trials: at deployment time the relevant question is comparative, and even a known utility does not reveal which collaboration units are indispensable. We therefore organize the same utility signal at multiple structural granularities, constructing supervision both across and within graphs.

Within-Task Structural Preference. For $G_i, G_j \in \mathcal{C}_Q$, we define the empirical utility gap

$$\widehat{\Delta}_{ij}(Q) = \widehat{U}_K(Q, G_i) - \widehat{U}_K(Q, G_j), \tag{7}$$

and set $G_i \succ_Q G_j$ whenever $\widehat{\Delta}_{ij}(Q) > 0$. This yields the preference dataset $\mathcal{D}_{\text{pref}} = \{(Q, G^+, G^-) : G^+ \succ_Q G^-\}$, where the preference direction is determined by the sign of $\widehat{\Delta}_{ij}$. Comparisons are restricted to graphs for the same task, preventing task difficulty from being mistaken for structural differences and directly aligning training with candidate ranking at test time.

Structural Intervention. Graph-level preferences compare entire graphs but do not provide local structural credit. We therefore perform controlled, single-step deletions: removing one eligible node v_i with its incident edges yields G^{-v_i}, and blocking one edge e_{ij} yields $G^{-e_{ij}}$. Each intervention removes exactly one structural unit from the original candidate graph; interventions are not accumulated, and no shortcut edges are introduced to reconnect the remaining nodes. Each surviving edit is re-executed under the same protocol. For $a \in V \cup E$, its empirical contribution is

$$\widehat{\Delta}_a(Q, G) = \widehat{U}_K(Q, G) - \widehat{U}_K(Q, G^{-a}). \tag{8}$$

A positive $\widehat{\Delta}_a$ indicates that removing a degrades utility, a near-zero value indicates redundancy, and a negative value indicates unnecessary cost or harmful information. We obtain $\mathcal{D}_{\text{int}} = \{(Q, G, G^{-a}, \widehat{\Delta}_a)\}$.

Both signals are derived from the same candidate construction function and the same utility definition, ensuring that graph-level preferences and local interventions are directly comparable.

3.4 TASK-CONDITIONED COMMUNICATION UTILITY MODEL

We implement the utility predictor $\widehat{U}_\phi(Q, G)$ as a lightweight task-conditioned Edge-aware Graph Transformer. The model jointly encodes task semantics, agent roles, and directed topology, and outputs a scalar utility score through task–graph interaction.

Model Architecture. Each node v_i is initialized with the embedding of its role description, and the task Q is encoded by a shared sentence encoder into a task vector $\mathbf{q}$. A stack of L edge-aware

graph attention layers propagates information along directed edges:

$$\mathbf{h}_i^{(\ell+1)} = \mathrm{Attn}\Big(\mathbf{h}_i^{(\ell)}, \{\mathbf{h}_j^{(\ell)} : (v_j, v_i) \in E\}, \mathbf{q}\Big), \tag{9}$$

where attention is computed over incoming neighbors and modulated by the task vector. The final node representations are pooled into a graph-level vector $\mathbf{h}_G$, which is concatenated with $\mathbf{q}$ and passed through a two-layer MLP to produce the scalar score:

$$\widehat{U}_\phi(Q, G) = \mathrm{MLP}([\mathbf{h}_G; \mathbf{q}]). \tag{10}$$

Learning Objective. Let $\mathcal{D}_{\mathrm{fit}}$ denote the union of the original candidate graphs and the executed intervention graphs. TOPORANKER learns graph utility from three complementary levels of supervision.

First, utility regression calibrates predicted scores against observed execution outcomes:

$$\mathcal{L}_{\mathrm{util}} = \mathbb{E}_{(Q,G)\sim\mathcal{D}_{\mathrm{fit}}}\left[(\widehat{U}_\phi(Q, G) - \widehat{U}_K(Q, G))^2\right]. \tag{11}$$

Second, a within-task ranking loss directly optimizes the relative ordering of candidate graphs:

$$\mathcal{L}_{\mathrm{rank}} = \mathbb{E}_{(Q,G^+,G^-)\sim\mathcal{D}_{\mathrm{pref}}}\left[-\log\sigma\Big(\widehat{U}_\phi(Q, G^+) - \widehat{U}_\phi(Q, G^-)\Big)\right]. \tag{12}$$

Finally, intervention consistency encourages the predictor to reproduce the utility change induced by node or edge deletion:

$$\mathcal{L}_{\mathrm{int}} = \mathbb{E}_{(Q,G,G^{-a},\widehat{\Delta}_a)\sim\mathcal{D}_{\mathrm{int}}}\left[\widehat{U}_\phi(Q, G) - \widehat{U}_\phi(Q, G^{-a}) - \widehat{\Delta}_a\right]^2. \tag{13}$$

The overall objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{util}} + \mathcal{L}_{\mathrm{rank}} + \mathcal{L}_{\mathrm{int}}. \tag{14}$$

The three terms constrain absolute utility, candidate ordering, and local utility differences, respectively. Utility regression provides numerical calibration, the ranking objective emphasizes within-task structural decisions, and intervention consistency further constrains the relative predictions of neighboring structures. All three losses optimize the same utility model, such that the learned scores can be directly used for structural selection at test time.

3.5 INFERENCE

At test time, given an unseen task $Q^\star$, the candidate mechanism first constructs a feasible set $\mathcal{C}_{Q^\star} \subseteq \mathcal{G}(Q^\star; \mathcal{R})$ using the same construction method as in training. The utility model performs lightweight forward scoring over all candidate graphs and selects

$$\widehat{G} = \arg\max_{G\in\mathcal{C}_{Q^\star}} \widehat{U}_\phi(Q^\star, G). \tag{15}$$

The system then executes only $\widehat{G}$ once to produce the final answer.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate TOPORANKER on eight benchmarks across four categories: mathematical reasoning (GSM8K, SVAMP, AQuA, MultiArith (Cobbe et al., 2021; Patel et al., 2021; Ling et al., 2017; Roy & Roth, 2015)), code generation (HumanEval (Chen et al., 2021)), distributed information integration (HiddenBench (Li et al., 2026c)), and scientific reasoning (OlympiadBench (He et al., 2024) and FrontierScience-Olympiad (Wang et al., 2026)).

Unified Configuration. To isolate the effect of communication topology, all methods share identical task inputs, a fixed role pool of five agents, a minimal task-agnostic prompt template, the same execution LLM (GLM 5.2), and the same verifier; only the graph construction or selection strategy differs. For each benchmark, we split the data into training, validation, and test sets with a ratio of $2:1:7$, and train a separate graph utility model on the training split, select hyperparameters on the validation split, and report results on the test split. The candidate graphs used by TOPORANKER are drawn from the same topology families as the fixed baselines. We compare against a single-agent baseline, fixed topologies (Chain, Star, Tree, Complete Graph, and Random Graph), and learnable or adaptive topology methods, including AgentPrune (Zhang et al., 2025a), AgentDropout (Zhang et al., 2025a), G-Designer (Zhang et al., 2025b), EIB-LEARNER (Shen et al., 2025), ARG-Designer (Li et al., 2026b), DyTopo (Lu et al., 2026), GTD (Jiang et al., 2026), and AGP (Li et al., 2025), which we adapt to the same interface. We set $\alpha = 10$, $\beta = 3.5 \times 10^{-4}$, $\gamma_v = 0.04$, and $\gamma_e = 0.07$. To reduce sampling cost, we set the number of executions to $K = 1$. We measure task performance using Pass@1 or task accuracy, and report average token consumption, average number of nodes, and average number of edges to characterize communication cost.

4.2 MAIN RESULTS

Method	GSM8K	SVAMP	AQuA	Multi-Arith	Human-Eval	Hidden-Bench	Olympiad-Bench	Frontier-Science
Single-Agent Baseline								
Vanilla	63.60	93.50	70.44	<u>99.16</u>	90.12	19.23	27.31	27.50
Multi-Agent Systems with Fixed Topologies								
Chain	85.02	95.00	84.23	98.53	96.89	55.76	34.96	46.25
Tree	86.25	96.12	83.74	98.75	96.12	50.00	35.63	42.50
Complete Graph	85.30	<u>96.25</u>	<u>86.69</u>	98.75	96.89	59.61	36.90	45.00
Random Graph	81.32	94.62	83.74	96.36	**98.44**	50.00	34.59	37.25
Star	85.20	95.11	81.23	98.75	96.53	51.28	37.79	40.00
Learnable or Adaptive Topologies								
AgentPrune	91.10	90.58	80.51	94.65	89.75	60.25	34.13	43.55
AgentDropout	91.70	91.04	80.94	95.60	92.98	62.81	35.29	44.80
G-Designer	93.80	93.10	81.60	96.50	91.33	68.53	37.60	47.20
EIB-LEARNER	95.20	94.70	83.46	96.83	94.15	73.29	38.87	49.15
ARG-Designer	94.37	95.63	85.51	98.93	93.74	<u>75.60</u>	<u>39.41</u>	<u>50.25</u>
DyTopo	<u>95.66</u>	92.67	81.50	**99.44**	94.12	74.91	39.02	49.80
GTD	94.14	91.33	82.43	98.88	92.46	72.83	38.51	48.75
AGP	95.01	92.30	81.20	**99.44**	94.62	73.50	38.90	49.30
TOPORANKER (Ours)	**96.30**	**96.37**	**89.65**	98.75	<u>97.67</u>	**78.84**	**40.50**	**51.25**

Table 1: Main results (%) across eight benchmarks. Bold and underlined values denote the best and second-best results in each column, respectively.

Task Performance. As shown in Table 1, compared with the best-performing fixed multi-agent baseline on each benchmark, TOPORANKER achieves higher accuracy on six of the eight benchmarks, matches the best fixed baseline on MultiArith, and decreases by 0.77 percentage points on HumanEval. The macro-average accuracy improvement across benchmarks is 4.91 percentage points. On the five classical benchmarks—GSM8K, SVAMP, AQuA, MultiArith, and HumanEval—the average improvement is 2.47 percentage points. On HiddenBench, OlympiadBench, and FrontierScience-Olympiad, the gains are 19.23, 2.71, and 5.00 percentage points, respectively. These results support the effectiveness of task-conditioned structural selection on most of the evaluated tasks, with the largest gain observed on HiddenBench.

The results also reveal task-dependent limits to the benefits of communication structure selection. On MultiArith, TOPORANKER achieves 98.75% accuracy, matching the best fixed multi-agent baseline but remaining below the single-agent result of 99.16%. On HumanEval, TOPORANKER

achieves 97.67%, compared with 98.44% for Random Graph. Thus, communication structure selection does not guarantee higher accuracy on every task, and its value should be assessed jointly with communication cost.

Dataset	Best Fixed	Fixed Tokens ↓	Best Learnable / Adaptive	Adaptive Tokens ↓	TopoRanker Tokens ↓
GSM8K	Tree	1,031	DyTopo	612 (↓41%)	**506** (↓51%)
SVAMP	Complete Graph	868	ARG-Designer	541 (↓38%)	**380** (↓56%)
AQuA	Complete Graph	1,601	ARG-Designer	1,007 (↓37%)	**685** (↓57%)
MultiArith	Tree	817	AGP	452 (↓45%)	**379** (↓54%)
HumanEval	Random Graph	3,334	AGP	1,332 (↓60%)	**1,174** (↓65%)
HiddenBench	Complete Graph	7,657	ARG-Designer	4,609 (↓40%)	**4,445** (↓42%)
OlympiadBench	Star	5,095	ARG-Designer	3,048 (↓40%)	**3,167** (↓38%)
FrontierScience	Chain	8,420	ARG-Designer	4,622 (↓45%)	**3,950** (↓53%)

Table 2: Communication efficiency relative to the most accurate fixed multi-agent topology and the strongest learnable or adaptive baseline on each benchmark. For both adaptive baselines and TopoRanker, the percentages in parentheses denote token reductions relative to the corresponding fixed topology, computed as $1 - C_{\text{method}}/C_{\text{fixed}}$.

Communication Efficiency. Table 2 compares TopoRanker with both the most accurate fixed topology and the strongest learnable or adaptive baseline on each benchmark. Relative to the fixed-topology references, TopoRanker reduces token consumption on all eight benchmarks, with reductions ranging from 38% to 65%. Combined with Table 1, it achieves higher accuracy and lower token consumption on six benchmarks, while matching the fixed-topology accuracy on MultiArith with a 54% token reduction. Compared with the strongest adaptive baselines, TopoRanker uses fewer tokens on seven benchmarks and achieves higher accuracy on seven, providing simultaneous accuracy and efficiency gains on six. The two exceptions reflect different trade-offs: on MultiArith, TopoRanker uses 379 tokens compared with AGP's 452, but its accuracy is 0.69 percentage points lower; on OlympiadBench, it improves accuracy over ARG-Designer by 1.09 percentage points while using 3,167 rather than 3,048 tokens. On HumanEval, relative to the fixed baseline, a 0.77-percentage-point accuracy decrease is accompanied by a 65% token reduction. Overall, TopoRanker provides a favorable performance–communication trade-off across diverse tasks.

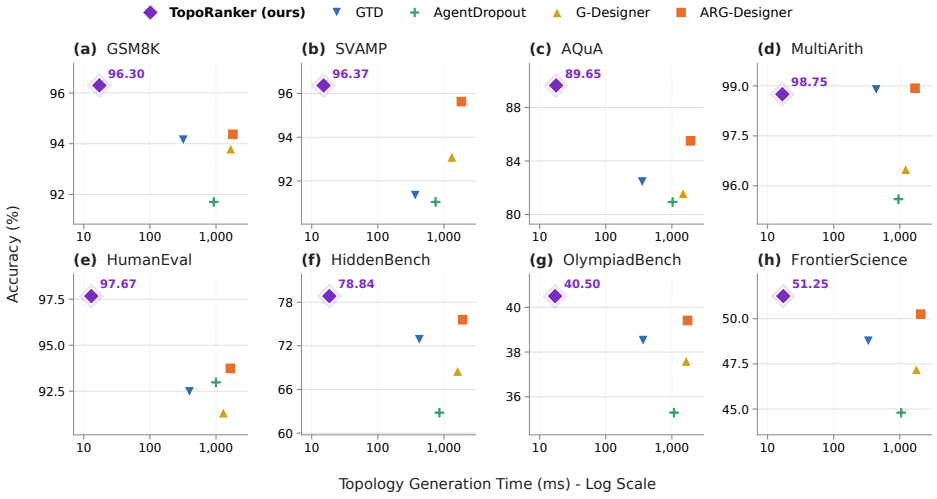


Figure 2: Accuracy–latency trade-offs across eight benchmarks. Each panel compares end-task performance against the latency required to construct or select a communication topology before multi-agent execution. The horizontal axis is logarithmic, and the upper-left region indicates a better performance–efficiency trade-off.

Topology Generation Efficiency. Figure 2 compares topology decision latency and end-task performance across eight benchmarks. Across these benchmarks, TopoRanker achieves a $24.2\times$–$125.0\times$ speedup compared to the baselines. These results indicate that lightweight pre-execution ranking provides a favorable and consistent performance–efficiency trade-off across tasks.

Dataset	Candidate-Pool Performance			Selector Quality		
Benchmark	Oracle Pass@1 ↑	Offline Pass@1 ↑	Oracle Coverage ↑	Top-1 Match ↑ (Random)	Pairwise Accuracy ↑	Normalized Regret ↓
GSM8K	98.10%	96.30%	98.17%	80.56% (66.67%)	55.88%	0.88%
SVAMP	97.83%	96.37%	98.51%	73.33% (65.33%)	63.24%	0.67%
AQuA	92.76%	89.65%	96.65%	37.50% (42.50%)	47.37%	4.21%
MultiArith	98.89%	98.75%	99.86%	50.00% (30.00%)	50.00%	0.28%
HumanEval	98.97%	97.67%	98.69%	76.92% (56.92%)	57.14%	3.09%
HiddenBench	89.74%	78.84%	87.85%	56.25% (50.00%)	54.49%	35.90%
OlympiadBench	85.34%	40.50%	47.46%	68.41% (30.31%)	71.24%	21.31%
FrontierScience-Olympiad	81.67%	51.25%	62.75%	76.67% (20.08%)	65.71%	12.40%

Table 3: **Candidate-pool coverage and offline selector quality.** Oracle Pass@1 counts a task as solved if any candidate succeeds, while Offline Pass@1 evaluates the selected candidate. Coverage is the fraction of Oracle-solvable tasks solved by the selection. Top-1 Match checks whether the selected graph attains maximum observed utility, with the tie-aware random expectation in parentheses. Pairwise Accuracy evaluates candidate pairs with unequal observed utilities. Normalized Regret averages $(u_{\max} - u_{\text{sel}})/(u_{\max} - u_{\min})$ across tasks, with zero assigned when all candidates have equal utility.

4.3 Ablation Studies

Candidate-Pool Upper Bound and Selector Quality. Table 3 separates candidate availability from selector quality. Oracle Pass@1, Offline Pass@1, and Coverage characterize the task-level upper bound and its recovery, while Top-1 Match, Pairwise Accuracy, and Normalized Regret assess exact selection, relative ordering, and the utility cost of ranking errors, respectively. This separation distinguishes limitations of the candidate pool from failures of the learned selector.

On the five classical benchmarks, the Oracle–Offline gap is at most 3.11 percentage points, Oracle Coverage exceeds 96%, and Normalized Regret does not exceed 4.21%, indicating that both candidate availability and selection are near saturation. The harder benchmarks expose a different regime. On HiddenBench, Oracle Pass@1 reaches 89.74% but Offline Pass@1 is 78.84%, a gap of 10.9 points, while Normalized Regret is 35.90%—the highest among all benchmarks—indicating that the selector's top-ranked graph is often far from optimal in utility terms even when the final task outcome remains close. On OlympiadBench and FrontierScience-Olympiad, the gaps are substantially larger: Oracle/Offline Pass@1 is $85.34/40.50\%$ and $81.67/51.25\%$, respectively, with Normalized Regret of 21.31% and 12.40%. These results locate the primary bottleneck in ranking rather than graph availability for the two hardest benchmarks, while HiddenBench shows that a large utility regret does not necessarily translate into a large task-level gap.

Crucially, ranking fidelity and decision quality need not coincide. On AQuA, Top-1 Match falls below its random expectation (37.50% vs. 42.50%) and Pairwise Accuracy is only 47.37%, yet Normalized Regret is only 4.21%, while the Oracle–Offline Pass@1 gap is 3.11 percentage points. This suggests that many ranking errors exchange near-equivalent graphs and therefore have little effect on the final decision. More broadly, topology selectors should be evaluated jointly through candidate coverage and utility-sensitive regret: once the candidate pool saturates, further graph generation offers diminishing returns, and progress depends primarily on reliably discriminating among the most consequential candidates.

Structural Supervision Ablation. Table 4 reveals a consistent benefit from combining global preferences with local structural interventions. Averaged across the four benchmarks, full TopoRanker achieves 78.33% Pass@1, compared with 75.11% without intervention supervision, 75.12% without preference supervision, and 71.85% with utility regression alone. Removing ei-

ther structural signal therefore produces a nearly identical macro-average drop of approximately 3.2 percentage points, whereas removing both enlarges the drop to 6.48 points. This near-additive pattern provides stronger evidence that graph-level ranking and local credit assignment capture complementary information rather than acting as interchangeable objectives.

Dataset	Variant	$\mathcal{L}_{\text{util}}$	$\mathcal{L}_{\text{rank}}$	$\mathcal{L}_{\text{int}}$	Pass@1 ↑	Avg. nodes	Avg. edges	Avg. Tokens ↓
GSM8K	Utility Only	✓	×	×	89.32	3.2	2.5	450
	w/o Intervention	✓	✓	×	93.70	2.8	3.1	403
	w/o Preference	✓	×	✓	92.67	3.2	2.6	458
	Full TopoRanker	✓	✓	✓	**96.30**	3.1	3.0	506
HumanEval	Utility Only	✓	×	×	90.50	4.1	3.5	1,045
	w/o Intervention	✓	✓	×	94.92	3.6	4.0	928
	w/o Preference	✓	×	✓	95.28	4.1	3.6	1,056
	Full TopoRanker	✓	✓	✓	**97.67**	3.9	3.8	1,174
HiddenBench	Utility Only	✓	×	×	73.07	4.5	4.1	3,957
	w/o Intervention	✓	✓	×	75.00	4.0	4.9	3,514
	w/o Preference	✓	×	✓	75.00	4.5	4.2	4,001
	Full TopoRanker	✓	✓	✓	**78.84**	4.4	4.9	4,445
OlympiadBench	Utility Only	✓	×	×	34.52	5.1	4.6	2,818
	w/o Intervention	✓	✓	×	36.81	4.5	5.4	2,504
	w/o Preference	✓	×	✓	37.52	5.1	4.7	2,849
	Full TopoRanker	✓	✓	✓	**40.50**	4.9	5.2	3,167

Table 4: Ablation of graph-level preference supervision and local structural intervention across four representative benchmarks. For each variant, we report Pass@1, the average numbers of selected nodes and communication edges, and average token consumption.

Their relative importance nevertheless varies across tasks. Preference supervision contributes more on GSM8K, where its removal decreases Pass@1 by 3.63 points, compared with a 2.60-point decrease after removing interventions. The opposite pattern emerges on OlympiadBench: intervention supervision contributes 3.69 points, while preference supervision contributes 2.98 points. On HiddenBench, removing either signal produces the same 3.84-point degradation. These differences suggest that global candidate discrimination is particularly useful for selecting among alternative solution structures, whereas local structural credit becomes increasingly important when success depends on identifying consequential roles and communication links.

The improvements cannot be explained by simply selecting larger graphs. Relative to utility-only learning, the full model selects slightly fewer nodes on every benchmark but consistently retains more communication edges, while improving Pass@1 by 5.77–7.17 points. Full TopoRanker also consumes more tokens than the partial variants, indicating that the supervision signals favor more effective information exchange rather than uniform sparsification. Nevertheless, its token consumption remains substantially below the strongest fixed-topology references reported in Table 2. Overall, the ablation results show that global preferences determine which collaboration patterns are worth executing, while local interventions refine where communication is useful; their combination yields the strongest and most consistent performance.

5 Conclusion

We introduce TopoRanker, a verifier-driven framework that learns task-conditioned topology utility from within-task graph preferences and node and edge interventions. Across eight benchmarks, TopoRanker achieves a favorable accuracy–communication trade-off while executing only the selected graph at test time; selector analyses and ablations further confirm the complementary value of global comparison and local structural feedback. These results highlight utility relations across candidate graphs and their local variations as effective supervision for communication topology selection.

REFERENCES

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, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11733–11763. PMLR / OpenReview.net, 2024.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211.

Eric Hanchen Jiang, Levina Li, Frank Wan, Xiao Liang, Sophia Yin, Yuchen Wu, Xinfeng Li, Yizhou Sun, Wei Wang, Kai-Wei Chang, and Ying Nian Wu. Dynamic generation of multi LLM agents communication topologies with graph diffusion models. In *ACL (1)*, pp. 38042–38060. Association for Computational Linguistics, 2026.

Hui Yi Leong, Yuheng Li, Yuqing Wu, Wenwen Ouyang, Wei Zhu, Jiechao Gao, and Wei Han. AMAS: adaptively determining communication topology for llm-based multi-agent system. In *EMNLP (Industry Track)*, pp. 2061–2070. Association for Computational Linguistics, 2025.

Boyi Li, Zhonghan Zhao, Der-Horng Lee, and Gaoang Wang. Adaptive graph pruning for multi-agent communication. In *ECAI*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, pp. 4305–4312. IOS Press, 2025.

Junzhi Li, Peng He, Qirui Ji, Wei Wang, Lixiang Liu, and Chuxiong Sun. Discovering efficient and explainable communication topologies for llm-based multi-agent systems via causal inference. 2026a.

Shiyuan Li, Yixin Liu, Qingsong Wen, Chengqi Zhang, and Shirui Pan. Assemble your crew: Automatic multi-agent communication topology design via autoregressive graph generation. In *AAAI*, pp. 23142–23150. AAAI Press, 2026b.

Yuxuan Li, Aoi Naito, and Hirokazu Shirado. Systematic failures in collective reasoning under distributed information in multi-agent llms, 2026c.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *ACL (1)*, pp. 158–167. Association for Computational Linguistics, 2017.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization. *CoRR*, abs/2310.02170, 2023.

Yuxing Lu, Yucheng Hu, Xukai Zhao, and Jiuxin Cao. Dytopo: Dynamic topology routing for multi-agent reasoning via semantic matching. *CoRR*, abs/2602.06039, 2026.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *NAACL-HLT*, pp. 2080–2094. Association for Computational Linguistics, 2021.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *EMNLP*, pp. 1743–1752. The Association for Computational Linguistics, 2015.

Xu Shen, Yixin Liu, Yiwei Dai, Yili Wang, Rui Miao, Yue Tan, Shirui Pan, and Xin Wang. Understanding the information propagation effects of communication topologies in llm-based multi-agent systems. In *EMNLP*, pp. 12347–12361. Association for Computational Linguistics, 2025.

Miles Wang, Robi Lin, Kat Hu, Joy Jiao, Neil Chowdhury, Ethan Chang, and Tejal Patwardhan. Frontierscience: Evaluating ai's ability to perform expert-level scientific tasks, 2026.

Zhexuan Wang, Yutong Wang, Xuebo Liu, Liang Ding, Miao Zhang, Jie Liu, and Min Zhang. Agentdropout: Dynamic agent elimination for token-efficient and high-performance llm-based multi-agent collaboration. In *ACL (1)*, pp. 24013–24035. Association for Computational Linguistics, 2025.

Jinxi Yu, Yubei Li, Eric Hanchen Jiang, Zhi Zhang, Dong Liu, Wenxiao Zhao, Levina Li, Kai-Wei Chang, and Ying Nian Wu. Codebook agent: Amortized topology design for llm multi-agent systems. 2026.

Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. In *ICLR*. OpenReview.net, 2025a.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. G-designer: Architecting multi-agent communication topologies via graph neural networks. In *ICML*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025b.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models. *CoRR*, abs/2304.09797, 2023.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Gptswarm: Language agents as optimizable graphs. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62743–62767. PMLR / OpenReview.net, 2024.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, title brainstorming, figure prototyping, and feedback on manuscript organization. All technical claims, experimental results, mathematical formulations, and final manuscript content were reviewed and verified by the authors. The authors take full responsibility for the content of this submission.