WEAKLY-SUPERVISED COOPERATION IN DECENTRALIZED LLM MULTI-AGENT SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing state-of-the-art methods for learning inter-agent communication topologies in LLM multi-agent systems mostly learn through task outcomes. However, outcome-based rewards have significant challenges, such as requirement of correct task verification, a calibration phase, and also failing to adapt to changes in environment or team dynamics. In this work, we show that explicit task outcome signals are not necessary to learn effective communication topologies for cooperation. We propose a novel weakly-supervised method, MERIT-MAS, predicting dynamic and directed agent-to-agent communication topologies, only from agent-generated content. MERIT-MAS quantifies goal-conditioned relevance and novelty between agent pairs via cross-encoder rank scoring, and trains a topology network without using any task outcomes. Across established benchmarks and LLMs, our method achieves comparable performance (yielding overlapping 95% Wilson confidence intervals) with outcome-based SOTA baselines while spending 15% lesser agent-LLM tokens. Since MERIT-MAS does not need a calibration phase, it shows robustness under structural changes and scales seamlessly to larger agent counts, where methods that cache a single topology per episode collapse.

1 INTRODUCTION

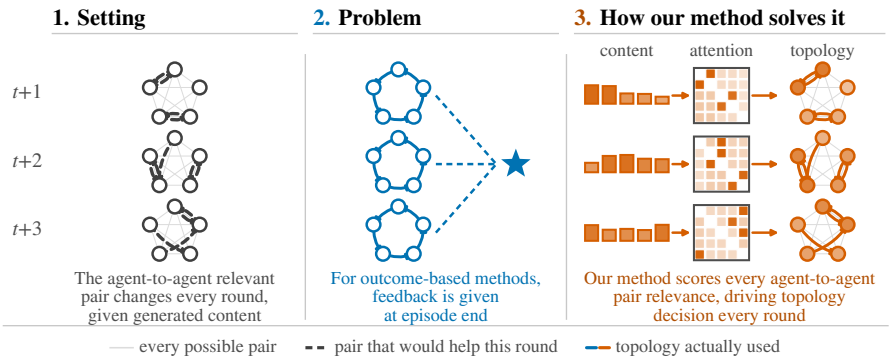


Figure 1: Each section shows three consecutive rounds $t{+}1, \ldots, t{+}3$ within an episode with N independent agents. **(1)** The communication topology between readers–sources required for the collective goal can differ across rounds and remains hidden from all agents. **(2)** Outcome-based topology learning methods get supervision signals from the final episode outcome ($\star$), which is available only after completing all T rounds, provided the outcome is verifiable. Some methods predict the topology in each round, while others decide a single communication topology per episode, which is illustrated here. **(3)** Our method estimates a round-specific topology directly from agents' content. The content retained by agents (bars) is processed via an inter-agent attention (heat map), where pairwise scores are used for directed communication. This operates without explicit final outcomes.

Coordination in multi-agent system (MAS) is crucial for solving complex problems where centralized orchestration is limited by either computation, capability, or privacy (Zhao et al., 2026; Park et al., 2026; Estornell et al., 2025; Chen et al., 2024). Established research community generally

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.

treats *decentralization* as a process, where agents can independently approximate optimal coordination policies using partial observations via rate-distortion theory (Dobbe et al., 2017). Extending this to dynamic systems, decentralized actor-critic algorithms allow agents with distinct rewards to maximize collective goals by diffusing value estimates across time-varying networks without a central orchestrator, provided rewards are accurately designed (Zhang et al., 2018). Modern LLM orchestration frameworks build on these paradigms, using consensus protocols for sub-task allocation and reasoning aggregation to improve scalability and fault tolerance, though sub-task allocation still requires prior competence (Wang et al., 2025a).

Despite many advancements, conceptual gaps remain between practical LLM-agent deployments and classical MAS principles (La Malfa et al., 2026). Empirically, MAS frameworks observe high failures due to design flaws, inter-agent misalignment, and weak task verification (Cemri et al., 2026). Since LLMs are typically developed to work in isolation, current designs struggle to demonstrate true cooperation, which is why explicit communication protocols sometimes show cooperation than purely experience-based training (Madmoun & Lahlou, 2026).

As La Malfa et al. (2026) discuss, a *decentralized* setup, where peer agents are independent, capable, and specialized, need a genuine cooperation mechanism for LLM-based multi-agent systems (LLM-MAS). Although a planner and (dynamic) sub-agents design is though common in practice, it has no topology to learn. The planner generally decides who communicates, which is a scripted coordination and hinges heavily on planner's ability. Since communication topology is one of the few coordination levers in LLM-MAS, the current state-of-the-art (SOTA) methods mostly train against task reward. GPTSwarm (Zhuge et al., 2024), AgentPrune (Zhang et al., 2025a), G-Designer (Zhang et al., 2024) and ARG-Designer (Li et al., 2026) frame topology selection as a graph optimization problem solved by reinforcement learning, graph neural networks, or supervised learning.

SOTA methods depend on a well-defined episode-reward, and they generally need an initial calibration phase, which may degrade under tasks which are not represented in the calibration phase (similar to real-world decision-making capable MAS). Training in frozen calibration phase treats the environment as a static target rather than an adaptive counterpart, leaving optimization vulnerable to Goodhart-style drift (Trivedi et al., 2026). In practice, reward-trained topologies fail to transfer and must be learned from scratch whenever the coordination dynamics shift (Cemri et al., 2026).

Based on these motivations, we formulate our research question: **can an LLM-MAS communication topology be learned purely from the agent-generated content, without explicitly conditioning on task outcome?** We conceptualize this setting in Figure 1. In this work, we show that *it is possible*, and an explicit-outcome free, content-driven method (ours) does not collapse the topology prediction compared to SOTA. Across three broad-coverage benchmarks and three reasoning LLM families, our method reaches comparable performance with outcome-trained SOTA, at 15% lower agent-LLM token cost, while remaining functional under a mid-run structural change and at increasing agent count, regimes where methods caching one topology per episode collapse right-away.

Our key contributions are as follows:

- We propose **MERIT-MAS**, a novel method to compute MEssage-Relevance for Inter-agent Topology learning in decentralized LLM-MAS, operating without task rewards, episode outcomes, calibration phase or central orchestration/planner agent.
- We score each agent pair by goal-conditioned relevance and content novelty, quantifying weak-supervision signal for topology learning.
- Comprehensive evaluation on broad and stress-test tasks to show our method's feasibility.

2 Related Work

Cemri et al. (2026) categorize over a third of LLM-MAS failures to inter-agent misalignment and coordination errors, across annotated traces in their study, and La Malfa et al. (2026) argue that the broad LLM-MAS field has loosely borrowed the vocabulary of classical multi-agent systems. Both of which motivate to consider *reader–source pair* as a trainable substrate.

Many SOTA methods train topology through task outcomes, and differ mainly in how the topology network is parameterized. GPTSwarm (Zhuge et al., 2024) learns one inclusion logit per directed

pair by REINFORCE (Williams, 1992), AgentPrune (Zhang et al., 2025a) adds a sparsity regularizer and resamples in every round, G-Designer (Zhang et al., 2024) uses a variational graph neural network, and ARG-Designer (Li et al., 2026) is supervised by a reward-verified pruning curriculum with an auto-regressive decoder. AgentDropout (Wang et al., 2025b) learns per-round node and edge masks by gradient on task performance and RADAR (Zhang et al., 2026b) generates one graph per query by conditional discrete diffusion under a structural-redundancy prior. On the other hand, DyLAN (Liu et al., 2023) comes closest to moving away from reward by pruning agents between rounds using LLM-scored importance, but that grades answer quality, which is a proxy for outcome. Some works hard-code the topology and study scaling (Qian et al., 2025), or search over agentic workflow programs through a validation metric (Zhang et al., 2025b; Zhou et al., 2026). Appendix A.1 covers the remaining works (Mao & Mirhoseini, 2026; Madmoun & Lahlou, 2026).

Trivedi et al. (2026) discuss that a fixed reward is not a valid stationary target. They define it as *solipsistic* paradigm and show that it is structurally vulnerable to Goodhart dynamics. Wang et al. (2026) provide the quantitative version of this, where continually adapting G-Designer or ARG-Designer to new tasks degrades their performance on previously learned tasks (*topology forgetting*).

Training a predictive communication network without outcome rewards has precedent outside of LLM-MAS. Contrastive objectives over messages of a trajectory, learning effective communication in classical MARL with no reward in the loss (Lo et al., 2024), and mutual-information or information-bottleneck objectives learning graph structure directly in numeric-state MARL (Ding et al., 2023; 2024; Wang et al., 2023). They all operate on low-dimensional numeric states, where metric is computable via vectors, which generally does not transfer to free-form LLM-MAS content.

Three properties that separate MERIT-MAS from related works are: *(i) The supervision signal does not influence the environment*, since our loss depends on relevance between agents' content, with no episode reward in it. *(ii) The communication topology predictor is not parameterized by agent identity*, since we score a (reader, source) pair from the text those two agents produced in a given round, without any explicit agent's position, unlike many reward-trained methods. *(iii) There is no calibration phase*, since the topology is predicted every round from agents' content, which also allows for adjusting to mid-run structural changes. No prior work, to our knowledge, trains a topology predictor using semantic relevance estimates derived from LLM-generated content or uses a cross-encoder re-ranker as a weakly-supervised relevance target.

We consider a prior method as an eligible baseline in our study when it (i) predicts directed agent reader–source pairs, (ii) runs without a central orchestrator, and (iii) has a specified procedure. The eligible baseline methods are in Table 1 (Section 4.1). Appendix A.1 covers other methods.

Table 1: MERIT-MAS compared to baselines. *Signal* is what the training objective is computed from. *Agent Metadata* marks methods which explicitly use agent profile and metadata. *Evaluated* marks the *methods implemented in our evaluation* (Section 4.1).

Method	Signal	Decided	Conditions on	Agent Metadata	Evaluated
DyLAN (Liu et al., 2023)	LLM answer score	per round	agent outputs	yes	×
GPTSwarm (Zhuge et al., 2024)	episode reward	per episode	agent pair	yes	✓
G-Designer (Zhang et al., 2024)	episode reward	per round	agent embeddings	yes	✓
AgentPrune (Zhang et al., 2025a)	episode reward	per round	agent pair	yes	✓
AgentDropout (Wang et al., 2025b)	task accuracy	per round	agent embeddings	yes	×
MacNet (Qian et al., 2025)	none (hard-coded)	fixed	—	n/a	×
RADAR (Zhang et al., 2026b)	task accuracy	per query	the query	yes	×
ARG-Designer (Li et al., 2026)	verified curriculum	per episode	agent pair, order	yes	✓
MERIT-MAS (ours)	**agent message relevance**	**per round**	**round's content**	**no**	✓

3 METHODOLOGY

MERIT-MAS learns *only* the communication topology of an LLM-MAS. It does not modify any individual agent, and no episode outcome is ever used. This section is organized in three parts: Section 3.1 formalizes the agents and our decentralized multi-agent system (MAS) model. Section 3.2 discusses the dynamic topology network design, including the agents' content representation and the weak-supervised objective. And, Section 3.3 discusses the loss function and training procedure.

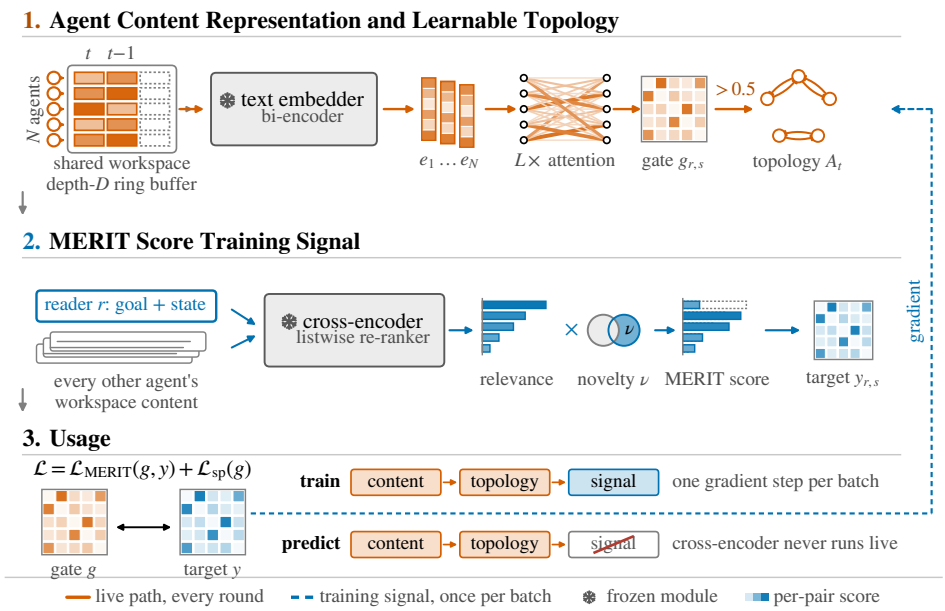


Figure 2: **MERIT-MAS Methodology.** *Snowflakes means frozen modules*. **(1)** A frozen bi-encoder transforms each agent's workspace slot with a depth-D ring buffer into one vector per round, which is the only input to the topology network. L self-attention blocks process the N vectors without any positional encoding, followed by scoring every ordered (reader, source) pair. Thresholding the topology gate $g_{r,s}$ at 0.5 provides the given round's directed adjacency matrix A_t, defining the communication. **(2)** Once per batch of B episodes, a frozen list-wise cross-encoder ranks every other agent's content against reader r's goal and content. A novelty factor discounts whatever r already holds, giving the weak-supervision target $y_{r,s}$. **(3)** The g is regressed onto y under a uniform penalty on open edges, without any calibration phase. The cross-encoder only during training.

3.1 System Model

Agents: An *agent* is defined as an autonomous actor with access to LLM and a harness (system instructions, tools, memory, live APIs, skills, knowledge base or sub agents) (Guo et al., 2026b; Chowa et al., 2026; Ferrag et al., 2026). Agents are independent to each-other and are free to evolve themselves, neither of which affects our topology prediction.

Episodes and Rounds: An *episode* is a single instance of a task, solved by a team of N independent agents over T synchronous rounds. At $t = 1$ of each episode, the task query is provided to all N agents. Each agent holds a private context, and explicitly publish shareable content for others, and a local history of its own content over rounds. At round t our method (Section 3.2) produces a directed adjacency set $A_t \in \{0, 1\}^{N \times N}$ from the current round's content, *before* t's agent execution. Each agent then reads the content the topology allows, act, and publish what they decide is relevant to the collective episode goal, independently. After $t = T$, a separate synthesizer agent, which is neither part of N agents nor gets affected by communication topology decisions, aggregates the final contents available in the shared workspace for the user-facing output.

Shared Workspace with Bounded Ring-buffer Slots: The *shared workspace* SW is a common read/write memory object (Nii, 1986) (not point-to-point channels) with one dedicated slot per agent. Reader r's observation at round $t + 1$ is assembled from source s contents $(r, s) \in A_t$. To bound the memory and avoid staleness, each slot is a ring buffer of depth D, retaining the D recent entries, while evicting the rest (Lamport, 1977; Packer et al., 2023). Agent's content therefore stays private until published in SW, read by the readers A_t allows, and expires beyond the depth D. The SW resets after every episode.

Decoupled Control and Execution: MERIT-MAS operates independently of the agent runtime. It predicts A_t once per round and updates its own parameters once per batch, while agents consume

allowed content as ordinary context. Consequently, the topology module neither modifies agent internals nor depends on their harness design.

3.2 Agent Communication and Topology Network

Agent Content Representation: At round t, agent r's workspace content $SW_r^{(t)}$ till D-deep ring buffer is encoded into $e_r \in \mathbb{R}^d$ by a frozen instruction-aware bi-encoder BE. We omit explicit agent-identity indicators and positional encoding in our design, so a routing decision is purely based on what the agents currently hold, irrespective of how the agents are positioned.

Learnable Topology Network: The routing module maps $E_t = [e_1, \ldots, e_N]^\top \in \mathbb{R}^{N\times d}$ to a directed adjacency matrix $A_t \in \{0,1\}^{N\times N}$, where $A_t[r,s] = 1.0$ allows reader r to see source s's workspace content at round t. A communication edge $A_t[r,s]$ is inherently a conditional relation. The usefulness (See Appendix A.2 for informal proposition) of source s to reader r depends not only on their local contents, but also on the information already available elsewhere in the team. Consequently, the desired score is $\ell_{r,s} = f(e_r, e_s, \{e_i\}_{i\neq r,s})$, rather than a function of only (e_r, e_s). We therefore use a permutation-equivariant set encoder (Lee et al., 2019) that contextualizes every agent representation using the contents of all other agents before pair scoring.

A linear projection to width d_h is followed by L blocks of M-head self-attention (Vaswani et al., 2017) over the N agent slots with post-block layer normalization (Ba et al., 2016), and each pair is scored from concatenated reader and source content:

$$H \;=\; \mathrm{Enc}_\theta(E_t) \in \mathbb{R}^{N\times d_h}, \qquad \ell_{r,s} \;=\; w^\top \big[\, H_r \,\|\, H_s \,\big] + \beta, \qquad w \in \mathbb{R}^{2d_h},\ \beta \in \mathbb{R} \tag{1}$$

with θ as network parameters. The role of Enc_θ is not to predict edges directly, but to transform each agent's content representation into a team-contextualized representation before pair scoring. Each contextualized representation H depends on the full set of agent contents. Thus, the score $\ell_{r,s}$ is conditioned on the global team state rather than only the reader-source pair itself. This allows the same pair (r,s) to receive different scores when the information held by other agents changes. As the Enc_θ captures the interactions among agents via self-attention, the scoring does not require explicit pair-wise cross-terms. Through self-attention, each contextualized vector H allows reader-specific preferences to be encoded in the representations. We empirically found no benefit from introducing bilinear or MLP interaction heads. Splitting $w = [w_\mathrm{r}; w_\mathrm{s}]$ gives $\ell_{r,s} = w_\mathrm{r}^\top H_r + w_\mathrm{s}^\top H_s + \beta$, which provides a parameter-efficient training update, costing only $2d_h + 1$ parameters across all N^2 pairs. Since θ is trained jointly with (w, β), the encoder contextualizes each H, so that contents itself are sufficient for the representation of the learning signal with edge (pair) relevance.

Without any explicit positional encoding or agent index, Enc_θ is permutation-equivariant for $\ell(PE_t) = P\,\ell(E_t)\,P^\top$ for every permutation matrix P. Hence, changing the agents' position does not affect the routing decision. This is a property of our network design by construction. The pair score $\ell_{r,s}$ is converted into a continuous communication gate through a logistic sigmoid:

$$g_{r,s} = \sigma(\ell_{r,s}), \quad A_t[r,s] = \mathbb{1}\big[g_{r,s} > \tfrac{1}{2}\big] \tag{2}$$

During routing execution, a directed reader–source edge is opened for communication when $g_{r,s} > 0.5$. The continuous gate value $g_{r,s}$ is optimized directly through Equation 4, while the thresholding is applied only during topology decision making (inference only). This design allows for a differentiable training objective while producing an explicit binary communication graph during execution.

Weak-Supervised Training Signal and the MERIT Score: The routing module is trained against a per-pair inter-agent message relevance score, defined as *MERIT score* $m_{r,s}$, where we estimate a proxy for how helpful source s's content may be to reader r for the episode's goal, without referencing the episode success (Zhang et al., 2026a). Once per k^{th} batch of B completed episodes in training, a frozen list-wise cross-encoder CE (Nasika et al., 2026; Nogueira & Cho, 2019) scores each candidate c_s against a goal-conditioned query q_r built from reader r's content, returning a relevance score $\in [0,1]$.

The raw relevance computed above can be biased. The available open-source CE reward familiar contents over new information, so a redundant restatement can outrank the content that might actually move the reader towards solution to episode's goal. Therefore, we scale it by a novelty factor $\nu_{r,s} \in [0,1]$, estimating semantic dissimilarity between the reader and source representations, and

normalize by running statistics:

$$\hat{m}_{r,s} \;=\; \mathrm{CE}(q_r,\, c_s)\cdot \nu_{r,s}, \qquad y_{r,s} \;=\; \sigma\!\left(\frac{\hat{m}_{r,s}-\mu_k}{\max(\kappa_k,\,\epsilon)}\right) \in (0,1),$$
$$\mu_k \;=\; \gamma\,\mu_{k-1} + (1-\gamma)\,\hat{m}_k, \qquad \kappa_k \;=\; \gamma\,\kappa_{k-1} + (1-\gamma)\,s_k \tag{3}$$

where $\hat{m}_k$ and s_k are the mean and standard deviation of $\{\hat{m}\}$ over batch k, γ is the EMA decay, $(\mu_0, \kappa_0) = (\hat{m}_1, s_1)$, with ϵ to guard the scale. $y_{r,s}$ is a semantic relevance proxy, not a causal communication utility, allowing our method to imitate the relevance induced by CE. We take $\nu_{r,s} = \frac{1-\cos(e_r, e_s)}{2} \in [0,1]$ which quantifies semantic novelty between the reader and source representations. Since both factors are non-negative, the product is a clean *relevance discounted by redundancy*. We preferred multiplication because it preserves monotonicity in both relevance and novelty and suppresses redundant highly-relevant content.

Equation 3 makes $y_{r,s}$ a *relative* score, computing on a pair against the recent score distribution, and not on an absolute scale. Raw CE scores might drift with task, phrasing and content length, whereas our routing decision only needs the pairs which are more worthy than the typical pairing. Due to our scaling, we don't need the calibration phase which other methods require. As (μ_0, κ_0) are seeded from the first batch's own scores, the target is never anchored to statistics from stale agent-team structure. $y_{r,s}$ estimates goal-conditioned semantic usefulness of source content to a reader.

3.3 Training Procedure

Loss Function: Let $\mathcal{P}_k$ be the ordered pairs with a *resolved* $y_{r,s}$ in batch k (unresolved pairs are those where a reader didn't produce valid output, and the particular round is omitted) and $\mathcal{E}_k$ be all possible ordered pairs across the batch's rounds. The loss regresses the $g_{r,s}$ (Equation 2) onto the $y_{r,s}$ (Equation 3) with a uniform price (penalty) for every open edge:

$$\mathcal{L}(\theta) \;=\; \underbrace{\frac{1}{|\mathcal{P}_k|}\sum_{(r,s)\in\mathcal{P}_k}\big(g_{r,s}-y_{r,s}\big)^2}_{\mathcal{L}_{\mathrm{MERIT}}} + \lambda_{\mathrm{sp}}\underbrace{\frac{1}{|\mathcal{E}_k|}\sum_{(r,s)\in\mathcal{E}_k} g_{r,s}}_{\mathcal{L}_{\mathrm{sp}}} \tag{4}$$

$\mathcal{L}_{\mathrm{MERIT}}$ is the only supervised term. Since $y_{r,s}$ is a continuous relevance target rather than a binary supervision signal, we formulate training as a regression problem. We found BCE, KL, and MSE loss comparable, hence use MSE for stability. Because y is centered on the recent score distribution, an MSE-trained g is already anchored from both directions. $\mathcal{L}_{\mathrm{sp}}$ is a differentiable penalty on open edges that balances the recall of useful edges against the context dilution (context-rot) caused by over-connection (Wan et al., 2026). Both the terms are means, so λ_{sp} is independent of T and N.

Training: At the end of each k^{th} batch of B episodes, the contents are scored, (μ_k, κ_k) are updated and applied in Equation 3, followed by one AdamW step (Loshchilov & Hutter, 2017) on $\mathcal{L}$. Since every batch updates the same parameters throughout deployment, MERIT-MAS works in online-learning setting where topology prediction and parameter updates continue, allowing adjustment to structural shifts without a calibration phase. *This is ordinary online descent, not continual learning*, so it does not have any replay buffer or anti-forgetting term. When a batch resolves no target ($\mathcal{P}_k = \varnothing$), the gradient step is skipped. $\mathcal{L}_{\mathrm{sp}}$ is never applied alone, so it doesn't drive the gate logits into saturation. The live topology prediction costs $O(N)$ BE calls per round and the batch training update covers $O(N^2TB)$ ordered pairs via CE. Algorithm 1 (Appendix A.3) states both topology prediction inference and batch training, where the inference is LiveRound alone.

4 Evaluation

4.1 Experiment Setup

Baselines: We use four reward-trained methods for the SOTA comparison: *GPTSwarm* (Zhuge et al., 2024) and *ARG-designer* (Li et al., 2026), which decide topology per episode, and *Agent-Prune* (Zhang et al., 2025a) and *G-designer* (Zhang et al., 2024), which decide every round. All four freeze their parameters after calibration phase. We also use three static topologies: *Isolated*, *Fully-connected*, and *Random* (each directed edge included independently at fixed probability, redrawn

each round), primarily to cover the range of possibilities. MERIT-MAS follows its ordinary online-learning protocol, whereas baselines follow their original protocol and stay frozen post-calibration.

Models: We use three frozen LLMs for Agents: Qwen3-235B-A22B-2507 (Yang et al., 2025), OpenAI-GPT-OSS-20B (Agarwal et al., 2025) and Moonshot Kimi K2 Thinking (Team et al., 2025). All agents share same LLM in a team, and only differ in harness. These three different LLMs with different reasoning ability let us separate a method's strength from the LLM it is evaluated on. For the frozen instruction-aware bi-encoder BE we use Qwen3-Embedding-0.6B (Zhang et al., 2025c), and for the frozen list-wise cross-encoder CE, jina-reranker-v3.5 (Nasika et al., 2026) is used.

Task scope: A topology learning can only be evaluated when routing decisions are possible. Therefore, we evaluate on tasks which satisfy three conditions: (i) $N \geq 3$, (ii) The communication structure is hidden from the agents, and (iii) The information agents need travels through the communication topology mechanism rather than a shared observation. Our focus is cooperation through communication rather than evaluating the intrinsic reasoning of the agents. Reasoning required by the task is just a substrate on which cooperation is performed. This choice allows the evaluation to isolate topology learning from the intrinsic capabilities of the underlying LLMs. Three benchmarks provide broad coverage for our evaluation: *Collab Meeting Scheduling* (Mahmud et al., 2025), *Concordia Pasture* (Smith et al., 2026) and *Marble Bargaining* (Zhu et al., 2025). We further include two more structural stress-tests tasks. *Hidden Teams* (Li et al., 2025) evaluates robustness to shift in team structure during mid-run. *Coalition Signal* (Guo et al., 2026a) is designed where over-connection carries no penalty and under-connection is fatal.

Metrics: Success (%) for every method is computed via each task's own scoring mechanism, at its native episode budget, along with Wilson 95% intervals (Brown et al., 2001) (Appendix A.6).

Configurations: Table 2 lists hyperparameters of MERIT-MAS. We perform a sweep on a small set of representative tasks (few episodes) and then fix it across every task and LLM combination reported below. Appendix A.4 provides more details about the selection procedure and sensitivities.

Table 2: **Configuration of the MERIT-MAS**, fixed across all tasks and LLMs combination. See Appendix A.4 for extended discussion.

	Configuration	Value	Governs
Network	L	2	self-attention blocks in Enc_θ
	M	4	attention heads per block
	d_h	64	head dimension
Objective	λ_{sp}	0.1	price on over-connection (Equation 4)
	γ	0.9	EMA decay for (μ_k, κ_k) (Equation 3)
Execution	D	2	workspace ring-buffer depth per agent slot
	B	4	episode batch size per optimizer step
	η	10^{-3}	AdamW learning rate (Loshchilov & Hutter, 2017)

4.2 RESULTS

Explicit-Outcome free approach does not collapse, and costs fewer tokens: Across the three broad-coverage tasks, as shown in Figure 3(a), *MERIT-MAS* achieves 58.7% against 56.7% for the strongest reward-trained baseline, while using only 111k tokens per episode against 132k. With learned method's overlapping Wilson intervals (Appendix A.6), our results show that MERIT-MAS achieves *comparable performance* rather than a significant win. This *comparable performance* is achieved without explicit outcome signal, and at least 15% fewer tokens than the closest baseline. We observe that topology learned without outcomes does not collapse topology prediction compared to outcome-based baselines, removes the need for verifiable outcomes, and optimizes the token cost.

A mid-run change of team dynamics: *Hidden Teams* (Li et al., 2025) task changes team membership once at the middle of evaluation run, after every reward-trained baseline's calibration is completed, as shown in Figure 3(b). *MERIT-MAS* is comparable (Appendix A.6) from baselines. Since it never calibrates, it keeps adjusting θ even after the change, unlike others. Continuous adaptation is a core property of MERIT-MAS by construction. Reward-trained baselines are evaluated under their original post-calibration protocols.

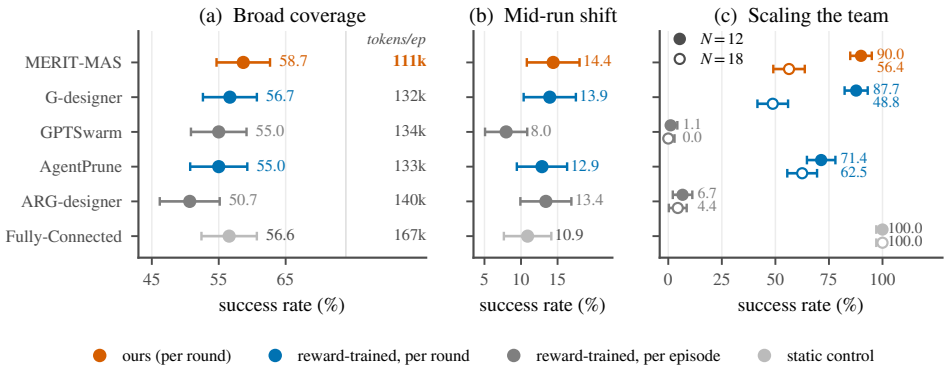


Figure 3: **Success (%), calculated as unweighted mean over the three LLM backbones** for each task at its native episode count. Wilson 95% intervals are plotted as bars using the per-LLM intervals (Appendix A.6 for detailed report). *Fully-Connected* is the *ceiling* static topology, while the remaining static topologies are in the Appendix A.6. **(a)** On tasks for broad coverage, MERIT-MAS shows the highest mean at the lowest token/episode cost. **(b)** Under a mid-run change of team structure MERIT-MAS is above the best baseline. **(c)** On a task where under-connection is fatal, the two methods that decide topology per episode are at the floor for both N, while others are functional.

Scaling agents where under-connection is fatal: *Coalition Signal* (Guo et al., 2026a) is designed so that over-connection does not penalize the outcome, whereas under-connection is fatal, making *Fully-Connected* the ceiling and *Isolated* the floor. In Figure 3(c), the two methods predicting topology per episode collapse at both $N{=}12$ and $N{=}18$, while every per-round method remains functional. Among the methods predicting topology per-round, *MERIT-MAS* stays similar to other baselines, with overlapping Wilson intervals (Appendix A.6), depicting comparable performance.

Appendix A.7 provides the complexity analysis of our topology module against other baselines.

4.3 Ablation Analysis

Components: In Table 3, we disable one component at a time from MERIT-MAS. The novelty correction (Equation 3) contributes up to 22.9 points. Without it the CE's bias for similar and redundant phrasing can outrank genuinely informative content for the episode's goal. The $\nu_{r,s}$ affects only the weak-supervision $y_{r,s}$ directly. The contextual set encoder allows topology decisions to depend on the distribution of information across the entire team of agents rather than on reader-source pairs in isolation. Replacing the contextual set encoder with *G-designer*'s graph-neural network (alternative network design from baselines) costs 32.7 points, suggesting that contextualized team-level representations are important for communication routing. Removing the sparsity term costs 12.3 points, showing that sparsity is important to avoid potential context dilution (context-rot; Section 3.3). Though retention depth's effect is almost neutral in our evaluation with only 2.0 points, we believe this primarily depends on the tasks. Intuitively, long-horizon agentic tasks and short-horizon queries will have different optimal D.

Table 3: **Component ablation.** Success (%) is average across the five tasks at 300 episodes each (not the individual task budget from Figure 3). *Full* is MERIT-MAS with every component present. *Ablated* removes or replaces only that particular component. Δ = Ablated − Full. The right-hand column names the methodological impact of each component.

Component	Full	Ablated	Δ	Methodological Impact
Novelty correction $\nu_{r,s}$ (Eq. 3)	70.0	47.1	-22.9	target (y) quality
Contextual set encoder (Eq. 1)	70.0	37.3	-32.7	content representation
Sparsity term $\mathcal{L}_{\text{sp}}$ (Eq. 4)	70.0	57.7	-12.3	connection density
Retention depth, D (Sec. 3.1) *(ablated column is mean of $D = \{1, 3, 6\}$)*	70.0	68.0	-2.0	agent content span

Training signal while the architecture is fixed: In Table 4, we use the architecture *G-designer*'s graph network (the closest baseline to MERIT-MAS by performance) and inference, and just swap the training signal to it. We run this on *G-designer*'s network (not on MERIT-MAS) to solely study the effect of training signal without attention encoder. We replace outcome-reward with the MERIT objective (Equation 4), which scores $+4.4$ points overall, while winning on two of three tasks. This observation supports our hypothesis that the weakly-supervised training signal (like MERIT-MAS's) is at least as good as the outcome-reward, on a network built for outcome-reward.

Tables 4 and 3 have different evaluation scope, containing only broad-coverage tasks and aggregation of two additional structural stress tests, respectively. The advantage of the contextual set encoder is therefore observed in structural-shift and scaling more than standard coordination benchmarks.

Table 4: **Training signal with *G-designer*'s architecture held fixed.** The graph network and inference rule are those of *G-designer*'s. The training signal is swapped. Success (%) is mean over the three LLMs at the episode budgets of Figure 3. Δ = MERIT mechanism in G-designer − G-designer's Reward mechanism. We evaluate on *G-Designer*'s network (not MERIT-MAS) to isolate the training signal's effect from the *MERIT-MAS*'s attention encoder.

Task	MERIT objective (Eq. 4)	Reward (*G-designer*)	Δ
Collab Meeting Scheduling	**91.7**	81.7	$+10.0$
Concordia Pasture	54.0	**55.3**	-1.3
Marble Bargaining	**37.7**	33.0	$+4.7$
Mean	**61.1**	56.7	$+4.4$

5 Conclusion

Communication topology is one of the direct cooperation mechanism in LLM multi-agent systems, where SOTA methods learn using the task outcomes. Our core research question is whether the explicit task outcome is necessary for learning effective communication topology for cooperation among independent agents in decentralized LLM-MAS, and we find that it is not.

Our proposed method, MERIT-MAS, learns a communication-topology predictor from semantic relevance estimates computed over agent-generated content, where a frozen list-wise cross-encoder ranks how relevant one agent's content is to another, given they all cooperate towards episode's goal. We also use a novelty factor, which discounts what the agent already knows. This allows us to train our communication topology network in weakly-supervised manner using semantic relevance targets rather than explicit task outcomes, with simple online descent learning, purely based on agent-generated content. We also use a permutation-equivariant network to avoid learning the topology based on agent's explicit position.

Through comprehensive evaluation, our method achieves comparable performance with outcome-reward-trained SOTA, while consuming 15% fewer agent-LLM tokens per episode on average, compared to the closest baseline. It also adjusts to changes in agent-team structure and scales at larger team sizes. MERIT-MAS provides an explicit-outcome free alternative to decentralized LLM-MAS communication topology paradigm, which is dominated by outcome-driven methods. Importantly, removing the verifiable rewards for training may improve robustness to changes in coordination structure and avoid any reward-bias, while decreasing the overall token-cost in LLM-MAS.

6 Limitations and Future Work

Our topology's training covers $O(N^2TB)$ ordered pairs. Though it is the only quadratic term in the method, the cost is directly proportional to CE's runtime, which is quite cheap compared to LLMs. An approximate top$-K_c$ filtering (Malkov & Yashunin, 2018) is a natural extension for faster relevance during de-coupled training, reducing the cost from $O(N^2TB)$ to $O(NK_cTB)$.

The quality of relevance quantification is decided by the current of-the-shelf open-source frozen bi-encoder and cross-encoder models. A parameter-efficient fine-tuning (Xu et al., 2023) of those models could enhance relevance estimation in specialized domains.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Lawrence D Brown, T Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical science*, pp. 101–117, 2001.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. Why do multi-agent llm systems fail? *Advances in Neural Information Processing Systems*, 38, 2026.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *International Conference on Learning Representations*, volume 2024, pp. 20094–20136, 2024.

Sadia Sultana Chowa, Riasad Alvi, Subhey Sadi Rahman, Md Abdur Rahman, Mohaimenul Azam Khan Raiaan, Md Rafiqul Islam, Mukhtar Hussain, and Sami Azam. From language to action: a review of large language models as autonomous agents and tool users. *Artificial Intelligence Review*, 59(2):71, 2026.

Shifei Ding, Wei Du, Ling Ding, Jian Zhang, Lili Guo, and Bo An. Multiagent reinforcement learning with graphical mutual information maximization. *IEEE Transactions on neural networks and learning systems*, 36(11):19515–19524, 2023.

Shifei Ding, Wei Du, Ling Ding, Lili Guo, and Jian Zhang. Learning efficient and robust multi-agent communication via graph information bottleneck. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17346–17353, 2024.

Roel Dobbe, David Fridovich-Keil, and Claire Tomlin. Fully decentralized policies for multi-agent systems: An information theoretic approach. *Advances in neural information processing systems*, 30, 2017.

Andrew Estornell, Jean-Francois Ton, Yuanshun Yao, and Yang Liu. Acc-collab: An actor-critic approach to multi-agent llm collaboration. In *International Conference on Learning Representations*, volume 2025, pp. 90558–90581, 2025.

Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. From llm reasoning to autonomous ai agents: A comprehensive review. *IEEE Access*, 2026.

Dongxin Guo, Jikun Wu, and Siu-Ming Yiu. Coalition formation in llm agent networks: Stability analysis and convergence guarantees. *arXiv preprint arXiv:2604.14386*, 2026a.

Jianyuan Guo, Zhiwei Hao, Chengcheng Wang, Cheng Fan, Tingzhang Luo, Hongguang Li, Ying Gao, Hefei Mei, Jiankun Peng, Rongjian Xu, et al. From question answering to task completion: A survey on agent system and harness design. *arXiv preprint arXiv:2606.20683*, 2026b.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

Emanuele La Malfa, Gabriele La Malfa, Samuele Marro, Jie Zhang, Elizabeth Black, Michael Luck, Philip Torr, and Michael Wooldridge. Large language models miss the multi-agent mark. *Advances in Neural Information Processing Systems*, 38, 2026.

Leslie Lamport. Concurrent reading and writing. *Communications of the ACM*, 20(11):806–811, 1977.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 23142–23150, 2026.

Yuxuan Li, Aoi Naito, and Hirokazu Shirado. Systematic failures in collective reasoning under distributed information in multi-agent llms. *arXiv preprint arXiv:2505.11556*, 2025.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. *arXiv preprint arXiv:2310.02170*, 2023.

Yat Long Richie Lo, Biswa Sengupta, Jakob Foerster, and Michael Noukhovitch. Learning multi-agent communication with contrastive learning. In *International Conference on Learning Representations*, volume 2024, pp. 51029–51046, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Hachem Madmoun and Salem Lahlou. Communication enables cooperation in llm agents: A comparison with curriculum-based approaches. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 307–321, 2026.

Saaduddin Mahmud, Eugene Bagdasarian, and Shlomo Zilberstein. Collab: A framework for designing scalable benchmarks for agentic llms. In *Workshop on Scaling Environments for Agents*, 2025.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Yuzhen Mao and Azalia Mirhoseini. Decentralized multi-agent systems with shared context. *arXiv preprint arXiv:2606.10662*, 2026.

Christina Nasika, Feng Wang, Antonis Krasakis, and Han Xiao. jina-reranker-v3. 5: An efficient listwise reranker with hybrid attention and self-distillation. *arXiv preprint arXiv:2607.18152*, 2026.

H Penny Nii. Part one: The blackboard model of problem solving and the evolution of blackboard architectures. *AI magazine*, 7(2):38–53, 1986.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Jongho Park, Vasilis Kontonis, Shivam Garg, Akshay Krishnamurthy, and Dimitris Papailiopoulos. Scaling discovery through test-time communication, 2026. URL `https://arxiv.org/abs/2609.21032`.

Chen Qian, Zihao Xie, YiFei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, et al. Scaling large language model-based multi-agent collaboration. In *International Conference on Learning Representations*, volume 2025, pp. 41488–41505, 2025.

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 and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

Chandler Smith, Marwa Abdulhai, Manfred Diaz, Marko Tesic, Rakshit Trivedi, Sasha Vezhnevets, Lewis Hammond, Jesse Clifton, Minsuk Chang, Edgar Duenez-Guzman, et al. Evaluating generalization capabilities of llm-based agents in mixed-motive scenarios using concordia. *Advances in neural information processing systems*, 38, 2026.

Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Rakshit S Trivedi, Natasha Jaques, Logan Cross, Alexander Sasha Vezhnevets, and Joel Z Leibo. Solipsistic superintelligence is unlikely to be cooperative. *arXiv preprint arXiv:2606.03237*, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yang Wan, Zheng Cao, Zhenhao Zhang, Zhengwen Zeng, Shuheng Shen, Changhua Meng, and Linchao Zhu. Mitigating conversational inertia in multi-turn agents. *arXiv preprint arXiv:2602.03664*, 2026.

Ji Wang, Kashing Chen, Xinyuan Song, Ke Zhang, Lynn Ai, Eric Yang, and Bill Shi. Symphony: A decentralized multi-agent framework for scalable collective intelligence. *arXiv preprint arXiv:2508.20019*, 2025a.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Wang Wang, Deheng Ye, and Zongqing Lu. Mutual-information regularized multi-agent policy iteration. *Advances in Neural Information Processing Systems*, 36:2617–2635, 2023.

Xuefei Wang, Jialu Wang, Fengbo Zhang, Yihan Hu, Di Zhang, Yutong Ye, Yikun Ban, Jun Han, and Ruijie Wang. Masfact: Continual multi-agent topology learning via geometry-aware posterior transfer, 2026. URL https://arxiv.org/abs/2605.17361.

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 *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24013–24035, 2025b.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pp. 641–649, 2024.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment (no. arxiv: 2312.12148). arxiv, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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. *arXiv preprint arXiv:2410.11782*, 2024.

Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Yu, and Tianlong Chen. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. In *International Conference on Learning Representations*, volume 2025, pp. 75389–75428, 2025a.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025b.

Kai Zhang, Xinyuan Zhang, Ejaz Ahmed, Hongda Jiang, Caleb Kumar, Kai Sun, Zhaojiang Lin, Sanat Sharma, Shereen Oraby, Aaron Colak, et al. Assomem: Scalable memory qa with multi-signal associative retrieval. In *International Conference on Learning Representations*, volume 2026, pp. 103385–103405, 2026a.

Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *International conference on machine learning*, pp. 5872–5881. PMLR, 2018.

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

Zhen Zhang, Wanjing Zhou, Juncheng Li, Hao Fei, Jun Wen, and Wei Ji. Radar: Redundancy-aware diffusion for multi-agent communication structure generation. *arXiv preprint arXiv:2605.09907*, 2026b.

Wanjia Zhao, Mert Yuksekgonul, Shirley Wu, and James Zou. Sirius: Self-improving multi-agent systems via bootstrapped reasoning. *Advances in Neural Information Processing Systems*, 38: 124475–124504, 2026.

Han Zhou, Xingchen Wan, Ruoxi Sun, Hamid Palangi, Shariq Iqbal, Ivan Vulić, Anna Korhonen, and Sercan Arik. Multi-agent design: Optimizing agents with better prompts and topologies. In *International Conference on Learning Representations*, volume 2026, pp. 15844–15872, 2026.

Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Daisy Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, et al. Multiagentbench: Evaluating the collaboration and competition of llm agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8580–8622, 2025.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Language agents as optimizable graphs, 2024. URL `https://arxiv.org/abs/2402.16823`.

AI USE STATEMENT

We used generative AI for: (1) Debugging infrastructure and CUDA issues and (2) Grammatical and spelling checks.

We have **not used** generative AI for: (1) Generate synthetic data sets, (2) Develop or polish theoretical models or conceptual frameworks, (3) Formulate mathematical claims, (4) Provide critical ingredients for proving mathematical claims, (5) Assist in the writing of proofs, (6) Propose or refine hypotheses, (7) Design or provide feedback on research methodology or experiments, (8) Implement methods, (9) Assist with translation, (10) Clean and reformat dataset, (11) Support qualitative and thematic data analysis, and (12) interpret results.

We take complete responsibility for all the content of this work, including text, claims or artifacts produced in this paper.

ETHICS STATEMENT

The work presented in this paper is an algorithmic contribution with benchmarks and baselines comparisons using publicly available models, datasets and algorithms, without any privacy, security or ethical violations.

Reproducibility statement

The paper primarily introduces a new algorithm, which is well described in Section 3 and Appendix A.3. The baseline empirical studies are conducted based on the code, designs and datasets described by the original publications mentioned in Section 4.1. The code for our algorithm is submitted in the Supplementary Materials, along with description in Appendix A.3. For the LLMs mentioned in Section 4.1, any available inference provider for open source LLM inference can be used such as HuggingFace, AWS Bedrock, GCP Vertex, OpenRouter, etc.

A Appendix

A.1 Extended Related Works

This section extends the discussion presented in Section 2.

RADAR (Zhang et al., 2026b). RADAR conditions its denoiser module on the query and outputs a single graph for each query. It selects edges based on structural redundancy under a policy gradient tuned for accuracy. In our benchmarks, the underlying coordination structure is never revealed within the task query, aligned with the second scope criterion outlined in Section 2. A per-query generator has no extra information to condition on. Its design paradigm, which relies on offline reward training before episode execution, is already represented in our baselines by GPTSwarm and ARG-Designer.

AgentDropout (Wang et al., 2025b). AgentDropout learns node and edge masks for each round by applying gradient descent to task performance, treating token cost as a secondary objective. This approach shares the same supervision and per-round design as of G-Designer. The primary difference lies in the specific sparsity mechanism. AgentPrune, which is implemented in our benchmarks, already serves as a sparsity-regularized baseline within this same supervision design.

DyLAN (Liu et al., 2023). DyLAN is the closest existing work to a reward-free formulation, with key differences. Its Agent Importance Score relies on evaluation from a large language model rather than what the environment computes. While this removes the necessity for designing rewards, it still depends heavily on observing final outcomes. The evaluating model grades generated answers, meaning the feedback signal arrives after execution and reflects overall task success, like a LLM-as-a-judge. Two further distinctions are important for our claims. First, DyLAN selects overall agent participation at the node level, whereas our method predicts directed pairs between specific readers and sources. Second, DyLAN performs this selection during inference by scoring candidates, so it lacks a trained predictor that can generalize across unseen rounds. Its per-round pruning behavior is already captured in our evaluation by AgentPrune and G-Designer.

MacNet (Qian et al., 2025). MacNet examines collaboration by scaling across fixed topologies such as chains, trees, stars, and general graphs. Because these networks are defined by hand and involve no learning, there is no training signal to compare against. The performance bounds for such static networks are represented in Appendix A.6 by our three static baselines, which include Fully-connected, Isolated, and Random configurations.

Workflow and agent-system search (Zhang et al., 2025b; Zhou et al., 2026). These frameworks search over complete agentic programs by tuning prompt content, operator selection, control flow, and agent counts against an offline validation metric. The search target is the program itself, not the per-round pairs. Furthermore, this optimization requires an offline training stage on a validation traces set drawn from the target distribution. These frameworks complement our work rather than compete with it. We keep the underlying workflow constant across all methods so that topology remains the only variable.

Shared-context and asynchronous designs (Mao & Mirhoseini, 2026). These systems establish implicit connections by letting agents communicate through a shared memory space. No separate module for predicting network topology or explicit adjacency matrices, which leaves no direct

network structure to compare against our method. We believe shared memory is an alternative architecture for multi-agent coordination rather than a direct baseline to ours.

Curriculum-based comparisons (Madmoun & Lahlou, 2026). Madmoun & Lahlou (2026) show that providing direct communication channels often works better than shaping agent experience through a reward-driven curriculum. Their findings indicate that optimizing purely for experience can lead to unstable training and suboptimal behavior. While this finding supports our premise that communication structure is worth learning independently, their work changes whether a channel exists rather than predicting which channels to open, leaving no predictive topology to evaluate.

The argument against a stationary reward target (Trivedi et al., 2026; Wang et al., 2026). Trivedi et al. (2026) explain a key limitation of reward-based optimization, noting that optimizing against a reward proxy metric can alter the distribution that made the proxy useful. This issue compounds when multiple adaptive agents interact simultaneously, which we believe is pertinent in real world LLM-MAS. This compounding effect directly impacts methods like those in Table 1, which attempt to train a topology in calibration phase. In a study on this specific method family, Wang et al. (2026) show that continually adapting G-Designer or ARG-Designer to new tasks hurts performance on previously learned tasks. This drop in performance persists even when using experience replay, showing that simple optimizer forgetting is not the sole cause.

Reward-free communication in numeric-state MARL (Lo et al., 2024; Ding et al., 2023; 2024; Wang et al., 2023). These works establish the theoretical motivations behind our approach but cannot be run directly in our setting. They assume continuous, low-dimensional vector messages generated by jointly trained policies. This assumption makes content relevance a continuous distance in a learned latent space and allows the communication function to be fully differentiable alongside the agents. Furthermore, Wang et al. (2023) show that separating individual policies from explicit task metrics helps models adapt zero-shot across changing team structures. In our setup, agents are frozen language models that output free-form text. Joint training is not directly possible, so relevance is calculated directly between text strings. The cross-encoder based objective in Section 3.2 bridges this gap by framing topology prediction as a regression onto an external relevance score rather than an estimation of mutual information bounds.

These related works were excluded from evaluation due to incompatible design choices and scope boundaries. Four of these approaches, namely RADAR, AgentDropout, DyLAN, and workflow search frameworks, offer helpful points of comparison if the main goal is maximizing overall task performance. However, our focus is narrower. We aim to determine whether network topologies can be trained without relying on outcome feedback. To address this question effectively, our baseline set covers primary supervision classes across per-episode and per-round settings, policy gradient and supervised learning, scalar and graph-neural parameters, and static topologies that establish baseline boundaries.

A.2 Motivation for Contextual Encoding

Informal Proposition: If the usefulness of a communication edge depends on the information distribution across the team, then a predictor operating solely on pair-local features (e_r, e_s) is generally insufficient. A permutation-equivariant contextual encoder provides a function class capable of representing such team-conditioned edge decisions. Self-attention is used as a practical instantiation of the contextual encoder because it naturally constructs content-dependent interactions among all agents while preserving permutation equivariance.

A.3 Algorithm

Algorithm 1 provides complete algorithm of MERIT-MAS method, including topology prediction and training part.

Reader query template:

$$q_r = \mathrm{concatenate}(\text{task goal}, \text{reader workspace content retained in the D-slot ring buffer})$$

We use $\nu_{r,s} = \frac{1-\cos(e_r, e_s)}{2}$ for novelty computation.

Algorithm 1 MERIT-MAS Training loop and LIVEROUND, the inference path, executed identically irrespective of whether the training update is present.

Require: batch size B; frozen BE, CE; (w, β, θ); carried state (μ, κ)

procedure LIVEROUND(t) ▷ runs every round of every episode
  $e_r \leftarrow \mathrm{BE}\big(SW_r^{(t)}\big)$ for each agent $r = 1, \ldots, N$ ▷ own accessible content only
  $g^{(t)}, A_t \leftarrow$ Equations 1 and 2
  agent r reads $\{s : A_t[r, s]{=}1\}$; agents act and write $SW^{(t+1)}$
  return $g^{(t)}$ ▷ retained for the batch objective; unused at inference
end procedure

loop over batches $k = 1, 2, \ldots$ ▷ never terminates; no calibration phase
  $\mathcal{R} \leftarrow \varnothing$
  for each round t of each of the next B episodes **do**
    $g^{(t)} \leftarrow$ LIVEROUND(t) ▷ retained from the live call, not recomputed
    q_r= [goal] + [reader workspace state] ▷ while c_s is the candidate workspace content
    stage $(q_r, c_s, \nu_{r,s})$ in $\mathcal{R}$ ▷ Equation 3
  end for
  $\hat{m} \leftarrow \mathrm{CE}(\mathcal{R}) \odot \nu$ ▷ once per batch, off the topology prediction path
  if $\mathcal{P}_k = \varnothing$ **then**
    continue ▷ skip the step entirely; $\mathcal{L}_{\mathrm{sp}}$ is never applied alone
  end if
  update (μ_k, κ_k), **then** $y \leftarrow$ Equation 3
  $(w, \beta, \theta) \leftarrow \mathrm{ADAMW}\big((w, \beta, \theta), \nabla\mathcal{L}, \eta\big)$ ▷ $\mathcal{L}$: Equation 4
  checkpoint $(w, \beta, \theta), (\mu_k, \kappa_k)$
end loop

Inference: LIVEROUND(t) alone; the batch block is training only.
Invariant to: explicit episode's outcome or agent index.

A.4 CONFIGURATION SELECTION AND SENSITIVITY

Every value in Table 2 was fixed for evaluation runs, after sweeping on a small set of representative episodes, all five tasks, and three LLMs.

How each value was set. L, M and d_h use the smallest setting at which the network trains stably. γ=0.9 and B=4 set how fast the MERIT Score's running statistics and topology network parameters adapt. η=10^{-3} is the AdamW default.

Sparsity weight λ_{sp}: Table 5 sweeps λ_{sp}. Table 5 reports results using *Qwen3-235B-A22B-2507* (Yang et al., 2025) on *Hidden Teams* task for 50 episodes only, for faster sweep, since it is only needed for relative performance. Success is flat within overlapping Wilson intervals across the whole range. This acts as a regularizer for connection density, which also effects the overall episode token usage indirectly. The success is therefore not sensitive to this coefficient over any range one would plausibly choose. We use the reported λ_{sp}=0.1.

Table 5: **Sensitivity to the sparsity weight.** Success (%) with Wilson 95% intervals.

λ_{sp}	0.05	0.1 (used)	0.5	1.0	3.0
Success (%)	24.2	25.5	22.5	23.5	23.5
Wilson 95% CI	[13.6, 39.3]	[14.6, 40.7]	[12.3, 37.5]	[13.1, 38.6]	[13.1, 38.6]

Retention depth D: D bounds how many rounds of a source's published content a reader agent can still read. The component ablation in Table 3 varies it directly and finds it neutral (-2.0 points), which is what the construction predicts: with a per-round topology the reader–source pair is re-decided every round, so a deeper buffer mostly provides the same (stale) content the current topology would have used anyway. Our ablation shows that the interaction is small.

A.5 Choice of the Frozen Encoders

We use one open source off-the-shelf BE and CE from Section 4.1. Both are implementation choices based on technical feasibility rather than claims of their capabilities. During the comparative analysis of these models, we give the identical goal-conditioned text for testing, which are reported in Table 6.

Table 6: **Encoder comparison** for technical feasibility in our experiments, ran on three broad coverage tasks for MERIT-MAS method with Qwen3-235B-A22B-2507 LLM. We use mean of success (%) for reporting.

Role	Frozen encoder	Setting	Success (%)
Bi-encoder BE	`Qwen3-Embedding-0.6B` (Zhang et al., 2025c)	1024-dim	76.0
	`bge-large-en-v1.5` (Xiao et al., 2024)	1024-dim	73.4
	`e5-base-v2` (Wang et al., 2022)	768-dim	73.5
	`all-MiniLM-L6-v2` (Reimers & Gurevych, 2019)	384-dim	73.6
Cross-encoder CE	`jina-reranker-v3.5` (Nasika et al., 2026)	list-wise	76.0
	`Qwen/Qwen3-Reranker-0.6B` (Zhang et al., 2025c)	point-wise	68.6
	`nli-deberta-v3-small` He et al. (2021)	point-wise	66.3

Our empirical results indicate that while `Qwen3-Embedding-0.6B` yields the highest success over broad coverage tasks, the performance across standard off-the-shelf bi-encoders remains consistently competitive, demonstrating that MERIT-MAS is robust to the choice of embedding backbone.

For the cross-encoder, we select `jina-reranker-v3.5`. The purpose of the CE is primarily to serve as a weak semantic supervision source derived entirely from agent-generated content. This suggests that effective topology learning does not require highly accurate outcome-derived supervision and can emerge from weak semantic guidance. For a better task-relevant information transfer, one can fine-tune ranking models according to the task setup, which we leave for future explorations.

Since both the bi-encoder and cross-encoder are used completely frozen and without task-specific adaptation, the reported success should be interpreted as a conservative lower bound on supervision quality rather than an optimized relevance estimator. Additionally, list-wise ranking provides significant practical benefit, reducing overall computational cost by approximately $6.4\times$ compared to point-wise alternatives. Depending on environment constraints and computational budgets, these modules can be seamlessly replaced with alternative pretrained encoders.

A.6 Per-model Results

Figure 3 provides results with averages over the three LLMs. Tables 7, 8 and 9 provide the scores for each LLM separately, with their Wilson 95% intervals at the given tasks's own episode count. Wilson intervals are reported for descriptive uncertainty quantification only and are not used as formal equivalence tests. These tables are reported for the per-LLM results, not for independent LLM-wise comparisons. Every mean score in Figure 3 is the arithmetic mean of the scores from the three corresponding LLMs here.

A.7 Computational and Memory Complexity

We analyze the *live inference* which runs in every round at deployment, and the *training*, which runs once per batch of completed episodes. Below, L is the number of attention blocks, d the bi-encoder output width. N is the number of agents, T is total number of rounds, d_h is head dimension and B is batch size. Edge probabilities are computed by applying a sigmoid to each pair score and thresholding during topology decision in inference.

Table 7: **Qwen3-235B-A22B-2507** (Yang et al., 2025), success (%) with Wilson 95% intervals. Success (%) with Wilson 95% intervals at each task's episode count, identical for every method. Intervals are wide at these task-episode counts by construction. We are primarily interested in whether they overlap, and the ordering of the point estimates is not the primary metric of comparison for us.

Method	*Collab Meeting*	*Concordia Pasture*	*Marble Barg.*	*Hidden Teams*	*Coalition* N=12	*Coalition* N=18
MERIT-MAS (ours)	90.0 [69.9, 97.2]	66.0 [52.2, 77.6]	72.0 [62.5, 79.9]	24.3 [17.5, 32.7]	96.0 [86.5, 98.9]	74.0 [60.4, 84.1]
G-designer	95.0 [76.4, 99.1]	64.0 [50.1, 75.9]	72.0 [62.5, 79.9]	24.2 [17.4, 32.6]	96.0 [86.5, 98.9]	70.0 [56.2, 80.9]
AgentPrune	85.0 [64.0, 94.8]	68.0 [54.2, 79.2]	74.0 [64.6, 81.6]	22.5 [15.9, 30.8]	88.0 [76.2, 94.4]	82.0 [69.2, 90.2]
GPTSwarm	80.0 [58.4, 91.9]	52.0 [38.5, 65.2]	80.0 [71.1, 86.7]	14.2 [9.1, 21.6]	2.0 [0.4, 10.5]	0.0 [0.0, 7.1]
ARG-designer	75.0 [53.1, 88.8]	48.0 [34.8, 61.5]	81.0 [72.2, 87.5]	23.3 [16.6, 31.6]	12.0 [5.6, 23.8]	8.0 [3.2, 18.8]
Fully-Connected	90.0 [69.9, 97.2]	62.0 [48.2, 74.1]	89.0 [81.4, 93.7]	19.2 [13.2, 27.2]	100.0 [92.9, 100.0]	100.0 [92.9, 100.0]
Random	70.0 [48.1, 85.5]	66.0 [52.2, 77.6]	71.0 [61.5, 79.0]	24.2 [17.4, 32.6]	90.0 [78.6, 95.7]	76.0 [62.6, 85.7]
Isolated	70.0 [48.1, 85.5]	16.0 [8.3, 28.5]	49.0 [39.4, 58.7]	0.0 [0.0, 3.1]	0.0 [0.0, 7.1]	0.0 [0.0, 7.1]

Table 8: **OpenAI-GPT-OSS-20B** (Agarwal et al., 2025), success (%) with Wilson 95% intervals. Success (%) with Wilson 95% intervals at each task's episode count, identical for every method. Intervals are wide at these task-episode counts by construction. We are primarily interested in whether they overlap, and the ordering of the point estimates is not the primary metric of comparison for us.

Method	*Collab Meeting*	*Concordia Pasture*	*Marble Barg.*	*Hidden Teams*	*Coalition* N=12	*Coalition* N=18
MERIT-MAS (ours)	90.0 [69.9, 97.2]	76.0 [62.6, 85.7]	15.0 [9.3, 23.3]	14.3 [9.1, 21.7]	95.3 [85.6, 98.6]	57.2 [43.5, 69.9]
G-designer	80.0 [58.4, 91.9]	92.0 [81.2, 96.8]	10.0 [5.5, 17.4]	13.8 [8.7, 21.1]	92.4 [81.7, 97.1]	54.0 [40.4, 67.0]
AgentPrune	80.0 [58.4, 91.9]	64.0 [50.1, 75.9]	13.0 [7.8, 21.0]	12.7 [7.9, 19.8]	78.7 [65.5, 87.8]	69.6 [55.8, 80.6]
GPTSwarm	90.0 [69.9, 97.2]	76.0 [62.6, 85.7]	18.0 [11.7, 26.7]	7.7 [4.1, 13.9]	1.0 [0.1, 8.9]	0.0 [0.0, 7.1]
ARG-designer	75.0 [53.1, 88.8]	74.0 [60.4, 84.1]	17.0 [10.9, 25.5]	13.3 [8.4, 20.5]	6.4 [2.3, 16.8]	4.2 [1.2, 13.7]
Fully-Connected	80.0 [58.4, 91.9]	78.0 [64.8, 87.2]	16.0 [10.1, 24.4]	10.7 [6.3, 17.5]	100.0 [92.9, 100.0]	100.0 [92.9, 100.0]
Random	85.0 [64.0, 94.8]	88.0 [76.2, 94.4]	13.0 [7.8, 21.0]	13.8 [8.7, 21.1]	81.9 [69.1, 90.2]	61.4 [47.6, 73.6]
Isolated	80.0 [58.4, 91.9]	56.0 [42.3, 68.8]	4.0 [1.6, 9.8]	0.0 [0.0, 3.1]	0.0 [0.0, 7.1]	0.0 [0.0, 7.1]

Table 9: **Moonshot Kimi-K2-Thinking** (Team et al., 2025), success (%) with Wilson 95% intervals. Success (%) with Wilson 95% intervals at each task's episode count, identical for every method. Intervals are wide at these task-episode counts by construction. We are primarily interested in whether they overlap, and the ordering of the point estimates is not the primary metric of comparison for us.

Method	*Collab Meeting*	*Concordia Pasture*	*Marble Barg.*	*Hidden Teams*	*Coalition* N=12	*Coalition* N=18
MERIT-MAS (ours)	85.0 [64.0, 94.8]	14.0 [7.0, 26.2]	20.0 [13.3, 28.9]	4.6 [2.1, 10.0]	78.6 [65.4, 87.7]	37.9 [25.8, 51.8]
G-designer	70.0 [48.1, 85.5]	10.0 [4.3, 21.4]	17.0 [10.9, 25.5]	3.8 [1.6, 8.9]	74.7 [61.2, 84.7]	22.3 [13.0, 35.6]
AgentPrune	75.0 [53.1, 88.8]	6.0 [2.1, 16.2]	30.0 [21.9, 39.6]	3.4 [1.3, 8.3]	47.4 [34.2, 60.9]	35.9 [24.1, 49.8]
GPTSwarm	75.0 [53.1, 88.8]	4.0 [1.1, 13.5]	20.0 [13.3, 28.9]	2.0 [0.6, 6.4]	0.3 [0.0, 7.7]	0.0 [0.0, 7.1]
ARG-designer	55.0 [34.2, 74.2]	6.0 [2.1, 16.2]	25.0 [17.5, 34.3]	3.6 [1.5, 8.6]	1.6 [0.2, 9.9]	1.1 [0.1, 9.1]
Fully-Connected	70.0 [48.1, 85.5]	6.0 [2.1, 16.2]	18.0 [11.7, 26.7]	2.8 [1.0, 7.5]	100.0 [92.9, 100.0]	100.0 [92.9, 100.0]
Random	75.0 [53.1, 88.8]	8.0 [3.2, 18.8]	10.0 [5.5, 17.4]	3.8 [1.6, 8.9]	52.5 [39.0, 65.7]	28.0 [17.5, 41.7]
Isolated	55.0 [34.2, 74.2]	2.0 [0.4, 10.5]	10.0 [5.5, 17.4]	0.0 [0.0, 3.1]	0.0 [0.0, 7.1]	0.0 [0.0, 7.1]

Table 10: **Complexity of the topology network**, at N=18, T=6, L=2.

	GPTSwarm	*AgentPrune*	*G-designer*	*ARG-designer*	*MERIT-MAS*
Supervision	reward	reward	reward	reward	**none**
Trainable params	324	324	14,449	40,641	99,265
Independent of N?	✗	✗	✓	✓	✓
Inference	$O(N^2)$	$O(N^2)$	$O(N^2 d)$	$O(N^2 d)$	$O\big(Ndd_h + L(Nd_h^2 + N^2 d_h)\big)$
External training supervision queries / episode	$O(N^2)$	$O(N^2)$	$O(1)$	$O(1)$	$O(N^2 T)$
Memory (trainable)	$O(N^2)$	$O(N^2)$	$O(1)$	$O(1)$	$O(1)$

Inference: Our routing network cost has comparable asymptotic inference complexity to *ARG-designer* and *G-designer* (which use $O(N^2)$ *sequential* decoding steps, which are latency-bound), while using parameter sharing independent of N. Every method is generally dominated by its frozen encoder, making encoder choice the only material difference, which is orthogonal to reward-free signal. MERIT-MAS also has additional cost of CE and BE encoder calls, but they are negligible compared to LLMs token cost in our evaluation and setup. The synthesizer agent's tokens are included in all methods equally and therefore do not affect relative comparisons.

Training: The row "External training supervision queries / episode" counts accesses to the external training signal (reward for baselines, relevance estimation for MERIT-MAS), rather than the internal neural-network FLOPs of each method. *MERIT-MAS* performs NTB list-wise cross-encoder calls covering all $N(N-1)TB$ ordered pairs, the only quadratic term in the system. It is also the only cost with no latency budget, because the update runs de-coupled from live topology decision predictor (Section 3.1).

Memory: Our trainable state is $99{,}265$ parameters and carries no N-dependent term, since query/key/value weights are shared across agent slots with no explicit positional encoding. *Agent-Prune* and *GPTSwarm* instead store an $N \times N$ logit table, tying their learned state to agent count *and* agent ordering, so it cannot transfer directly when the team changes size.