RESPONSE-CONDITIONED PARALLEL-TO-SEQUENTIAL ORCHESTRATION FOR MULTI-AGENT SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-agent systems can solve complex tasks through collaboration between multiple Large Language Model agents. Existing collaboration frameworks typically operate in either a parallel or a sequential mode. In the parallel mode, agents respond independently to queries followed by aggregation of responses. In contrast, sequential systems allow agents to communicate via a directed topology and refine one another step by step. However, both modes are inadequate for achieving the desired objectives of minimizing communication and latency while simultaneously maximizing the accuracy of the final response. In this work, we introduce a hybrid paradigm called NEXA, a trainable response-conditioned policy that bridges the gap between the two modes. NEXA begins with a parallel execution stage, embeds the resulting responses into a shared semantic space, and then predicts a sparse directed acyclic communication graph. If the graph is empty, the system remains purely parallel; if it is non-empty, the system performs one sequential message propagation. The policy is a lightweight transformer model, and the method avoids the need for external LLM judges or reward models, as well as hand-crafted test-time topology search. We formalize this hybrid execution problem, show that the resulting graph is acyclic by construction, and present a training procedure based on policy-gradient optimization. Results demonstrate that NEXA attains accuracy comparable to the strongest baseline while using 35.4% fewer tokens; the frozen policy also transfers across agent counts, tasks, and model scales, including 14B and 32B evaluations on GSM-Hard and AIME-2024.

1 INTRODUCTION

Large language models (LLMs) have become increasingly capable at reasoning, coding, planning, and dialogue, yet a single model still suffers from stochastic failures, brittle long-horizon reasoning, and occasional hallucinations. Multi-agent systems aim to address these weaknesses by distributing problem solving across multiple agents whose outputs can complement, critique, or refine one another. The central question of such systems is **how that collaboration should be orchestrated.**

Existing LLM-based multi-agent systems largely fall into two categories. In **parallel** systems, agents answer independently, and their outputs are combined by majority voting, self-consistency, or a learned aggregation rule (Wang et al., 2023; Jiang et al., 2023). In **sequential** systems, agents are arranged in a communication topology, often a chain, tree, or a more general graph, and information is propagated step by step (Zhuge et al., 2024; Qian et al., 2025). Parallel systems are simple and scalable, but they are computationally expensive, token-intensive, and often redundant, requiring multiple rounds of parallel message propagation while still being unable to exploit targeted communication when one draft could help repair another. Sequential systems can support error correction and information flow, but they require a topology and therefore inherit the burden of deciding who should communicate with whom. Prior work has explored fixed topologies, policy-gradient optimization over edges, graph generators conditioned on tasks or roles, and judge-based routing, each adding substantial token, compute, optimization, coordination overhead, or reducing transferability across settings (Qian et al., 2025; Zhuge et al., 2024; Zhang et al., 2025b).

These two paradigms are often treated as separate design choices. A system is either built as a parallel ensemble or as a sequential graph-based collaboration mechanism. Yet, this distinction is too rigid. In many realistic settings, the right approach is not to commit in advance to a single paradigm, but to start in parallel and then decide, based on the agent's actual outputs, whether sequential propagation

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.

is necessary. If the initial responses already contain strong agreement and sufficient information, additional communication may be unnecessary. If they disagree in informative ways, or if useful signals are scattered across agents, then structured propagation may help. This suggests that the real problem is not "parallel or sequential" in the abstract, but rather:

> Given the current pool of agent responses, should the system remain in the parallel regime, or should it instantiate a communication graph and perform sequential refinement?

To answer this question, we introduce **NEXA** (from "nexus", a connection or link), a trainable policy for *communication graph prediction* in multi-agent LLM systems. NEXA begins with a parallel draft stage in which all agents answer independently. The resulting response pool is embedded into a shared semantic space, producing a compact representation of the current response state of the team. A lightweight transformer-based policy then predicts a sparse directed acyclic graph (DAG). If the graph is empty, the system remains in the parallel regime and returns the parallel aggregate. If the graph is non-empty, the system executes one sequential consolidation pass in which selected agents update their responses using information from upstream nodes.

This formulation deliberately treats parallel and sequential execution not as mutually exclusive system designs, but as two outcomes of the same learned policy. In this sense, the central contribution of NEXA is not merely graph prediction. It is a mechanism for *bridging the gap between parallel execution and sequential execution* by using parallel drafts to decide whether structured propagation is needed and, if so, how it should proceed.

A second principle of the method is simplicity. We do *not* learn the topological order. Instead, we induce the order from agent contributions, retaining the most stable organizing principle of response-conditioned communication. The policy learns only the communication edges. We score the candidate communication edges using the affinity matrix formed from transformer-contextualized response representations. This makes the policy lightweight and keeps the graph decoder tightly coupled to the semantic interactions encoded by the backbone.

NEXA is also designed to be agnostic to superficial configuration details. The policy consumes semantic representations of agent outputs rather than role labels, agent identities, or model-family indicators. As a consequence, the planner is structurally insensitive to which agent is called "Programmer" or "Assistant"; what matters is what the agents actually say. This does *not* by itself guarantee transfer across all tasks or backbones, and we explicitly treat that as an empirical question. But it does mean that the policy class is not intrinsically tied to a fixed role inventory or a single team structure.

The paper makes four contributions. First, it formalizes a hybrid decision problem in which a learned communication graph determines whether a multi-agent system remains parallel or enters a sequential propagation regime. Second, it proposes a contribution-ordered, attention-based graph policy that predicts only communication edges, keeping the controller simple and acyclic by construction. Third, it establishes DAG validity, hybrid subsumption, and permutation-based identity agnosticism. Fourth, it evaluates NEXA across reasoning and programming tasks, with same-draft controls, latency measurements, larger-model tests, and transfer across agent counts, tasks, model scales, and generations.

2 PROBLEM FORMULATION AND PRELIMINARIES

Let $\mathcal{A} \in \{\mathcal{A}_1, \ldots, \mathcal{A}_N\}$ be a set of N agents, and let $\mathcal{Q}$ be a user query. Each agent may differ in prompt, role, or backbone model, but the communication policy introduced in this work does not rely on these identities explicitly. Instead, it operates on the semantic content of the agents' responses.

Given the query, each agent independently produces an initial response

$$\mathcal{R}_n^{(0)} = \mathcal{A}_n(\mathcal{Q}), \qquad n \in \{1, 2, \ldots, N\}. \tag{1}$$

The first phase is fully parallel and produces a draft response set $\mathcal{R}^{(0)} = \{\mathcal{R}_1^{(0)}, \ldots, \mathcal{R}_N^{(0)}\}$.

The purpose of this is twofold. First, it provides diverse candidate solutions to the query. Second, and more importantly for our setting, it exposes the **current response state** of the multi-agent system. Since LLM outputs are inherently stochastic, this realized state is more informative for downstream coordination than static task labels or role descriptions. This response-conditioned perspective is central to the present work and follows the same foundational motivation that underlies SelfOrg (Tastan et al., 2026).

To reason about relations among agent outputs, we map each response into a shared semantic embedding space using a fixed lightweight encoder f (all-MiniLM-L6-v2 (Reimers & Gurevych, 2019)): $r_n = f(\mathcal{R}_n^{(0)}) \in \mathbb{R}^d$.

Following SelfOrg (Tastan et al., 2026), we define the average response embedding $r_{\text{avg}} = \frac{1}{N}\sum_{n=1}^{N} r_n$ and contribution scores $\psi_n = \cos(r_n, r_{\text{avg}})$. SelfOrg motivates ψ_n as a linear-time approximation to a Shapley-style contribution value (Shapley, 1953) and shows that, under suitable separation conditions, ranking by ψ_n preserves the normalized Shapley ordering. This is precisely why NEXA uses contribution to define the topological ordering of the edges.

The orchestration problem is to predict a directed communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \pi)$, where $V = \{1, \ldots, N\}, E \subseteq V \times V$, and π is an order over the nodes. If the graph is empty ($\mathcal{E} = \varnothing$), the system stays in the parallel regime and outputs an aggregate of the initial drafts. If it is non-empty ($\mathcal{E} \neq \varnothing$), the graph induces a sequential propagation step.

For each node n, define its parent set $\mathrm{Pa}(n) = \{m : (m \to n) \in \mathcal{E}\}$. Then the updated response is

$$\mathcal{R}_n^{(1)} = \begin{cases} \mathcal{A}_n(\mathcal{Q}, \{\mathcal{R}_m^{(\star)} : m \in \mathrm{Pa}(n)\}), & \mathrm{Pa}(n) \neq \varnothing, \\ \mathcal{R}_n^{(0)}, & \mathrm{Pa}(n) = \varnothing, \end{cases} \tag{2}$$

where $\mathcal{R}_m^{(\star)}$ denotes the most recent available parent response under the topological execution order.

The final answer is selected from the resulting response pool using a judge-free aggregation rule. Let $\{\boldsymbol{z}_n\}$ be the final response embeddings and $\{w_n\}$ their contribution weights. We compute

$$z_{\text{centroid}} = \frac{\sum_{n=1}^{N} w_n \boldsymbol{z}_n}{\sum_{n=1}^{N} w_n}, \qquad n^\star = \arg\max_n \cos(\boldsymbol{z}_n, \boldsymbol{z}_{\text{centroid}}). \tag{3}$$

and return the corresponding response.

The learning objective is to maximize final task correctness. Given ground truth y and final prediction $\hat{y}_{\mathcal{G}}$ under graph $\mathcal{G}$, the reward is

$$R(\mathcal{G}) = \mathbb{1}\left[\mathrm{Eval}(\hat{y}_{\mathcal{G}}, y) = 1\right]. \tag{4}$$

The policy, therefore, learns to predict a communication graph that determines whether the initial parallel responses should remain as they are or be further refined through structured propagation.

3 METHODOLOGY

3.1 SYSTEM OVERVIEW

NEXA consists of five stages. First, all agents produce draft responses in parallel. Second, those responses are embedded into a shared semantic space. Third, a response-conditioned transformer policy predicts a sparse communication graph. Fourth, if the graph is non-empty, the corresponding destination nodes are updated sequentially. Fifth, the final answer is selected from the resulting response pool by weighted-centroid-based aggregation.

This design has a central conceptual consequence: parallel execution is not discarded when sequential communication is introduced. Instead, the parallel draft stage becomes the *source of evidence* that determines whether the system should remain in the parallel regime or transition into a sequential propagation regime.

3.2 CONTRIBUTION-DEFINED ORDER AND DAG VALIDITY

We set the topological order as $\pi = \mathrm{argsort}(\psi_1, \ldots, \psi_N; \psi_k \geq \psi_{k+1}, \forall k \in [N])$.

Higher-contribution agents are placed earlier because, when errors are dispersed, agreement tends to concentrate on the correct answer and the consensus-aligned cluster dominates the contribution score (Tastan et al., 2026). This is a distributional prior rather than a verifier: a rare, correct specialist that disagrees with the majority can still be placed downstream. Section E therefore reports descending, ascending, and randomized-order controls. The feasible edge set is restricted to

$$\mathcal{E}_\pi = \{(m, n) : \pi^{-1}(m) < \pi^{-1}(n)\}, \tag{5}$$

so that communication is only allowed to move forward under the contribution order.

Proposition 1 (Acyclicity by construction). *For any edge set $\mathcal{E} \subseteq \mathcal{E}_\pi$, the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \pi)$ is a directed acyclic graph.*

Proof. Assume for contradiction that $\mathcal{G}$ contains a directed cycle

$$v_1 \to v_2 \to \cdots \to v_K \to v_1. \tag{6}$$

Because every edge must go forward under π, we must simultaneously have

$$\pi^{-1}(v_1) < \pi^{-1}(v_2) < \cdots < \pi^{-1}(v_K) < \pi^{-1}(v_1), \tag{7}$$

which is impossible. Hence, no directed cycle can exist. □

This parameterization is simpler than detecting and repairing cycles after graph prediction because DAG validity is built directly into the action space of the policy.

3.3 Response-Conditioned Graph Policy

The graph policy consumes only the current response set, not agent identities, role labels, or model-family indicators. Let

$$\mathcal{X} = [r_1, \ldots, r_N]^\top \in \mathbb{R}^{N \times d}. \tag{8}$$

A transformer encoder (Vaswani et al., 2017) Enc_θ maps the response set to contextualized node states

$$\mathcal{H} = \mathrm{Enc}_\theta(X) = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N]^\top, \qquad \boldsymbol{h}_n \in \mathbb{R}^{d_h}. \tag{9}$$

Because the encoder operates on the response embeddings without identity-specific tokens, the policy is permutation-equivariant over the agent dimension.

Proposition 2 (Permutation-based identity agnosticism). *Assume that the response encoder f is applied independently to each response and that Enc_θ is permutation-equivariant. Then, for any permutation matrix P,*

$$\mathrm{Enc}_\theta(P\mathcal{X}) = P\,\mathrm{Enc}_\theta(\mathcal{X}), \tag{10}$$

the induced graph distribution is equivariant to any relabeling of agents.

Proof. Since Enc_θ (transformer) without positional encodings is permutation-equivariant, permuting agent indices permutes $\mathcal{X}$ and thus $\mathrm{Enc}_\theta(\mathcal{X})$; the remaining steps (cosine-to-mean scoring, ordering, and edge construction) are permutation-consistent, so the graph distribution is equivariant. □

We then predict communication edges directly from the globally contextualized hidden states. Concretely, we form a response-response score matrix from the contextualized states:

$$\Lambda = \mathcal{H}\mathcal{H}^\top. \tag{11}$$

Here, Λ is not the final adjacency matrix; it provides edge logits that are passed through a sigmoid and then sampled to obtain the communication graph. This construction is deliberate. The hidden states in $\mathcal{H}$ are already globally contextualized, so the resulting edge logits are informed by the entire response set rather than by isolated response pairs. In this way, the communication graph is read directly from the shared semantic structure induced by the encoder.

3.4 Response Propagation and Aggregation

NEXA does not require a separate stop network or node-activation network. The graph itself determines both whether sequential communication occurs and which nodes are updated. A node is updated if and only if it has at least one incoming edge:

$$u_n = \mathbf{1}\left[\sum_{m=1}^{N} \mathbf{1}[(m \to n) \in \mathcal{E}] > 0\right]. \tag{12}$$

If $\mathcal{E} = \varnothing$, then $u_n = 0$ for all nodes, no additional calls are made, and the system returns the parallel aggregate. If $\mathcal{E} \neq \varnothing$, the graph induces one sequential consolidation pass.

Proposition 3 (Hybrid subsumption). *The policy class of* NEXA *subsumes the pure parallel regime.*

Proof. The empty graph $\mathcal{E} = \varnothing$ is always attainable (all edge probabilities zero/small), in which case no updates occur, and the method reduces to pure parallel execution with aggregation. Any $\mathcal{E} \neq \varnothing$ induces at least one sequential update, so the policy class strictly contains the parallel regime. □

When $\mathcal{E} \neq \varnothing$, sequential propagation follows the contribution order π. For node n, define

$$\mathrm{Pa}(n) = \{m : (m \to n) \in \mathcal{E}\}. \tag{13}$$

The updated response is then

$$\mathcal{R}_n^{(1)} = \begin{cases} \mathcal{A}_n\Big(\mathcal{Q}, \{\mathcal{R}_m^{(\star)} : m \in \mathrm{Pa}(n)\}\Big), & \mathrm{Pa}(n) \neq \varnothing, \\ \mathcal{R}_n^{(0)}, & \mathrm{Pa}(n) = \varnothing, \end{cases} \tag{14}$$

where $\mathcal{R}_m^{(\star)}$ denotes the most recent available parent response under the topological execution order. Because all edges go forward under π, each parent response is available when a destination node is updated.

After either staying in the parallel regime or completing one propagation pass, NEXA selects the final answer without using an external judge. Let $\tilde{\mathcal{R}}_n$ denote the final candidate response for agent n and let

$$z_n = f(\tilde{\mathcal{R}}_n), \qquad z_{\text{avg}} = \frac{1}{N}\sum_{n=1}^{N} z_n, \qquad w_n = \cos(z_n, z_{\text{avg}}). \tag{15}$$

We then compute the contribution-weighted centroid $z_{\text{centroid}} = \frac{\sum_{n=1}^{N} w_n z_n}{\sum_{n=1}^{N} w_n}$ and select

$$n^\star = \arg\max_n \cos(z_n, z_{\text{centroid}}), \qquad \hat{y} = \tilde{\mathcal{R}}_{n^\star}. \tag{16}$$

This aggregation rule directly inherits the response-conditioned, judge-free philosophy of Self-Org (Tastan et al., 2026).

3.5 TRAINING OBJECTIVE

The deployment objective is the final task correctness. For a labeled example $(\mathcal{Q}, y)$, let $\hat{y}_{\mathcal{G}}$ denote the final output under graph $\mathcal{G}$. In the current implementation, correctness is checked with the same verifier used in evaluation, instantiated as an `xVerify`-based binary reward (Chen et al., 2025). We therefore define the task reward

$$R_{\text{task}}(\mathcal{G}) = \mathbb{1}\left[\mathrm{Eval}(\hat{y}_{\mathcal{G}}, y) = 1\right]. \tag{17}$$

Because the order π is fixed by the contribution scores, the graph log-probability decomposes over feasible forward edges:

$$\log p_\theta(\mathcal{E} \mid \mathcal{X}, \pi) = \sum_{(m,n)\in\mathcal{E}_\pi} \Big(e_{m\to n} \log p_{m\to n} + (1 - e_{m\to n}) \log(1 - p_{m\to n})\Big). \tag{18}$$

The algorithm also applies an explicit sparsity penalty to the sampled graph reward in the same spirit as topology-economical methods (Zhang et al., 2025a). Let $M = |\mathcal{E}_\pi| = \frac{N(N-1)}{2}$ be the number of feasible forward edges under the contribution-defined order. For a sampled graph $\mathcal{G}$, we define the sparsity-regularized reward

$$R_{\text{sp}}(\mathcal{G}) = R_{\text{task}}(\mathcal{G}) - \lambda_{\text{sp}} \frac{|\mathcal{E}|}{M}, \tag{19}$$

where $\lambda_{\text{sp}} \geq 0$ controls how strongly dense communication graphs are penalized.

We train NEXA with REINFORCE and a batch-mean baseline. For a mini-batch of sampled graphs $\{\mathcal{G}^{(i)}\}_{i=1}^{B}$, we set

$$b = \frac{1}{B}\sum_{i=1}^{B} R_{\text{sp}}(\mathcal{G}^{(i)}), \qquad A^{(i)} = R_{\text{sp}}(\mathcal{G}^{(i)}) - b. \tag{20}$$

The policy-gradient term is

$$\mathcal{L} = -\frac{1}{B}\sum_{i=1}^{B} A^{(i)} \log p_\theta(\mathcal{E}^{(i)} \mid \mathcal{X}^{(i)}, \pi^{(i)}). \tag{21}$$

Finally, we obtain the final optimization goal: it is REINFORCE with batch-mean advantage, while sparsity is enforced through the edge-count penalty in the reward.

The full procedure is summarized in Algorithm 1.

4 EXPERIMENTS

The empirical study is designed to demonstrate the generalizability of the learned communication policy. Rather than only reporting in-domain performance for the training configuration, we evaluate whether a response-conditioned policy learned in one setting can be reused when the number of agents, the task, or the underlying agent changes, thus emphasizing training efficiency.

4.1 EXPERIMENTAL SETUP

The base training setting uses Qwen2.5-1.5B-Instruct agents (Qwen et al., 2025) on AQUA-RAT (Ling et al., 2017) and GSM8K (Cobbe et al., 2021) with $N = 10$ agents. For each task, the official training pool is deterministically split into disjoint policy-optimization and checkpoint-selection subsets, with no test overlap. HumanEval is evaluation-only: the GSM8K-trained policy is loaded unchanged, and no HumanEval example is used for training or selection. Unless otherwise specified, the policy is trained with REINFORCE, a batch-mean baseline, batch size 32, 50 policy updates, learning rate 0.1, dropout 0.3, and edge-count sparsity coefficient $\lambda_{\text{sp}} = 0.1$. The policy is a one-layer, one-head bidirectional Transformer without positional encodings, followed by the $\mathcal{H}\mathcal{H}^\top$ edge construction in Section 3. Additional implementation details appear in Appendix D.

We consider single-agent system, chain-of-thought (CoT) (Wei et al., 2022), self-consistency (Wang et al., 2023), SelfOrg*[1] (Tastan et al., 2026), and topology-learning or pruning methods, including GPTSwarm (Zhuge et al., 2024), AgentPrune (Zhang et al., 2025a), and G-Designer (Zhang et al., 2025b) as baselines. While the primary metric is accuracy of the final response, we also report the mean edge count and token consumption usage as proxies for the communication burden and inference cost, respectively.

4.2 MAIN RESULTS

Table 1 reports the main comparison across AQUA-RAT (Ling et al., 2017), HumanEval (Chen et al., 2021), and GSM8K (Cobbe et al., 2021). NEXA reaches 60.90% average accuracy versus 60.36% for SelfOrg*; this 0.54-point difference is not statistically significant ($z = 0.63$, $p = 0.53$). The supported conclusion is therefore comparable accuracy at substantially lower communication cost, rather than raw-accuracy superiority over SelfOrg*. The gain over the single-agent baseline is significant ($z = 2.82$, $p = 0.005$); complete tests are in Appendix E.

Table 1: Main comparison across AQUA-RAT, HumanEval, and GSM8K. Per-task entries are mean ± std.; average uncertainty is recomputed across each run's three-task average. Token usage includes prompt and completion tokens. Lower rating and token usage are better.

Method	AQUA-RAT	HumanEval	GSM8K	Avg. Acc.	Avg. Rating	Token Usage
Single	52.62±0.99	50.41±3.73	70.07±1.30	57.70±1.84	5.67	1,149,321
CoT	54.46±1.94	45.93±1.27	70.47±1.03	56.95±0.68	5.33	1,298,471
SC	56.82±2.27	12.52±3.41	71.53±0.90	46.96±1.63	5.00	11,215,153
SelfOrg*	56.46±1.28	**52.03±2.44**	72.60±0.60	60.36±1.32	2.67	28,405,941
GPTSwarm	55.91±1.97	36.79±1.96	69.33±2.61	54.01±0.73	6.33	38,021,213
AgentPrune	57.58±2.84	29.47±0.93	71.07±1.94	52.71±0.93	4.00	31,444,709
GDesigner	57.13±1.74	28.25±1.54	70.00±1.06	51.79±0.74	5.67	31,798,540
NEXA	**57.74±2.31**	51.42±0.70	**73.53±0.23**	**60.90±0.68**	**1.33**	18,363,825

End-to-end latency. Latency is measured over three 100-query runs, with concurrent draft generation and graph-level batching for refinements. Table 2 reports this experiment independently of the accuracy and transfer studies. NEXA is about 2.5× faster than fixed one-round SelfOrg* while retaining response-conditioned communication; pure parallel is faster because it performs no refinement calls. Full scaling measurements appear in Appendix E.

Table 2: End-to-end GSM8K latency at $N = 5$ (seconds/example).

Method	Latency (s)
Pure parallel	2.68 ± 0.05
SelfOrg*	19.89 ± 0.20
NEXA	7.89 ± 0.07

[1] SelfOrg* indicates SelfOrg with a single sequential communication round.

4.3 Generalizability Across Different Axes

Number of agents. We first examine generalizability across the number of agents. NEXA is trained with $N = 10$ agents and evaluated without retraining for $N \in \{5, 10, 15, 20\}$. To separate communication from ordinary ensemble scaling, the pure-parallel control reuses the exact same drafts at each N. Communication changes accuracy by $+1.50/+0.08/+2.13/+1.97$ points on AQUA-RAT and $+1.60/+0.80/+1.00/+1.33$ on GSM8K for $N = 5/10/15/20$, respectively. Thus the gains cannot be attributed only to enlarging the initial draft pool.

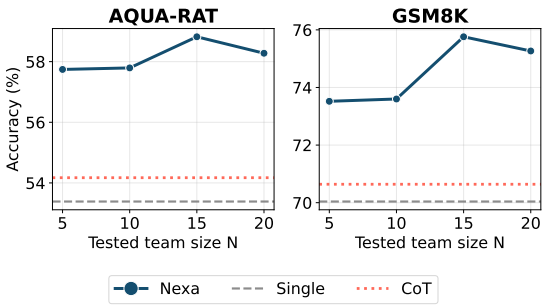

Figure 1: Agent-count transfer for NEXA. The policy is trained with N=10 Qwen2.5-1.5B agents and evaluated without retraining at $N \in \{5, \ldots, 20\}$.

Task transfer. We next consider generalizability across tasks while keeping the model family, model size, and training team size fixed. NEXA is trained with $N = 10$ Qwen2.5-1.5B agents on either AQUA-RAT or GSM8K, then evaluated without retraining on both tasks. Figure 2 compares same-task training against cross-task training at two tested team sizes, $N = 5$ and $N = 20$. Across all four settings, the transfer gap remains small: 0.18 and 0.14 points on AQUA-RAT, and 0.08 and 0.05 points on GSM8K. This suggests that NEXA may learn a reusable response-conditioned rule rather than merely memorizing a task-specific pattern, although this requires further confirmation under more heterogeneous model families and agent pools.

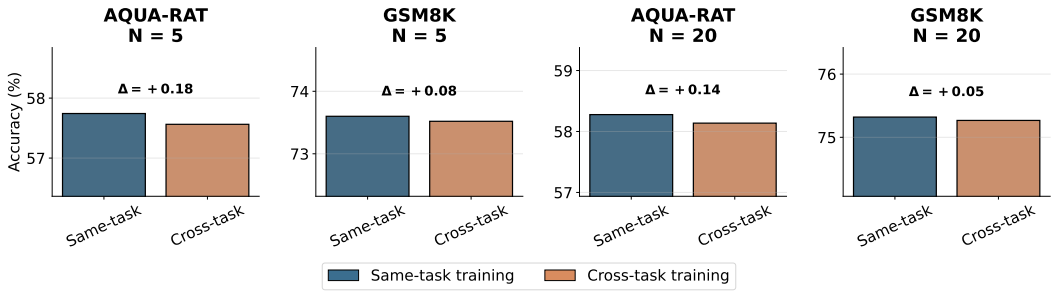

Figure 2: Task-transfer comparison for NEXA on Qwen2.5-1.5B.

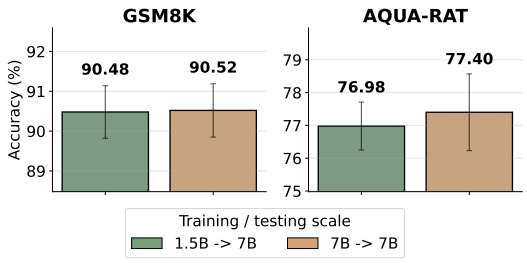

Figure 3: Model-scale transfer for NEXA on GSM8K and AQUA-RAT.

Model scale generalizability. The 1.5B-trained policy is evaluated without retraining on 7B, 14B, and 32B agents. It closely matches a policy trained directly on 7B (GSM8K: 90.48 vs. 90.52; AQUA-RAT: 76.98 vs. 77.40). On the harder GSM-Hard benchmark, frozen-policy NEXA reaches 63.13 ± 1.36 at 14B and 64.67 ± 0.31 at 32B, versus single-agent 60.73 ± 1.01 and 60.87 ± 0.58. On AIME-2024 it improves from 11.11 ± 1.92 to 17.78 ± 1.92 at 14B and from 20.00 ± 0.00 to 24.44 ± 1.93 at 32B (both $p < 10^{-4}$). Appendix E reports full same-draft and SelfOrg comparisons.

Model generation transfer. Finally, we evaluate whether the learned communication policy remains usable when the underlying model is updated to a newer generation. NEXA trained on Qwen2.5-1.5B is evaluated without retraining on Qwen3.5-2B (Qwen Team, 2026) and compared against a policy trained directly on Qwen3.5-2B. At $N = 5$, the transferred policy reaches 77.40, compared with 77.73 for the target-generation policy. The resulting 0.17-point gap suggests that

an existing policy can remain effective after a model upgrade, reducing the need to retrain the communication controller every time the base model is changed.

4.4 How Communication Changes Answers

We further analyze how NEXA changes answers after communication by decomposing each example according to whether the initial draft (parallel execution responses) and final answer (sequential execution responses) are correct. Figure 5 reports rescue, harm, and preservation rates for Qwen2.5-7B agents on GSM8K. As the tested team size increases from $N = 5$ to $N = 20$, the rescue rate rises from 19.2% to 23.8%, showing that additional agents provide useful opportunities for correcting initially wrong answers. At the same time, harm remains low, between 1.6% and 2.5%, while preservation stays above 97.5% across all tested values of N. These results suggest that NEXA does not simply perturb answers through extra communication; it mostly preserves correct predictions while selectively improving initially incorrect ones.

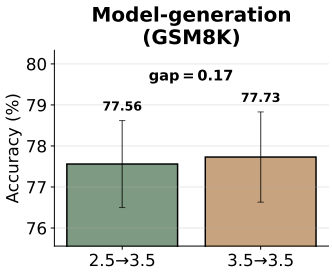

Figure 4: Model-generation transfer for NEXA.

Section 4.6 further examines whether these answer changes require dense communication.

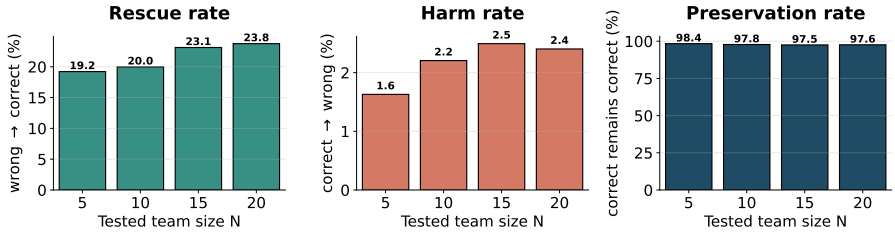

Figure 5: Policy behavior analysis for NEXA with Qwen2.5-7B agents on GSM8K. Rescue, harm, and preservation rates compare initial draft correctness with final answer correctness after communication.

4.5 Ablations

The contribution-order ablation reuses the same encoder and initial drafts so that only the ordering changes. Table 3 shows that descending contribution order outperforms ascending order by $3.29/3.47$ points and query-seeded random order by $4.60/2.40$ points on AQUA-RAT/GSM8K. The remaining encoder, aggregation, policy-architecture, and optimizer controls are reported with their artifacts in Appendix F.3.

Table 3: Contribution-order ablation with fixed encoder and identical drafts ($N = 5$, three repeats).

Ordering	AQUA-RAT	GSM8K
Descending	58.01 ± 1.78	73.67 ± 1.21
Ascending	54.72 ± 0.79	70.20 ± 2.09
Query-seeded random	53.41 ± 3.01	71.27 ± 1.10

4.6 Communication Sparsity

Figure 6 reports the fraction of GSM8K examples whose predicted graph uses at most half of the possible edges. With Qwen2.5-1.5B agents, low-edge plans occur in 35.0% of examples at $N = 5$ and roughly 47-50% at larger team sizes. With 7B agents, the rate rises from 43.4% at $N = 5$ to more than 70% at $N = 15$ and $N = 20$. Thus, larger teams do not force dense interaction; stronger agents, in particular, often require fewer refinement edges.

5 Related Work

LLM-based multi-agent collaboration. Multi-agent LLM systems have been studied as role-based societies, conversational workflows, and dynamically routed agent networks. CAMEL instantiates

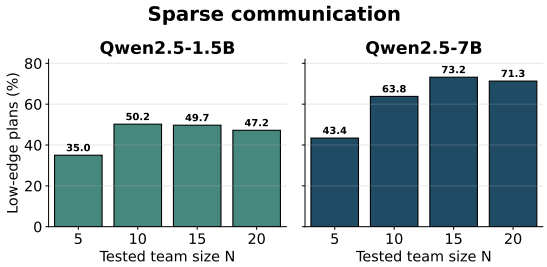

Figure 6: Low-edge plans on GSM8K (at most half of the candidate edges).

role-playing agents for cooperative problem solving (Li et al., 2023), ChatDev organizes specialized agents into staged communicative workflows (Qian et al., 2024), AutoGen provides a general framework for multi-agent conversations (Wu et al., 2024), and AgentVerse studies collaborative behaviors across agent groups (Chen et al., 2024). DyLAN adapts the active agent set during task solving (Liu et al., 2024), while multi-agent debate methods use disagreement to improve reasoning or factuality (Du et al., 2023; Liang et al., 2024). Multiagent finetuning further studies whether diverse reasoning chains can improve a base model through self-improvement (Subramaniam et al., 2025). These systems show that collaboration can improve reasoning, but they typically require a chosen communication protocol, a task-specific workflow, or an explicit judging mechanism. NEXA instead begins with independent responses and learns whether any sequential communication should occur at all.

Communication topology and workflow design. Several recent methods treat agent orchestration as a graph or workflow optimization problem. GPTSwarm represents language agents as optimizable computational graphs (Zhuge et al., 2024); AgentPrune removes unnecessary communication to reduce costs (Zhang et al., 2025a); G-Designer learns communication topologies with graph neural networks (Zhang et al., 2025b); and MacNet studies scaling laws for LLM-based multi-agent collaboration (Qian et al., 2025). Related work also explores training LLMs to construct multi-agent systems (Ye et al., 2025c), automated agentic workflow generation (Hu et al., 2025; Zhang et al., 2025c), decentralized evolutionary coordination (Yang et al., 2025), self-evolving agent profiles (Lu et al., 2024), heterogeneous multi-agent systems (Ye et al., 2025b), and unified experimental platforms for multi-agent evaluation (Ye et al., 2025a). NEXA is closest in spirit to graph-based topology learning but differs in three ways: the graph is conditioned on the realized response pool rather than only on a task or role template; the empty graph is a valid decision corresponding to pure parallel execution; and sparsity is controlled directly through an edge-count penalty in the task reward.

Judge-free aggregation and acyclicity. Response selection and ensemble fusion are often performed by majority voting, learned rankers, or generative fusion models such as LLM-Blender (Jiang et al., 2023); other systems introduce credibility scores or adversary-resistant judges (Ebrahimi et al., 2025). SelfOrg takes a different route by estimating response contribution from semantic embeddings and using that signal to organize communication without an external judge (Tastan et al., 2026). Its contribution score is motivated by Shapley-style valuation (Shapley, 1953; Tastan et al., 2025) and can be computed from sentence embeddings (Reimers & Gurevych, 2019). NEXA keeps this judge-free contribution ordering but replaces the stochastic self-organization rule with a trainable structure policy. The method, therefore, preserves the stable ordering principle of SelfOrg while learning the forward edges that determine whether and where refinement should happen.

6 CONCLUSION

We introduced NEXA, a response-conditioned policy that bridges parallel and sequential multi-agent execution by learning sparse acyclic communication graphs from initial agent drafts. It matches the strongest baseline within uncertainty while using 35.4% fewer tokens and running about $2.5\times$ faster at $N = 5$. Same-draft controls isolate communication gains, and the frozen policy transfers across tasks, agent counts, model generations, heterogeneous teams, and 14B/32B agents on harder benchmarks.

AI USE STATEMENT

Generative AI tools were used to assist with drafting and editing portions of the manuscript for clarity, organization, and presentation. They were not used to generate experimental data or determine the reported results. All AI-assisted text was reviewed by the authors for accuracy and consistency with the underlying experiments. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work studies the orchestration of large language model agents and does not involve human subjects or the collection of private or sensitive data. Experiments use established public benchmarks and publicly available model families. Like other multi-agent language-model systems, NEXA may propagate or reinforce errors present in its constituent models; in particular, semantic agreement among agents does not guarantee correctness. We discuss this and related limitations in Appendix A. The authors have read and adhere to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

We provide the algorithmic specification of NEXA in Appendix B, experimental and implementation details in Appendix D, and additional controlled evaluations and ablations in the remaining appendices. These include statistical analyses, same-draft controls, latency measurements, transfer experiments, encoder and policy controls, and controller-scaling measurements. The main text specifies the policy architecture, training objective, response encoder, aggregation procedure, and principal hyperparameters used in our experiments.

REFERENCES

Ding Chen, Qingchen Yu, Pengyuan Wang, Wentao Zhang, Bo Tang, Feiyu Xiong, Xinchi Li, Minchuan Yang, and Zhiyu Li. xverify: Efficient answer verifier for reasoning model evaluations, 2025. URL https://arxiv.org/abs/2504.10481.

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. URL https://arxiv.org/abs/2107.03374.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=EHg5GDnyq1.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv: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 *Forty-first International Conference on Machine Learning*, 2023.

Sana Ebrahimi, Mohsen Dehghankar, and Abolfazl Asudeh. An adversary-resistant multi-agent llm system via credibility scoring. *arXiv preprint arXiv:2505.24239*, 2025.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10764–10799. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/gao23f.html.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=t9U3LW7JVX.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14165–14178, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.792. URL https://aclanthology.org/2023.acl-long.792/.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=3IyL2XWDkG.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17889–17904, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.992. URL https://aclanthology.org/2024.emnlp-main.992/.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 158–167, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1015. URL https://aclanthology.org/P17-1015/.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic LLM-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=XII0Wp1XA9.

Siyuan Lu, Jiaqi Shao, Bing Luo, and Tao Lin. Morphagent: Empowering agents through self-evolving profiles and decentralized collaboration. *arXiv preprint arXiv:2410.15048*, 2024.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. 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. 15174–15186, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.810. URL https://aclanthology.org/2024.acl-long.810/.

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. Scaling large language model-based

multi-agent collaboration. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=K3n5jPkrU6.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

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

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084.

Lloyd S Shapley. A value for n-person games. In Harold W. Kuhn and Albert W. Tucker (eds.), *Contributions to the Theory of Games II*, pp. 307–317. Princeton University Press, Princeton, 1953.

Vighnesh Subramaniam, Yilun Du, Joshua B. Tenenbaum, Antonio Torralba, Shuang Li, and Igor Mordatch. Multiagent finetuning: Self improvement with diverse reasoning chains. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=JtGPIZpOrz.

Nurbek Tastan, Samuel Horváth, and Karthik Nandakumar. Aequa: Fair Model Rewards in Collaborative Learning via Slimmable Networks. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59210–59236. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/tastan25a.html.

Nurbek Tastan, Samuel Horváth, and Karthik Nandakumar. Stochastic self-organization in multi-agent systems. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=rS3Jb9AAej.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Yingxuan Yang, Huacan Chai, Shuai Shao, Yuanyi Song, Siyuan Qi, Renting Rui, and Weinan Zhang. Agentnet: Decentralized evolutionary coordination for LLM-based multi-agent systems. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=tXqLxHlb8Z.

Rui Ye, Keduan Huang, Qimin Wu, Yuzhu Cai, Tian Jin, Xianghe Pang, Xiangrui Liu, Jiaqi Su, Chen Qian, Bohan Tang, et al. Maslab: A unified and comprehensive codebase for llm-based multi-agent systems. *arXiv preprint arXiv:2505.16988*, 2025a.

Rui Ye, Xiangrui Liu, Qimin Wu, Xianghe Pang, Zhenfei Yin, Lei Bai, and Siheng Chen. X-mas: Towards building multi-agent systems with heterogeneous llms. *arXiv preprint arXiv:2505.16997*, 2025b.

Rui Ye, Shuo Tang, Rui Ge, Yaxin Du, Zhenfei Yin, Siheng Chen, and Jing Shao. MAS-GPT: Training LLMs to build LLM-based multi-agent systems. In *Forty-second International Conference on Machine Learning*, 2025c. URL https://openreview.net/forum?id=3CiSpY3QdZ.

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 *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=LkzuPorQ5L.

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 *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=LpE54NUnmO.

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. AFlow: Automating agentic workflow generation. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=z5uVAKwmjf.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62743–62767. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhuge24a.html.

CONTENTS

1 Introduction **1**

2 Problem Formulation and Preliminaries **2**

3 Methodology **3**

3.1 System Overview 3

3.2 Contribution-Defined Order and DAG Validity 3

3.3 Response-Conditioned Graph Policy 4

3.4 Response Propagation and Aggregation 4

3.5 Training Objective 5

4 Experiments **6**

4.1 Experimental Setup 6

4.2 Main Results 6

4.3 Generalizability Across Different Axes 7

4.4 How Communication Changes Answers 8

4.5 Ablations 8

4.6 Communication Sparsity 8

5 Related Work **8**

6 Conclusion **9**

A Limitations **15**

B Algorithm **15**

C Policy Behavior with Smaller Agents **15**

D Experimental Settings **16**

E Additional Controlled Evaluation **16**

E.1 Statistical reliability 16

E.2 Same-draft parallel control and latency 17

F Additional Experiments **17**

F.1 Extended Task Transfer 17

F.2 Larger-model transfer 18

F.3 Encoder, aggregation, and policy controls 19

F.4 Heterogeneous agents, tool use, and controller scaling 20

A LIMITATIONS

NEXA is evaluated primarily on reasoning and programming benchmarks where answer correctness can be measured reliably. A bounded code-execution study and a heterogeneous-agent experiment broaden this scope, but they do not establish performance for open-ended generation, long-horizon tool use, state tracking, or multi-turn planning.

The method also depends on response embeddings. Semantic similarity is not logical correctness: shared hallucinations can form a high-consensus cluster, while a rare correct specialist can be placed late and cannot send reverse feedback to earlier nodes in the single forward pass. Stronger encoders sharpen contribution ordering but do not remove this common-mode failure, and they add substantial latency.

Finally, NEXA deliberately uses one parallel draft round as the evidence-gathering stage. Initial calls scale linearly with the team size, but attention and the candidate-edge tensor scale quadratically. Controller measurements remain below one millisecond and 2 MB of incremental memory through $N = 100$, yet much larger pools may require per-query agent selection or hierarchical edge pruning. The edge penalty is provider-independent and does not directly optimize critical-path latency, monetary API cost, or prompt-cache behavior; heterogeneous-agent deployment may therefore benefit from rewards that explicitly price activated nodes and graph depth.

B ALGORITHM

Algorithm 1 NEXA

Require: Query $\mathcal{Q}$, agents $\{\mathcal{A}_n\}_{n=1}^{N}$, encoder f, policy $p_\theta(\mathcal{E} \mid \mathcal{X}, \pi)$

for $n = 1$ to N **do**
  $\mathcal{R}_n^{(0)} \leftarrow \mathcal{A}_n(\mathcal{Q}), \quad r_n \leftarrow f(\mathcal{R}_n^{(0)})$
end for
$r_{\text{avg}} \leftarrow \frac{1}{N}\sum_n r_n, \quad \psi_n \leftarrow \cos(r_n, r_{\text{avg}})$
$\pi \leftarrow \text{argsort}(\{\psi_n\}_{n=1}^{N}; \text{desc})$
$\mathcal{X} \leftarrow [r_1, \ldots, r_N]^\top, \quad \mathcal{H} \leftarrow \text{Enc}_\theta(\mathcal{X})$
Compute masked forward logits $\ell_{m \to n}$
Sample/decode $\mathcal{E} \sim p_\theta(\cdot \mid \mathcal{X}, \pi)$
if $\mathcal{E} = \varnothing$ **then** return centroid response from $\{\mathcal{R}_n^{(0)}\}_{n=1}^{N}$
for each node n in order π **do**
  $\text{Pa}(n) \leftarrow \{m : (m \to n) \in \mathcal{E}\}$
  $$\mathcal{R}_n^{(1)} \leftarrow \begin{cases} \mathcal{A}_n\Big(\mathcal{Q}, \{\mathcal{R}_m^{(\star)} : m \in \text{Pa}(n)\}\Big), & \text{Pa}(n) \neq \varnothing, \\ \mathcal{R}_n^{(0)}, & \text{otherwise.} \end{cases}$$
end for
return centroid response from $\{\mathcal{R}_n^{(1)}\}_{n=1}^{N}$

C POLICY BEHAVIOR WITH SMALLER AGENTS

We observe the same qualitative behavior with Qwen2.5-1.5B agents. As shown in Figure 7, the rescue rate increases from 15.6% at $N = 5$ to 20.6% at $N = 20$, while the preservation rate remains above 92.8% for all tested team sizes. Harm is higher than in the 7B setting (refer to Figure 5), ranging from 5.6% to 7.2%, which is expected given the weaker base agents. Nevertheless, the policy still improves a meaningful fraction of initially wrong answers while preserving the vast majority of initially correct ones, indicating that the same communication behavior appears even with smaller models.

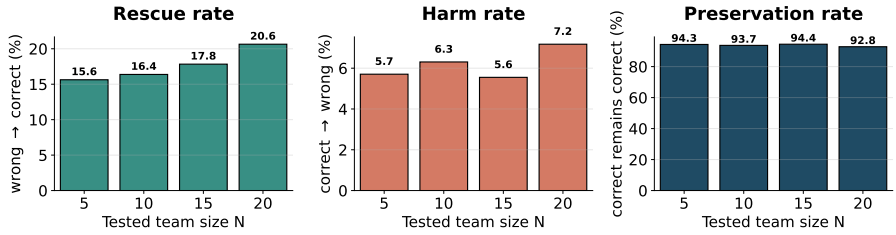

Figure 7: Policy behavior analysis for NEXA with Qwen2.5-1.5B agents on GSM8K. Rescue, harm, and preservation rates are computed by comparing each initial draft answer with the final answer after communication. NEXA rescues 15.6%-20.6% of initially wrong answers while preserving 92.8%-94.4% of initially correct answers across tested team sizes.

D EXPERIMENTAL SETTINGS

Compute resources. All experiments were run on NVIDIA A100 40GB GPUs. The same GPU class was used both to serve the LLM agents during multi-agent inference and to train the NEXA policy.

Backbone ablation setting. Transformer, GNN, and shallow-MLP variants receive the same 32 AQUA-RAT training examples for five complete passes. Validation alone selects each checkpoint, and every architecture is evaluated on the exact same test drafts. GSM8K is a zero-shot evaluation of the AQUA-RAT-selected checkpoints. The GNN adapts GDesigner's graph architecture from role-specific design to response-conditioned edge prediction.

Policy-optimization ablation setting. For the policy-optimization ablation, we compare the policy-gradient objective used in NEXA against a GRPO-style update. Both use Qwen2.5-1.5B agents, response-only inputs, contribution-based ordering, weighted aggregation, and XVerify-based evaluation at temperature 0.0. Evaluation is conducted on AQUA-RAT at $N = 5$ for both same-task training, AQUA-RAT→AQUA-RAT, and cross-task transfer, GSM8K→AQUA-RAT.

Both optimizer variants use the Transformer backbone, 50 policy-training iterations, batch size 32, learning rate 0.1, hidden dimension 128, one Transformer layer, one attention head, dropout 0.3, Adam, gradient clipping 1.0, and a batch-mean baseline. GRPO uses four rollouts.

E ADDITIONAL CONTROLLED EVALUATION

All controls in this section reuse identical initial drafts whenever the compared methods permit it. Reported uncertainty is the standard deviation over independent runs.

E.1 STATISTICAL RELIABILITY

For average-accuracy uncertainty, each run's average across the three tasks is one observation; the standard deviation is then computed across runs. Table 4 corrects the earlier practice of averaging per-task deviations. With $z = \Delta/\sqrt{s_1^2/3 + s_2^2/3}$, NEXA differs significantly from the single-agent and other trainable-topology baselines, but not from SelfOrg*. We consequently make an accuracy-cost claim rather than claiming significant raw-accuracy superiority over SelfOrg*.

Table 4: Average accuracy with run-level corrected standard deviations and two-sided z-tests against NEXA.

Method	Avg. acc.	Δ	z	p
NEXA	60.90 ± 0.68	–	–	–
SelfOrg*	60.36 ± 1.32	$+0.54$	0.63	0.53
Single	57.70 ± 1.84	$+3.20$	2.82	0.005
CoT	56.95 ± 0.68	$+3.95$	7.09	< 0.001
GPTSwarm	54.01 ± 0.73	$+6.89$	12.00	< 0.001
AgentPrune	52.71 ± 0.93	$+8.19$	12.34	< 0.001
GDesigner	51.79 ± 0.74	$+9.11$	15.69	< 0.001
Self-consistency	46.96 ± 1.63	$+13.94$	13.67	< 0.001

E.2 SAME-DRAFT PARALLEL CONTROL AND LATENCY

Table 5 isolates communication from the benefit of drawing more parallel candidates. Communication is positive at every evaluated team size on both tasks, although the magnitude varies by response pool and task.

Table 5: Accuracy-point change from NEXA communication relative to pure-parallel aggregation of the exact same drafts. Results use three repeats per N and five on AQUA-RAT.

Task	$N = 5$	$N = 10$	$N = 15$	$N = 20$
AQUA-RAT	$+1.50$	$+0.08$	$+2.13$	$+1.97$
GSM8K	$+1.60$	$+0.80$	$+1.00$	$+1.33$

Latency is measured on GSM8K over three 100-query runs per setting using four prefix-cached replicas and 16 concurrent queries; warm-ups are excluded. Drafts run concurrently and refinements are batched by graph level. Fixed one-round SelfOrg takes 19.89 ± 0.20 s at $N = 5$, compared with 7.89 ± 0.07 s for NEXA. The controller itself is negligible relative to the LLM consolidation path.

Table 6: Measured wall-clock latency. Times are seconds per example except the embedding-plus-controller column.

N	Parallel (s)	NEXA (s)	Controller (ms)	Consolidation (s)
5	2.68 ± 0.05	7.89 ± 0.07	20.8	5.18
10	3.35 ± 0.10	15.37 ± 0.23	52.0	11.95
15	3.91 ± 0.16	22.66 ± 0.71	81.3	18.64
20	4.31 ± 0.05	31.20 ± 0.72	168.2	26.69

F ADDITIONAL EXPERIMENTS

F.1 EXTENDED TASK TRANSFER

We additionally provide studies on whether a policy trained on one source task can transfer to additional target tasks beyond the main transfer experiments. In this setting, NEXA is trained on GSM8K and evaluated without retraining on GSM-Hard (Gao et al., 2023), HumanEval (Chen et al., 2021), and MMLU (Hendrycks et al., 2021). We compare against single-call and chain-of-thought baselines using the same target-task evaluation protocol. As shown in Table 7, the GSM8K-trained NEXA policy improves over both baselines on GSM-Hard and HumanEval and remains comparable to the baselines on MMLU. These results suggest that the learned communication policy can transfer beyond the training task, especially when the target task benefits from structured multi-agent reasoning.

Table 7: Extended task-transfer results. NEXA is trained on GSM8K and evaluated without retraining on GSM-Hard, HumanEval, and MMLU. Accuracy is reported as mean ± standard deviation.

Trained on	Tested on	Single	CoT	NEXA
GSM8K	GSM-Hard	34.40±0.72	36.07±0.31	**37.13±1.21**
	HumanEval	50.41±3.73	45.93±1.27	**51.42±0.70**
	MMLU	**52.07±1.03**	51.27±1.50	52.00±1.06
	Avg. Acc.	45.63	44.42	**46.85**
	Avg. Rating	2.00	2.67	**1.33**

F.2 LARGER-MODEL TRANSFER

The controller trained with 1.5B agents is frozen for every evaluation in Tables 8, 9, 10, and 11. On the easier GSM8K task, all 14B/32B methods approach saturation; the harder GSM-Hard and AIME-2024 evaluations retain headroom and show gains at both scales.

Table 8: Frozen-policy transfer to larger Qwen2.5 agents on AQUA-RAT. Methods reuse the same draft pools.

Model	Single	CoT	Parallel	SelfOrg⋆	NEXA
14B	77.17 ± 0.39	79.00 ± 0.82	78.22 ± 0.45	81.23 ± 0.82	81.63 ± 1.82
32B	79.92 ± 0.79	79.79 ± 1.94	79.79 ± 2.02	79.66 ± 0.23	80.18 ± 0.60

Table 9: Frozen-policy transfer to larger Qwen2.5 agents on GSM8K. Methods reuse the same draft pools.

Model	Single	CoT	Parallel	SelfOrg⋆	NEXA
14B	93.93 ± 0.64	94.07 ± 0.12	94.00 ± 0.69	94.67 ± 0.23	94.40 ± 0.69
32B	95.20 ± 0.53	95.07 ± 0.81	95.40 ± 0.35	94.73 ± 0.42	95.40 ± 0.20

At 14B, NEXA uses $7.86/7.91$ calls and $8{,}895.6/6{,}106.2$ tokens per AQUA-RAT/GSM8K example, compared with 10 calls and $13{,}575.6/9{,}450.0$ tokens for SelfOrg⋆. At 32B the corresponding NEXA costs are $7.83/7.89$ calls and $7{,}989.4/5{,}825.1$ tokens, versus 10 calls and $12{,}614.8/9{,}214.1$ tokens for SelfOrg⋆.

GSM-Hard. The frozen controller improves on every GSM-Hard comparison at both scales. Relative to Single, the gains are 2.40 points at 14B and 3.80 points at 32B; relative to SelfOrg⋆, the gains are 0.86 and 1.40 points.

Table 10: Frozen-policy transfer to GSM-Hard at larger model scales.

Model	Single	CoT	Parallel	SelfOrg⋆	NEXA
14B	60.73 ± 1.01	60.93 ± 0.58	61.33 ± 1.30	62.27 ± 1.29	63.13 ± 1.36
32B	60.87 ± 0.58	63.93 ± 0.76	63.20 ± 0.92	63.27 ± 1.01	64.67 ± 0.31

AIME-2024. AIME-2024 is reported separately because its evaluation compares NEXA directly with Single and uses a distinct significance analysis. The frozen controller gains 6.67 points at 14B and 4.44 points at 32B; both differences are significant under the reported two-sided z-tests.

Table 11: Frozen-policy transfer to AIME-2024. Tests are against the Single baseline.

Model	Single	NEXA	Gain	p
14B	11.11 ± 1.92	17.78 ± 1.92	$+6.67$	2.12×10^{-5}
32B	20.00 ± 0.00	24.44 ± 1.93	$+4.44$	6.61×10^{-5}

F.3 ENCODER, AGGREGATION, AND POLICY CONTROLS

The main paper reports the same-draft contribution-order ablation in Table 3. Descending order is a label-free prior rather than a correctness oracle, and the limitations section records the minority-specialist failure case. The remaining controls below test whether the conclusions depend on the response encoder, aggregation rule, policy backbone, or policy-optimization update.

Stronger encoders improve the rate at which correct drafts outrank incorrect ones, but this does not consistently improve final accuracy. Table 12 swaps only the ordering/selection encoder while the frozen controller remains in its MiniLM coordinate system. The deployed MiniLM path takes 14.2 ms for embedding and 7.1 ms for the controller per 13.4-second example.

Table 12: Encoder sensitivity with descending order. Added time is relative to MiniLM.

Task	Encoder	Accuracy	Added ms	Size
AQUA-RAT	MiniLM	58.01 ± 1.78	–	88 MB
	MPNet	58.01 ± 1.98	91	418 MB
	Qwen3-Embedding	58.92 ± 1.20	1,612	1.2 GB
GSM8K	MiniLM	73.67 ± 1.21	–	88 MB
	MPNet	73.53 ± 0.70	93	418 MB
	Qwen3-Embedding	73.67 ± 2.14	5,273	1.2 GB

The final selector returns an existing complete response nearest a contribution-weighted centroid; it is not winner-takes-all over a plain center and requires no additional LLM call. On identical final-response pools, weighting improves from 56.69 to 58.01 on AQUA-RAT and from 72.73 to 73.67 on GSM8K.

Table 13: Policy-backbone control. GSM8K is zero-shot transfer from the AQUA-RAT-selected checkpoint.

Backbone	Parameters	GSM8K	AQUA-RAT
Transformer	1,774,464	74.00 ± 1.64	57.09 ± 1.04
GNN	148,737	73.67 ± 2.01	57.22 ± 0.60
Shallow MLP	132,097	73.87 ± 1.94	54.72 ± 1.72

We also compare the policy-gradient objective used in NEXA with a GRPO-style alternative. The ablation is conducted on AQUA-RAT with Qwen2.5-1.5B agents at $N = 5$, considering both same-task training and cross-task transfer from GSM8K. As shown in Table 14, PG slightly outperforms GRPO in both settings, with 57.74 versus 57.56 for AQUA→AQUA and 57.56 versus 57.48 for GSM8K→AQUA. The gaps are small, indicating that the learned communication policy is not highly sensitive.

Table 14: Policy-optimization ablation on AQUA-RAT with N=5 Qwen2.5-1.5B agents.

Optimizer	AQUA→AQUA	GSM8K→AQUA
GRPO	57.56 ± 3.83	57.48 ± 2.30
PG	57.74 ± 2.31	57.56 ± 1.49

F.4 HETEROGENEOUS AGENTS, TOOL USE, AND CONTROLLER SCALING

With the policy frozen, a mixed team of three 1.5B and two 7B agents improves its identical parallel drafts in every repeat (Table 15). This establishes zero-shot compatibility, not an efficiency gain: tokens roughly triple, and 7B refinement calls are about four times slower than 1.5B calls.

Table 15: Mixed-team AQUA-RAT results at $N = 5$ over three repeats.

Method	**Accuracy**	**Tokens/example**
Parallel, same drafts	65.75 ± 1.04	3,257.5
NEXA, mixed team	66.93 ± 1.42	10,096.6

For tool-assisted code generation, each agent may execute its draft on prompt-visible examples and revise once; hidden tests are used only for final scoring. Most gain comes from the tool, while graph-based revision adds three further correct solutions and leaves the 264 task-runs with no visible examples unchanged.

Table 16: Tool-assisted code study on 492 task-runs using identical five-draft pools.

Method	**pass@1**	**Correct**
Parallel drafts, no tool	57.72	284/492
Tool only	61.18	301/492
NEXA: tool, one revision, graph	61.79	304/492

The controller's candidate-edge count is quadratic, but its measured overhead remains small through $N = 100$ (Table 17). Draft generation is still linear in N. For substantially larger pools, agent preselection or hierarchical edge pruning is a natural extension.

Table 17: Controller microbenchmark on an A100 GPU.

N	5	10	15	20	50	100
Candidate pairs	10	45	105	190	1,225	4,950
Median (ms)	0.786	0.784	0.799	0.804	0.795	0.794
Incremental peak (MB)	1.09	1.14	1.19	1.23	1.51	1.98

The edge-count reward is a provider-independent structural proxy, not seconds or dollars. A cost-aware extension could penalize active destination calls, critical-path depth, and measured/provider-specific invocation cost; we leave training under such a changed objective to future work.