LLMs for sample efficiency in multi-agent reinforcement learning

Hannah Bansal
Computing and Information Systems
University of Melbourne
Melbourne, Australia
hannah.bansal@student.unimelb.edu.au

Abstract—Multi-agent reinforcement learning (MARL) faces challenges in achieving good accuracy in a sample-efficient manner due to the complexity of the environment and the difficulty in aligning individual agent objectives. "LLM4RL" is a body of research where Large Language Models (LLMs) assist in training RL models. In this paradigm, LLMs have shown remarkable potential in enhancing the sample efficiency of Reinforcement learning systems. This study proposes the integration of large language models (LLMs) as policy priors to enhance agent cooperation and improve sample efficiency in MARL. By harnessing the general reasoning capabilities and extensive real-world knowledge base of LLMs, the proposed approach aims to accelerate the training process and generate robust, context-aware policies. LLMs can provide precise policy motivations aligned with the desired outcomes of the agents, guiding them towards rewarding states more effectively in sparse reward settings. We also suggest a use for multi-agent debate (MAD) to align agent collaboration in cooperative tasks and improve decision-making skills. The relevance of these attributes is particularly pronounced in MARL environments, where agents must make predictions about appropriate actions. This work aims to validate the performance of single-agent systems and provide a framework for using LLMs as policy priors in MARL systems.

Keywords—**MARL, LLM, POLICY DISTILLATION**

I. INTRODUCTION

Multi-agent reinforcement learning (MARL) studies the interaction of multiple agents in a common environment. These can be cooperative, competitive or collaborative (mixed). In a cooperative environment, agents work to optimise long-term reward. Recent major breakthroughs in general intelligence include DeepMind AlphaStar surpassing top performance in StarCraft II [1] and Open AI Five [2]. However, training such systems requires huge, parallelised resources due to most useful environment cases having a sparse reward and sample-inefficient setting for training [3]. This means that they need a high number of interactions with the environment to learn an effective policy. Several approaches have been suggested to improve sample efficiency in MARL. Some of these include: Curriculum learning [4], Policy distillation [5], and Hierarchical learning [6].

In this study, we consider a relatively recent body of research that suggests using an LLM to improve sample efficiency in RL systems.

This work proposes a uniting framework for policy distillation from an LLM in multi-agent reinforcement learning. We aim to show that multi-agent debate can be an effective tool to ensemble LLM responses from distinct agents in such a design. Additionally, we also aim to verify the usefulness of the single-agent policy distillation method.

LLMs as policy priors

Large language models (LLMs) have emerged as profoundly influential tools in recent times [8][23]. Predicated on the concept of conditional probability models, modern LLMs are primely built on the transformer architecture [7]. Pre-trained LLMs are adept at token-wise prediction, using preceding input to forecast subsequent tokens. Owing to a vast parameter space due to training on large text corpora [8], they exhibit proficiency in a myriad of other natural language processing tasks, including but not limited to text generation, machine translation, sentiment analysis, summarization, question answering, named entity recognition, part-of-speech tagging, dialogue management, and semantic text similarity [8][9].

LLMs possess the ability to generate human-aligned outputs with minimal training data through few-shot learning [8]. This capability is especially advantageous in MARL contexts, where providing contextually appropriate guidance can significantly influence agent behaviour and policy development. By leveraging their understanding of natural language and contextual cues, LLMs can formulate precise policy motivations that align with the desired outcomes of the agents. This alignment not only accelerates the training process but also ensures that the policies generated are robust and well-suited to the specific environment and tasks at hand.

Due to their general reasoning capabilities, there has been substantial research into considering LLMs as "embodied agents", i.e. agents grounded in a real-world environment that is working to achieve a task [9][10][11]. Another advantage of using LLMs as embodied agents is their extensive real-world knowledge base. LLMs, trained on vast and diverse datasets, can draw upon this knowledge to explore new and potentially advantageous trajectories within the environment. This capability is particularly beneficial in sparse reward settings, where traditional RL agents might struggle with costly and inefficient exploration. By intelligently guiding the agents based on the rich contextual information available from their observation space, LLMs can help navigate towards rewarding states more effectively. This judicious use of observational data not only enhances the efficiency of the exploration process but also reduces the overall computational resources required.

This is where "LLM4RL" research comes in which can be described as models where an LLM assists in the training of an RL model that performs a task that is not inherently related to language. There are 3 main categories [9] in which a LLM4RL system may be formulated:

1. **Reward**: LLM is used to design the reward function [10]
2. **Goal**: LLM is used to determine an internal goal for curriculum learning

3. **Policy**: Pretraining, representing or updating the policy function of the agent [11][12][18]

While LLMs demonstrate broad reasoning abilities, they are not specifically trained to tackle environment-dependent problems and cannot thus directly interact with or adapt to the specific contexts where tasks occur. To address this gap between the LLMs' generalized statistical knowledge and specific task environments, Carta et al. [12] introduced a method for functionally grounding LLMs. In their approach, LLMs are used as direct policies that can be updated, within a structured framework.

In their study, [12] use a basic grid-world text-based environment called MiniGrid [14]. They designed a goal-augmented, partially observable Markov Decision Process (MDP), in which the LLM, given a prompt, generates a probability distribution over potential action. The prompt is generated by giving a description of the environment and the current observation state which is converted to a textual format. An action is then chosen based on this distribution. They use a Flan-T5 780M model, which they fine-tuned using Proximal Policy Optimization (PPO) [13]. After 250,000 training steps, this setup achieved an 80% success rate—a notable improvement over previous models.

Multi-Agent Debate (MAD)

The multi-agent debate is a paradigm in which agents engage in a structured exchange of responses to refine and enhance the accuracy, truthfulness, and reliability of textual answers. In this model, multiple agents are presented with the same initial prompt, and each generates an independent response. Subsequently, these responses are shared among the agents in a structured "debate" round, during which each agent revaluates its initial response by incorporating the perspectives provided by others. This iterative process aims to converge on a more refined answer, effectively employing a form of ensemble technique to harness the collective intelligence of multiple LLM agents.

The efficacy of this approach is substantiated by various studies, such as those documented in [15], in which authors use different prompting strategies for Q&A. In the seminal work "Society of Minds" [16], the authors observed notable enhancements in accuracy across diverse benchmarks, including the Massive Multitask Language Understanding (MMLU), the Grade School Math (GSM8K) dataset, and assessments of chess move optimality. These benchmarks are indicative of the model's capabilities in general reasoning, mathematical problem-solving, and strategic planning.

The relevance of these attributes is hypothesized to be particularly pronounced in the context of multi-agent reinforcement learning environments, where agents are required to make predictions about appropriate actions based on their observations of a dynamic environment. The multi-agent debate paradigm offers a compelling framework for improving the decision-making processes of such agents[15]. By leveraging a diverse array of responses that are iteratively refined through debate, agents can achieve a more comprehensive understanding of the task at hand, potentially leading to more sophisticated action strategies. This method not only amplifies the individual capabilities of each agent but also significantly enhances collective problem-solving effectiveness, which is crucial for multi-agent systems.

II. METHODOLOGY

In this work, we primely consider a multi-agent variation of MiniGrid as our exemplar for the problem description namely MultiGrid. This environment is represented by a 2-dimensional grid world with attributes such as keys, doors, boxes and goals. There are several predefined tasks with cooperative rewards. For this study, we consider 2 procedurally generated scenarios [Fig 1].

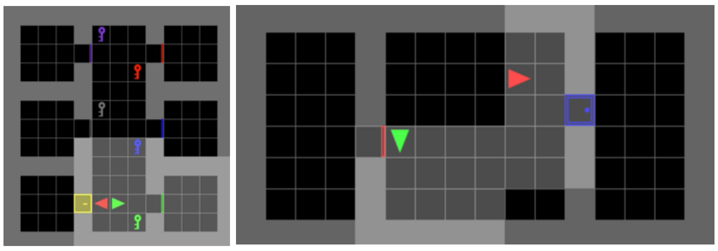
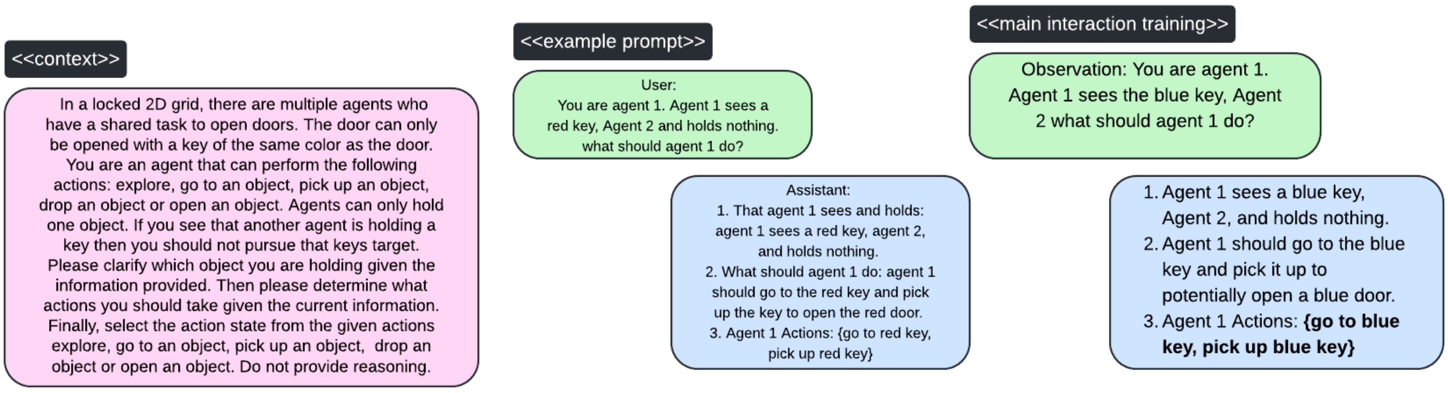


Fig 1: [Right] LockedHallway: the agents must (a) find a key and (b) go to the corresponding door to unlock it. [Left] RedBlueDoor: the agents must a) open the red door b) and then c) open the blue door in that order.

Problem Description

Within this work we consider a sequential decision-making scenario that is formalised as a cooperative, partially observable Markov Game. This setting is a sequential decision-making problem also formalized as a cooperative partially observable Markov game; an extension of a Markov decision process like the work in [4][17]. A Markov game for N agents is defined by a

- Tuple $\langle S, A, O, R, P, n, \gamma \rangle$ where S is the State Space.
- A is the shared action space for each agent *i*.
- $o_i = O(s; i)$ is the local observation for agent *i* at global state s.



<<context>>

In a locked 2D grid, there are multiple agents who have a shared task to open doors. The door can only be opened with a key of the same color as the door. You are an agent that can perform the following actions: explore, go to an object, pick up an object, drop an object or open an object. Agents can only hold one object. If you see that another agent is holding a key then you should not pursue that keys target. Please clarify which object you are holding given the information provided. Then please determine what actions you should take given the current information. Finally, select the action state from the given actions explore, go to an object, pick up an object, drop an object or open an object. Do not provide reasoning.

<<example prompt>>

User:
You are agent 1. Agent 1 sees a red key, Agent 2 and holds nothing. what should agent 1 do?

Assistant:
1. That agent 1 sees and holds: agent 1 sees a red key, agent 2, and holds nothing.
2. What should agent 1 do: agent 1 should go to the red key and pick up the key to open the red door.
3. Agent 1 Actions: {go to red key, pick up red key}

<<main interaction training>>

Observation: You are agent 1. Agent 1 sees the blue key, Agent 2 what should agent 1 do?

1. Agent 1 sees a blue key, Agent 2, and holds nothing.
2. Agent 1 should go to the blue key and pick it up to potentially open a blue door.
3. Agent 1 Actions: **{go to blue key, pick up blue key}**

Fig 2. An example of the prompt context and example that is provided to LLM for making one shot prediction of action

- *P (s' |s, A)* denotes the transition probability from *s* to *s'* given the joint action $A = (a_1, a_2 \ldots, a_n)$ for all n agents
- *R(s, A)* denotes the shared reward function. γ is the discount factor.
- Agents use a policy $\pi_\theta(a_i|o_i)$ parameterized by θ to produce an action a_i from the local observation o_i
- Observations are given by function $o_i : \mapsto S \;\; O_i$.

Within this environment, we set that all agents share the same policy, and receive the same reward. Additionally, we consider a finite horizon for the time length of an episode bounded by T. This horizon is considered in case our agents end up in stagnation loops of executing the same action, or if the learning plateaus. In our cooperative setting, the goal is to jointly optimise the accumulated reward given by:

$$\max(J(\theta)), where \quad J(\theta) = E_{A^t, s^t}[\textstyle\sum_t^T \gamma^t R(s^t, A^t)]$$

those actions are directly utilized. Otherwise, we query it again until a response with valid actions is provided.

The key change from this and previously referenced work is that we are running the routine for multiple agents concurrently. After generating actions, these are input into our multi-agent debate (MAD) module. In this phase:

1. We collect actions from all the agents.
2. Then is prefaced with the same observation and "Agent *i* perform action X" for all *i* except for the current agent's selected action
3. Then this is passed through the LLM again to generate a new set of actions.

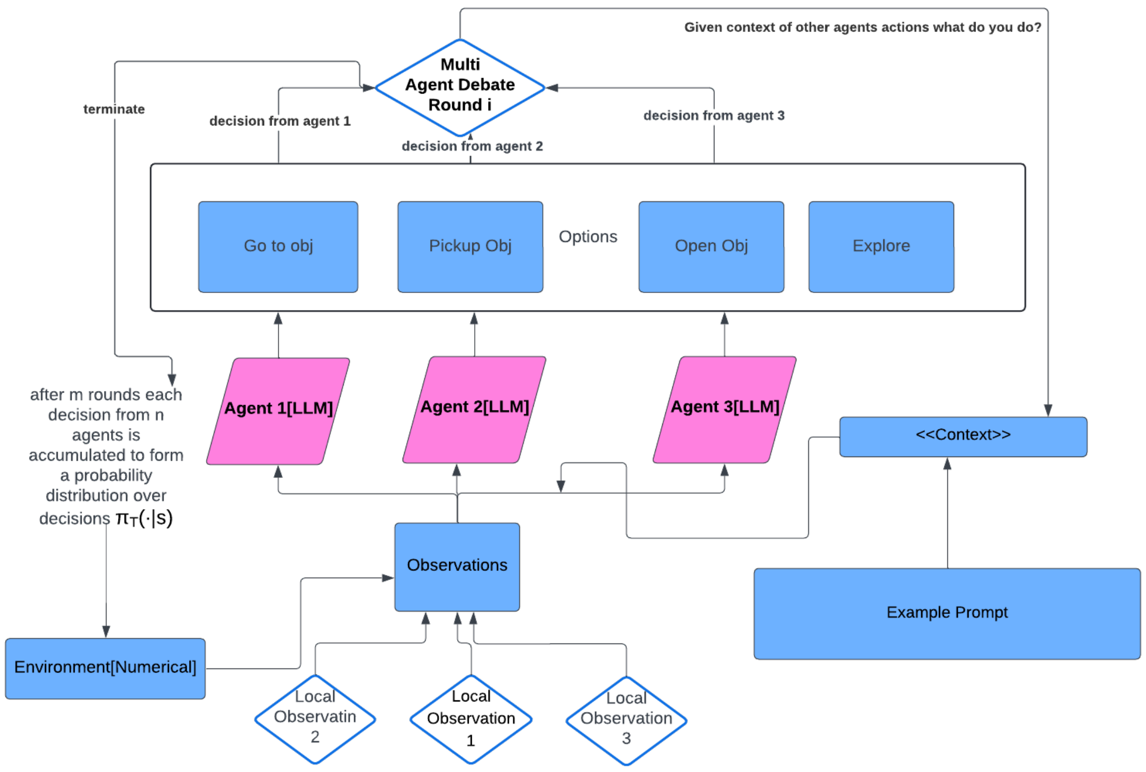


Fig 3. A high level overview of the proposed multi-agent RL policy distillation framework

Constructing policy prior using an LLM in a multi-agent setting

To operationalize LLM agents within a multi-agent framework, we devise a textual representation of each agent's observation space, which is constrained to a limited grid surrounding the agent. This representation also incorporates a defined set of potential actions available to the agent. For the MiniGrid/MultiGrid, we define these to be "*explore, goToObject, pickup, toggle, drop, wait*" derived from [18]. A consistent prompt template is used for a specific test scenario within an environment, with modifications introduced only to reflect changes in the observation space. This uses a chain of thought (CoT) style prompt [Fig 2] with 3 steps to help the understand the case better. [22]

For each prompt, each agent's LLM generates generates a textual response. If this response includes actions from our predefined list,

We then form a probability distribution per agent which forms our shared teacher policy $\pi_T(\cdot|s)$ given by the expression:

$$\pi_T(\cdot \mid s) = \sum_i \sum_k Pr_{\text{LLM}}(k \mid c(s))\, \pi_{k_i}(\cdot \mid s)$$

Here $Pr_{\text{LLM}}(k \mid c(s))$ is the probability of an agent choosing the k^{th} option given a textual representation *c(s)* (subject to changes from the iterative MAD process). $\pi_{k_i}(\cdot \mid s)$ is the associated policy with the k^{th} option and agent i. These are then summed over all the agent distributions to give $\pi_T(\cdot|s)$. This process has been described in Fig 3. This iterative debate stage is hypothesized to confer multiple benefits, particularly within a multi-agent system:

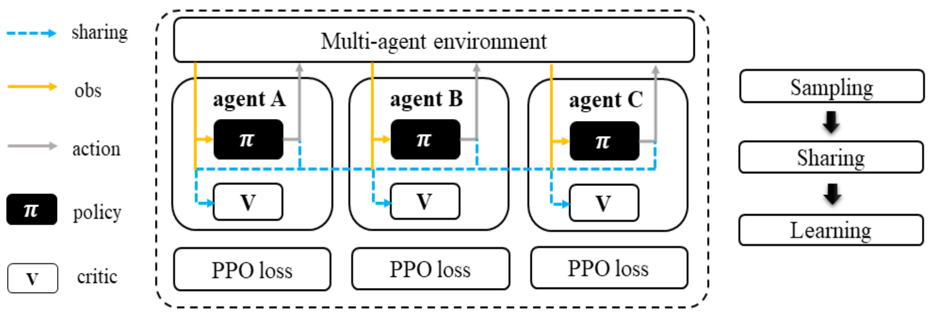


Fig 4. MAPPO works by using a shared centralized critic value function and agents to share information, including observations and predicted action during the sampling stage. They then follow their own PPO pipelines based on specific local observations.

Conflict Resolution: In scenarios where agents may inadvertently choose conflicting objectives—such as simultaneously attempting to retrieve a single key usable by only one agent at a time—the integration of other agents' actions into the decision-making context allows for enhanced strategic deliberation. By receiving textual information about the actions of their peers, agents can make informed decisions that avoid conflict and optimize cooperative strategies. This approach leverages the LLM's capacity for processing complex scenarios, thereby facilitating more effective and harmonious interactions among the agents.

Multiple rounds improve reasoning: As it has been shown that multiple rounds of MAD lead to better decision quality [16], we have hardcoded our agents in such a fashion to enhance the sample efficiency of the system.

Policy Distillation to multi-agent PPO (MAPPO)

In cooperative MARL, it is useful for the agent to take in extra global information that is not included in the agent's local observations. To this end, the MAPPO algorithm [17] uses a Centralized value function for all the agents to give them global information to calculate the Generalized Advantage Estimation after which a training procedure like a single agent PPO training is followed by each of the agents [Fig 4].

The shared policy for the student agents denoted by $\pi_\theta(.\mid s)$ is learned by minimizing the combined loss function of the standard RL loss function $\mathcal{L}_{\mathrm{RL}}(\theta)$ and teacher guidance which is produced to update the policy function in the MAPPO training procedure. This loss is given by:

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{RL}}(\theta) + \lambda\, \mathrm{E}_{s\sim\pi_\theta}\, \mathcal{H}\big(\pi_T(\,.\mid s)\mid \pi_\theta(.\mid s)\big)$$

Here Learning policy for the RL algorithm is the policy function per agent defined by $\mathcal{L}_{\mathrm{RL}}(\theta) = L(s, a, \theta_k, \theta)$:

$$\min\big(dA^{\pi_{\theta_k}}(o,s,\boldsymbol{u}^-),\ clip(\mathrm{d}, 1-\epsilon, 1+\epsilon)\, A^{\pi_{\theta_k}}(o,s,\boldsymbol{u}^-)\big),$$

$where\ d = \frac{\pi_\theta(u|O)}{\pi_{\theta_k}(u|O)}$ which is ratio between the old and the new policy.

Here the clip function ensures that the policy only gets updated to some degree controlled by hyperparameter ϵ. $A^{\pi_{\theta_k}}$ is the general advantage estimation at step k. For more details check the MAPPO and PPO paper for implementation details [3][13].

The teacher's guidance of what actions to take is incorporated into the student policy by introducing the regularization term $\mathcal{H}\big(\pi_T(\cdot\mid s)\mid \pi_\theta(\cdot\mid s)\big)$ which is a mathematical formulation for the distance between the student policy $\pi_\theta(\cdot\mid s)$ which can be shared across students and the teacher policy $\pi_T(.\mid s)$. This is the KL divergence which describes how a probability distribution is different from another reference distribution. An annealing term λ is used to demonstrate dependency on the teacher policy. During the initial stages, λ is high, noting that the teacher guidance is given more attention. As training progresses, this term is linearly decayed to 0 so that the student policy can focus on improving its loss function independently.

III. Experiments and Results

Presently, our experiments for the LLM prompting framework were limited to a single-agent setting due to implementation constraints. At this stage, we have made the LLM querying and multi-agent debate framework but they still need to be integrated with the MAPPO algorithm as described in the Methodology section.

Benchmark results for the multi-agent environment using pure reinforcement learning algorithms have been produced. Comparative analysis with the suggested algorithm to test its transferability from a single agent to a multi-agent setting is planned for future work.

First we compare the performance between the baseline i.e. a pure PPO implementation and then the LLM policy distilled student agent on the SimpleDoorKey and LavaDoorKey scenarios [18] which are similar to the ones described in the methodology section but for single-agent systems. For this study, we are concerned with 2 key factors and associated metrics as defined below.

1. **Sample efficiency**: we consider a normalised Area under the learning curve metric for comparing performance. This is defined by:

$$\mathrm{AULC}_{\mathrm{actual}} \approx \sum_{i=1}^{n-1} \frac{(x_{i+1}-x_i)\cdot(y_i+y_{i+1})}{2}$$

$$\mathrm{AULC}_{\mathrm{max}} = \sum_{i=0}^{n-1} \max\_reward\ \cdot (y_{i+1}-y_i)$$

$$\text{Normalized AULC } = \frac{\mathrm{AULC}_{\mathrm{actual}}}{\mathrm{AULC}_{\mathrm{max}}}$$

Where (x_i, y_i) represents the learning curve in terms of the training steps and performance measures respectively.

2. **Accuracy**: we consider success rate and mean reward during the evaluation stage as well as cumulative reward during the training phase.

Other than these we are also concerned with visual inspection of the plots to see which training curve reaches an inflection point faster.

In this study, we use the Vicuna 13B model [19] with 5-bit quantisation locally for our experiments. We achieve this by compressing the model with llama-cpp [20], which provides a

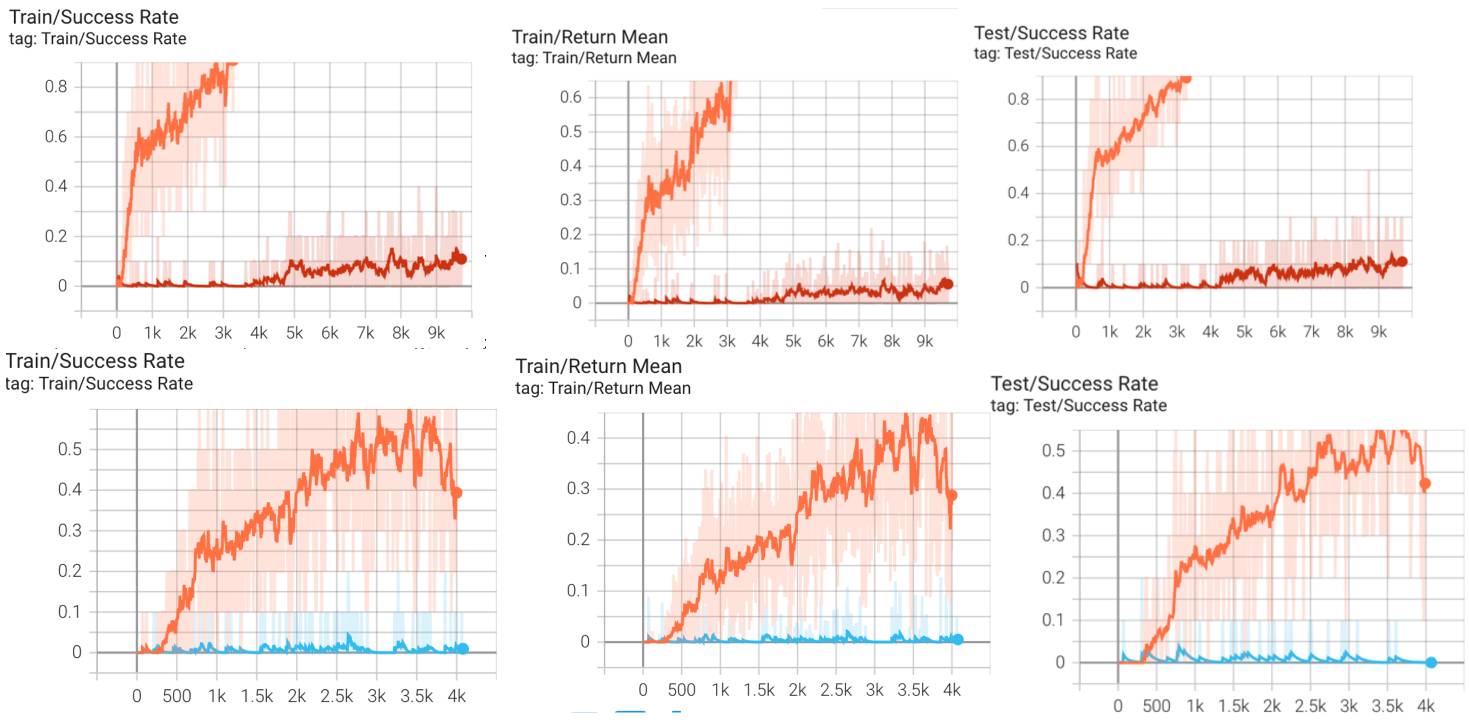


Figure 5. Top: Results on MiniGrid case of SimpleDoorKey. Orange line is the PPO agent while red line is LLM4teach agent. Bottom: Results on case of LavaDoorKey. Orange line is PPO, light blue is LLM4teach

lightweight implementation and allows us to perform inference on an Apple M3 Pro chip. This cuts the need for API calls which require the setup of a dedicated server and has similar performance in terms of response quality to some of the experiments seen from related research.

On SimpleDoorKey scenario, we get the following training curve for 10k training iterations using baseline PPO, and 3.2k steps using the LLM4Teach agent. On the BabyAI code repository [14] we see that the authors use a training step of magnitude ~100k to get comparable performance on a similar environment of DoorKey. The difference in sample efficiency is also seen by comparing the AULC scores which are detailed in Table 1.

For LavaDoorKey, TwoDoor we follow a similar procedure for the LLM4Teach agent training of 4k steps and notice an even worse performance for the pure RL counterpart due to more environmental complexity such as obstacles (lava) and multi-objectives. [Fig 5]

	AULC (training)	AULC normal	Mean Reward (eval)	Success Rate (eval)
	SimpleDoorKey			
PPO	197.6852	0.0931	0.0646	0.1
PPO+LLM Teacher	**1351.8870**	**0.4748**	**0.7654**	**1.0**
	LavaDoorKey			
PPO	14.8301	0.0285	0.0102	0
PPO+LLM Teacher	**925.1988**	**0.3367**	**0.3226**	**0.4**

Table 1: Comparison of results for sample efficiency and accuracy.

IV. Discussion

We describe a Multi-agent policy distillation system that can be used to improve sample efficiency in coordination-based corporative tasks as suggested by its implementation in the MultiGrid environment. LLMs reduce the need for RL models to be trained from scratch by enabling the transfer of pre-trained policies. This transferability is seen to be a major advantage in MARL, where training from the ground up can be prohibitively time-consuming and resource-intensive. With relation to our results, it was observed from the literature that it takes in the order of a 100k training steps to get an environment like SimpleDoorKey compared to about 4k in our experiments. Using the described normal AULC metric we can see this this discrepancy in sample efficiency reflected numerically as well.

Additionally, we see that policy transfer in a single-agent system leads to a much lower model size compared to the initial LLM which is in the order of 2*10k for the student agent and 13B for the LLM.

Pre-trained LLMs, with their advanced reasoning and understanding capabilities, can provide a foundation upon which further refinements can be built. This not only saves considerable training time but also allows for the deployment of effective policies in new and varied environments. The use of LLMs in this capacity ensures that the agents can adapt and perform well, even in complex and dynamic scenarios.

Applicability and further work

Currently, the multi-agent framework being described is still in development stages. The LLM policy generation system along with the multi-agent debating module has been implemented separately and still needs integration with the MAPPO algorithm.

In this work, we only considered a framework for policy shaping through an LLM teacher but future studies for MARL in this space could approach other avenues such as reward shaping or goal generation as has been done for single-agent systems. A comparative analysis between these techniques and seeing whether the other fits with MARL would be useful to study.

After a review of the literature, a potential gap in real-world applicability can also be identified since most approaches have only been tested on simulated visual environments such as MiniGrid and Minecraft [21], or in text environments. However, a potential reason for this is that most current research is looking at novel methods to bring LLMs and RL methods in conjunction. We expect real-world applications in sectors such as robotics, healthcare and finance soon [9].

Conclusion

In this research paper, we introduced a novel design for a policy distillation framework for multi-agent reinforcement learning from LLMs and validated the results based on single-agent systems with measures such as AULC. In the future, we would like to provide a more complete implementation for the suggested ideas and test them more rigorously regarding real-world applications.

ACKNOWLEDGMENT

I would like to thank my supervisors Dr. Andrew Cullen and Prog. Tansu Alpcan along with the University of Melbourne Computing and Information Systems department for supporting this work.

BIBLIOGRAPHY

[1]
O. Vinyals *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, Oct. 2019, doi: https://doi.org/10.1038/s41586-019-1724-z.

[2]
OpenAI *et al.*, "Dota 2 with Large Scale Deep Reinforcement Learning," *arXiv:1912.06680 [cs, stat]*, Dec. 2019, Available: https://arxiv.org/abs/1912.06680

[3]
C. Yu *et al.*, "The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games," *arXiv.org*, Nov. 04, 2022. https://arxiv.org/abs/2103.01955

[4]
S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey," *Journal of Machine Learning Research*, Sep. 2020, Available: https://arxiv.org/abs/2003.04960

[5]
A. Rusu *et al.*, "POLICY DISTILLATION," 2016. Accessed: Jun. 19, 2024. [Online]. Available: https://arxiv.org/pdf/1511.06295

[6]
José J. F. Ribas-Fernandes *et al.*, "A Neural Signature of Hierarchical Reinforcement Learning," *Neuron*, vol. 71, no. 2, pp. 370–379, Jul. 2011, doi: https://doi.org/10.1016/j.neuron.2011.05.042.

[7]
A. Vaswani *et al.*, "Attention Is All You Need," 2016. Available: https://papers.neurips.cc/paper/7181-attention-is-all-you-need.pdf

[8]
T. B. Brown *et al.*, "Language Models Are Few-Shot Learners," *arxiv.org*, vol. 4, May 2020, Available: https://arxiv.org/abs/2005.14165

[9]
M. Pternea *et al.*, "The RL/LLM Taxonomy Tree: Reviewing Synergies Between Reinforcement Learning and Large Language Models," *arXiv.org*, Feb. 02, 2024. https://arxiv.org/abs/2402.01874 (accessed Jun. 19, 2024).

[10]
S. Mirchandani, S. Karamcheti, and D. Sadigh, "ELLA: Exploration through Learned Language Abstraction," 2021. Accessed: Jun. 19, 2024. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/f6f154417c4665861583f9b9c4afafa2-Paper.pdf

[11]
G. Wang *et al.*, "VOYAGER: An Open-Ended Embodied Agent with Large Language Models," 2023. Available: https://arxiv.org/pdf/2305.16291

[12]
T. Carta, C. Romac, T. Wolf, S. Lamprier, O. Sigaud, and P.-Y. Oudeyer, "Grounding Large Language Models in Interactive Environments with Online Reinforcement Learning," Proceedings of ML Research, 2023. Accessed: Jun. 19, 2024. [Online]. Available: https://proceedings.mlr.press/v202/carta23a/carta23a.pdf

[13]
J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Openai, "Proximal Policy Optimization Algorithms," Aug. 2017. Available: https://arxiv.org/pdf/1707.06347

[14]
M. Chevalier-Boisvert *et al.*, "BABYAI: A PLATFORM TO STUDY THE SAMPLE EFFI- CIENCY OF GROUNDED LANGUAGE LEARNING," 2018. Accessed: Jun. 19, 2024. [Online]. Available: https://arxiv.org/pdf/1810.08272

[15]
A. Smit , P. Duckworth , N. Grinsztajn , T. D. Barrett , and A. Pretorius, "Should we be going MAD? A Look at Multi-Agent Debate Strategies for LLMs," *arxiv.org*, 2024. https://arxiv.org/html/2311.17371v2 (accessed Jun. 19, 2024).

[16]
Y. Du, S. Li, A. Torralba, J. Tenenbaum, I. Mordatch, and G. Brain, "Improving Factuality and Reasoning in Language Models through Multiagent Debate," 2023. Accessed: Jun. 19, 2024. [Online]. Available: https://arxiv.org/pdf/2305.14325

[17]
I. Mordatch and P. Abbeel, "Emergence of Grounded Compositional Language in Multi-Agent Populations," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018, doi: https://doi.org/10.1609/aaai.v32i1.11492.

[18]
Z. Zhou, B. Hu, C. Zhao, P. Zhang, and B. Liu, "Large Language Model as a Policy Teacher for Training Reinforcement Learning Agents," 2023. Accessed: Jun. 19, 2024. [Online]. Available: https://arxiv.org/pdf/2311.13373

[19]
B. Peng, C. Li, P. He, M. Galley, and J. Gao, "INSTRUCTION TUNING WITH GPT-4," 2023. Accessed: Jun. 19, 2024. [Online]. Available: https://arxiv.org/pdf/2304.03277

[20]
H. Touvron *et al.*, "LLaMA: Open and Efficient Foundation Language Models," *arXiv:2302.13971 [cs]*, Feb. 2023, Available: https://arxiv.org/abs/2302.13971

[21]
D. Hafner, "BENCHMARKING THE SPECTRUM OF AGENT CAPABILITIES," 2022. Accessed: Jun. 19, 2024. [Online]. Available: https://arxiv.org/pdf/2109.06780

[22]
J. Wei *et al.*, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models Chain-of-Thought Prompting," 2023. Available: https://arxiv.org/pdf/2201.11903

[23]
S. Zhang *et al.*, "Instruction Tuning for Large Language Models: A Survey," 2024. Available: https://arxiv.org/pdf/2308.10792