Qeios

Peer Review

Review of: "AgentNet: Decentralized Evolutionary Coordination for LLM-based Multi-Agent Systems"

Kostas Skianis[1]

1. Department of Computer Science & Engineering, University of Ioannina, Greece

AgentNet is a novel framework for coordinating multiple Large Language Model (LLM) based agents without a central controller. The paper addresses shortcomings of prior multi-agent systems – notably their reliance on centralized orchestration and static role assignments – which can cause scalability bottlenecks, single points of failure, and hindered adaptability. By contrast, AgentNet proposes a fully decentralized approach: agents form a dynamically evolving network (structured as a Directed Acyclic Graph) where each agent autonomously routes tasks and learns from experience via Retrieval-Augmented Generation (RAG) memory. The authors evaluate AgentNet on diverse tasks (math problem solving, coding, and logical QA) and compare it against both single-agent strategies and other multi-agent frameworks.

The methodology of AgentNet is distinguished by its decentralized evolutionary coordination. Rather than predefining agent roles or using a master coordinator, each agent in AgentNet has two internal modules, a Router and an Executor, both powered by an LLM. The Router decides how to handle incoming tasks (whether to solve it, forward it to another agent, or split it into subtasks), while the Executor carries out the actual operations on tasks. This dual-role design means decision-making is fully distributed: every agent independently coordinates and delegates tasks based on its local knowledge, eliminating any single point of control. Such an architecture is a clear departure from earlier LLM-agent frameworks that relied on centralized controllers.

To enable evolutionary specialization, AgentNet equips each agent with RAG memory. Both the Router and Executor maintain fixed-size memory pools of past task “fragments” (records of the agent’s observations, context, and actions for steps it participated in). When a new task arrives, the agent retrieves relevant snippets from these memories using semantic similarity (via a trained embedding

 doi.org/10.32388/J10AFH

model) to inform its reasoning and decisions. This mechanism allows continual learning: the agent effectively does few-shot learning from its own past successful experiences, refining its skillset over time. Notably, the memory is managed dynamically – when it's full, less useful or outdated entries are pruned based on usage frequency, recency, and uniqueness. This ensures the agent's knowledge stays fresh and relevant, an approach that bolsters the soundness of the methodology by preventing memory saturation with stale data.

Key Strengths

- Decentralization & Scalability: By removing the central orchestrator, AgentNet avoids the bottlenecks and failure points of centralized systems. All agents make decisions in parallel, enabling the system to scale to larger numbers of agents and more complex tasks without a coordinating agent becoming overwhelmed. This coordination also improves robustness – if one agent fails or underperforms, others can reroute tasks accordingly, and the overall system continues functioning. The dynamic DAG topology inherently supports growth; as tasks evolve, new agents or connections can be added without redesigning a fixed workflow. This scalability was reflected in experiments where increasing the number of agents led to greater specialization and maintained performance gains.

- Adaptive Performance Gains: AgentNet demonstrated improved performance over baselines in the evaluated tasks. Experimental results showed that it achieved higher task accuracy and efficiency than both single-agent methods and other multi-agent frameworks. For example, on a math reasoning benchmark, AgentNet attained a significantly higher score (about 85.00) compared to a non-evolutionary multi-agent baseline (77.86), as shown in Table 3. These gains suggest that the evolutionary coordination (the "warm-up" phase of agents learning and specializing) indeed yields tangible benefits in problem-solving success. Agents in the network can pool their diverse expertise more effectively than a single large model or a centrally guided team, often arriving at solutions faster or with fewer errors.

- Emergent Specialization: A compelling strength of AgentNet is its ability to foster specialization among agents without predefining roles. Through the adaptive learning mechanism, each agent gradually focuses on tasks where it is most successful. The results showed that when multiple agents collaborate, they naturally diverge in expertise (e.g., some excelling at reasoning tasks while others handle language-intensive subtasks). This emergent specialization was observed to become more pronounced as the number of agents increased. Such specialization is beneficial for complex task-solving: agents effectively become experts in different domains or subtasks, which improves the collective ability to handle a wide range of challenges. AgentNet's roles evolve in a data-driven manner, based on actual performance. This

adaptive division of labor is a novel source of efficiency and one of the framework's conceptual innovations.

- Privacy and Cross-Organizational Collaboration: AgentNet's design inherently limits information sharing to only what's necessary for task completion, which the authors highlight as a strength for real-world deployments across organizational boundaries. Because there is no central server aggregating all data, each agent can keep proprietary data local, exposing only minimal metadata or partial results when routing tasks. This is crucial if, for instance, different companies' agents are cooperating on a project – they can collaborate to solve a problem without fully revealing their internal knowledge bases. Although this was not tested empirically in the paper, the privacy-preserving architecture is forward-looking. It increases the framework's real-world applicability in sensitive domains, where data governance and trust are as important as task performance. By enabling "federated" collaboration of LLM agents, AgentNet could unlock use cases where multiple stakeholders contribute expertise to a shared goal (e.g., research labs or departments solving a complex multi-faceted problem) while respecting data confidentiality.

- Novelty: AgentNet brings together concepts from evolutionary algorithms, distributed systems, and prompt-based learning in a unique way. The framework's natural evolution analogy allows the agent society to evolve over time (connections strengthen or die off, agents "learn" and mutate their behavior with experience). This is an innovative step beyond prior multi-agent paradigms. The incorporation of RAG-based memory for continual learning is a clever solution to the challenge of keeping LLM agents relevant over long deployments. It provides a form of long-term memory to agents without requiring external retraining of the underlying LLM for new knowledge. Moreover, the way AgentNet avoids propagating reasoning context when passing subtasks shows a nuanced understanding of system design, balancing information sharing with error containment. These design choices collectively make AgentNet a conceptually robust framework.

Weaknesses and Limitations

- Assumption of Homogeneity: In the current implementation and experiments, agents appear to be fairly homogeneous – likely using the same type of LLM backbone and similar capability representations. The framework's performance in a heterogeneous agent environment (where agents might have widely varying models, tools, or knowledge bases) remains unproven. Real-world multi-agent ecosystems could include agents of different sizes or specialties (e.g., one agent might be a code-specific model, another a vision model). How AgentNet handles such heterogeneity is an open question. The task routing strategy might struggle if an agent's capability vector is not directly comparable to another's (e.g.,

different modalities or scales), and training a robust similarity metric for capabilities could be challenging. This is a limitation because one promise of multi-agent systems is leveraging diversity; further or future work is needed to test and potentially adapt the framework for heterogeneous agent collectives.

- Routing Scalability & Knowledge of Agents: The decentralized routing mechanism in AgentNet may encounter difficulties as the number of agents grows very large. The Router in each agent currently selects among a relatively small candidate set of agents when delegating tasks. In experiments, the system was shown up to 5, 7, or 9 agents, which already exhibited good specialization. But in a scenario with hundreds of agents, having each agent evaluate all others to find the best next hop could be inefficient or infeasible. Structures like clustering may be useful at a large scale. Additionally, the Router's decision process might get stuck in local optima – if it only knows about a fixed neighborhood of connections, it might miss sending a task to a far-away agent that is actually the ideal expert. The authors note this and suggest improving the routing algorithms, possibly by introducing exploration incentives so agents can discover new peers beyond their initial connections. Until such enhancements are made, the routing approach may not seamlessly scale to very large networks or highly dynamic membership (agents joining/leaving frequently).

- Experiments: The evaluation, while spanning math, coding, and logical Q&A domains, is still limited to text-based problem-solving tasks. All tasks essentially reduce to prompting LLM agents with well-defined inputs and checking their outputs for correctness. This leaves out entire classes of multi-agent coordination problems – for instance, continuous control tasks, real-time interactive scenarios, or physical-world tasks (multi-robot planning, game environments, etc.). The framework assumes a task can be decomposed and solved via LLM reasoning steps, which may not hold for other tasks. Moreover, the tasks used are benchmarks that likely play to the strengths of LLMs and the proposed system. We do not see experiments on planning tasks or mixed-modality problems. This raises the possibility that AgentNet might be overfitted to the evaluated task types – i.e., its impressive gains could diminish on tasks that don't involve pure text or that require knowledge the agents didn't see in their RAG memories. Last, the authors do not state clearly if they conducted simple or multiple runs, showing the stability of the method and if agents continue to learn similarly.

- Complexity, Overhead, and Total Cost: While decentralization brings benefits, it also introduces system complexity. AgentNet requires maintaining a network of agents that communicate and pass tasks around, which could be resource-intensive. Each agent runs its own LLM-based reasoning for routing

and for execution; this could multiply the computational cost compared to a single-agent solution. For example, solving a task might involve multiple agents each invoking an LLM in sequence or in parallel, whereas a single large model could perhaps solve it in one go (albeit less efficiently or accurately, in some cases). The paper does not deeply discuss the computational overhead or latency implications of the approach. In real deployments, issues like network communication cost, concurrency control, or how to handle an agent that fails mid-task would need consideration. The current implementation assumes a reliable environment where agents always eventually produce an answer or hand off a task. This might be a simplifying assumption that glosses over practical engineering challenges in decentralized systems. Reproducibility can also suffer due to this complexity – one must orchestrate multiple LLM instances and track their evolving states, which is non-trivial. Thus, from an engineering perspective, AgentNet trades a single complex agent for a complex multi-agent setup, which might limit its immediate practicality if not packaged with robust tooling. Finally, combining multiple agents may introduce significant costs by the multiple calls to LLMs (open or closed-source).

- Privacy Claims: A motivating feature of AgentNet is privacy preservation (agents share minimal information). However, there isn't a specific experiment measuring data exposure. In practice, if agents are truly in different organizations, issues like network security, trust, and standardization of communication protocols would come into play. AgentNet assumes all agents cooperate honestly and follow the protocol (since it was tested in a controlled setting). In a real cross-org scenario, there could be incentives to cheat or to guard information beyond what the DAG communication enforces. The framework would need additional mechanisms (encryption, verification of results, etc.) to be viable for sensitive collaborations. Therefore, while conceptually AgentNet is privacy-preserving by design, this remains to be demonstrated. It's a forward-looking strength, but also a current limitation that no quantitative privacy or security assessment is provided.

Suggestions

- The authors could improve the Related Work section by comparing more thoroughly with previous work.
- Consider experimenting with more diverse tasks
- Complexity analysis: show the expected routing cost per task (in terms of avg degree and hop limit).
- Improve readability by decreasing font size in Figures 6, 7 and increasing font size in Figures 9, 10

Declarations

Potential competing interests: No potential competing interests to declare.