Matheo Xenakis

8 August 2024

Project Proposal: Optimization of Multi Agent LLM Networks

Multi-agent systems of large language models (LLMs) have seen a spike in interest in recent years owing to their ability to complement some of the weaknesses of individual LLM agents and produce solutions to a wide variety of problems in different contexts (such as games, data analysis, and responses to prompting). From encoding text-based environments so as to allow LLM agents to readily interpret their environment, to encoding natural-language feedback from LLM's to simulate rewards in a complex environment in the style of reinforcement learning, to using classic reinforcement learning algorithms to optimize a network of communicating LLM agents by removing antagonistic nodes, experimenting with different multi-agent LLM architectures at both the agent and network level has led to a burgeoning field of practical research that leverages the natural language capabilities of LLMs for wider use cases.

One specific area of interest for LLM research is in dynamic environments, like games. It is practical to use NLP in games, because it provides a very natural way of providing feedback to the learners/agents in a game setting, as well as a nuanced medium by which different learners/agents in a game environment can communicate. Recent research has focused on developing robust methods for grounding LLM agents in their environment. An early approach (2023) uses a multi-agent approach called "GLAM", which uses functional grounding to create 'alignment' between an LLM's understanding of its environment & its true state, where LLM agents are used to describe the environment, and an encoder uses the description as a prompt to generate the softmax policy for the agents action space [1]. The policy is refined using PPO. However, a more recent (2023) paper minimizes encoding by allowing agents to learn using "verbal reinforcement learning", where agents learn from feedback signals ("wow, this is great!") and use episodic memory buffers to induce better decision making in subsequent trials [2].

While significant research has focused on cooperative or competitive multi-agent learning with a small number of learners/agents in each game, we believe that the progress made towards encoding environments, along with the demonstrated ability to support learning with a larger number of agents [3], sets the stage well for evolutionary or population-based approaches to learning complex tasks or game. That is, rather than training a single agent to complete a simple task, we train a population of agents to achieve a more complex or dynamic task, where the agents themselves are acting solely from their own self interest. A very simple example of a population based approach to learning is Darwinian evolution, where the genotypes that are selected are the ones that are consistent with the dynamics of the larger ecosystem. In general, evolutionary or population based approaches to learning have found uses in epidemiology, the social sciences, economics, and artificial life due to their ability to model complex or evolving phenomena.

The proposed project will focus on selecting, recreating, or engineering an existing population-based reinforcement learning testbed(s), particularly a multi-agent particle environment such as the ones available in OpenAI's multiagent particle environments [4] or agar.io, in order to compare and contrast the effectiveness of three different kinds of learners: Random learners, learners with simple neural networks, LLM learners that use the Reflexion paradigm using only natural language, and learners that contain LLM's that can read from the outputs of neural networks. The actual metrics of success will vary greatly between environments. The purpose of the project is to experiment with new approaches for population based reinforcement learning, which has seen an increase in popularity recently, to identify environments where LLM agents are highly likely to be successful, and if luck should allow, to identify/ design a useful framework for population based reinforcement learning using multiple LLM agents– as an example, this could be designing agents such that are able to communicate in order to coordinate a strategy to avoid spreading a pandemic– despite that they may have a strong desire to interact. The end deliverable will be a research paper summarizing the findings of the experimentation.

Due to the fact that this project is highly environment dependent, and there are not a lot of LLM agent testbeds available out of the box, a substantial portion of the time devoted will be researching and designing the population-based testbed. The first phase of the project will be to extensively research the testbeds available for use or adaptation in a comprehensive literature review, and identify a single testbed (or set of testbeds) that can be built for our experimentation. A complete deliverable will be a design at the level of pseudocode accompanied with a detailed requirements document.

Next, the testbeds will be created in Python using standard libraries where possible, as well as any libraries for the LLM implementation (The selection of which will be a part of the previous step). The environments will be built out, debugged, and run. Although the details will be left to be decided during phase one of the project, the experiments to run will be run, data will be gathered, and an initial draft of the paper will be created.

In the final phase of the project, the report will be edited, key graphs will be beautified, and the papers will be submitted to the appropriate journal(s).

Timeline:

August - October 15: Research, Design, and Select the testbeds/problems to solve. Deliverable: Detailed requirements for the environment/ agent classes.

October 15 ‘24- March ~15 ‘25: Create the testbeds, code the agents, run experiments and gather results. Deliverables: Working testbed, First draft of paper

March ~15 2025 - Late April 2025: Edit paper, finalize experiments, beautify graphics. Deliverable: Final draft of paper

Late April 2025: Submit papers to journals

Bibliography

1. Carta, Thomas, et al. "Grounding large language models in interactive environments with online reinforcement learning." *International Conference on Machine Learning*. PMLR, 2023.https://arxiv.org/abs/2302.02662
2. Shinn, Noah, et al. "Reflexion: Language agents with verbal reinforcement learning." *Advances in Neural Information Processing Systems* 36 (2024).https://arxiv.org/abs/2303.11366
3. Zhuge, Mingchen, et al. "Language agents as optimizable graphs." *arXiv preprint arXiv:2402.16823* (2024). https://arxiv.org/abs/2402.16823
4. OpenAI MultiAgent Particle Environments https://github.com/openai/multiagent-particle-envs
5. Joseph Suarz, Github webpage describing MMO learning https://jsuarez5341.github.io/