Memory-Based Advantage Shaping for LLM-Guided Reinforcement Learning (Student Abstract)

Narjes Nourzad[1]*, **Carlee Joe-Wong**[2]
[1]University of Southern California
[2]Carnegie Mellon University
nourzad@usc.edu, cjoewong@andrew.cmu.edu

Abstract

In environments with sparse or delayed rewards, reinforcement learning (RL) incurs high sample complexity due to the large number of interactions needed for learning. This limitation has motivated the use of large language models (LLMs) for subgoal discovery and trajectory guidance. While LLMs can support exploration, frequent reliance on LLM calls raises concerns about scalability and reliability. We address these challenges by constructing a *memory graph* that encodes subgoals and trajectories from both LLM guidance and the agent's own successful rollouts. From this graph, we derive a *utility function* that evaluates how closely the agent's trajectories align with prior successful strategies. This utility *shapes the advantage function*, providing the critic with additional guidance without altering the reward. Our method relies primarily on offline input and only occasional online queries, avoiding dependence on continuous LLM supervision. Preliminary experiments in benchmark environments show improved sample efficiency and faster early learning compared to baseline RL methods, with final returns comparable to methods that require frequent LLM interaction.

Introduction

Reinforcement learning (RL) has achieved state-of-the-art results across diverse domains (Nourzad et al. 2024; Liu et al. 2024), yet its limitations become evident in settings that more closely resemble real-world conditions. Sparse or delayed rewards, partial observability, and long decision horizons often result in poor sample efficiency and unstable training dynamics. Large language models (LLMs) can help address these issues by providing high-level priors, suggesting subgoals, and structuring exploration (Schoepp et al. 2025). At the same time, LLMs introduce new challenges. Their outputs may be unreliable, queries can be expensive, and continuous supervision risks diluting the role of environment-driven feedback. Thus, it is challenging to integrate LLM guidance in a way that enhances RL without creating excessive dependence. The goal is to ensure external signals complement rather than distort optimization.

In this work, we introduce an approach that incorporates external guidance into RL models through *advantage shaping*. Central to the method is a *memory graph* co-constructed from the agent's rollouts together with offline priors and infrequent online outputs from an LLM. The graph provides a compact representation of subgoals and task-relevant knowledge that evolves during training, reducing dependence on real-time LLM access. A utility signal derived from the memory graph is used to softly shape advantage estimates, guiding policy updates and improving early exploration. The influence of this shaping term becomes asymptotically negligible once the agent's policy surpasses the usefulness of LLM-derived guidance, ensuring that the convergence guarantees of PPO (proximal policy optimization) (Schulman et al. 2017) remain intact.

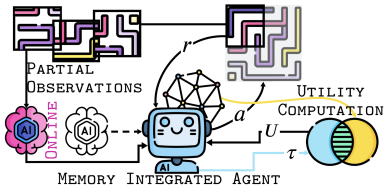

Figure 1: Overview of the proposed method.

Methodology

Our method integrates external guidance through a memory graph, initialized with offline LLM priors (e.g., partial trajectory and subgoal decompositions) and augmented with agent rollouts and occasional online queries. Online queries are triggered adaptively when the agent fails to extract useful guidance from memory for several consecutive episodes. The LLM is constrained to the same partial observability as the agent and returns short plans that are either added to the graph or used to bias action preferences through soft logit injection. Formally, the memory graph is given by

$$\mathcal{G} = \left\{ \left((o,a)_{\tau_j}, \zeta_j, \hat{r}_j \right) \right\}_{j=1}^{N} \cup \left\{ \kappa_\ell \right\}_{\ell=1}^{L} \cup \{ g_\triangleright \}.$$

Each trajectory node j consists of a partial observation o_{τ_j}, action a_{τ_j}, goal term $\zeta_j \in \{g_j, \kappa_\ell^{g_j}\}$ indicating a final goal (g_j) or an abstract subgoal $(\kappa_\ell^{g_j})$, and estimated reward $\hat{r}_j$ for the corresponding action sequence. The second set of nodes $\{\kappa_\ell\}_{\ell=1}^{L}$ represents subgoals inferred from environment descriptions, while the final term $\{g_\triangleright\}$ denotes the

*This work was conducted during an internship at Carnegie Mellon University.

agent's target goal(s). These nodes are connected through goal–subgoal relationships provided directly by the LLM. The graph evolves during training, with new nodes added when novel behaviors are observed and rarely accessed nodes pruned as obsolete to stay keep the structure compact while ensuring consistent guidance with little LLM reliance.

For each state-action pair (o_t, a_t), utility is computed by evaluating the agent's observed behavior against the memory graph. It leverages the same rollouts used for advantage estimation under the current policy. Each pair in the trajectory $\tau = (o_t, a_t)_{t=1}^{T}$ is compared to a stored trajectory node (m) from the memory graph, aligned with the environment layout in the current rollout. Formally, $U_t \doteq \hat{r}_m \cdot \rho(g_{\triangleright}, \zeta_m) \cdot \int\big((o_t, a_t), (o_{t'}, a_{t'})_{\tau_m}\big)$. The similarity term $\int(\cdot,\cdot)$ measures action agreement and how closely the agent's trajectory follows the path implied by the stored segment (e.g., position or directional overlap). The factor $\rho(\cdot,\cdot)$ serves to down-weight matches that are behaviorally similar but pursue different subgoals, computed as the Jaccard similarity between the goals in the rollout and those encoded in memory. This ensures that utility reflects both behavioral similarity and semantic alignment with successful prior strategies. Overview of the method in Fig. 1.

Algorithm 1: Shaped PPO actor (changes in blue)

Collect $\mathcal{D} = \{(s_t, a_t, r_t)\}$ using π_θ
Compute A_t and U_t from rollouts
$\tilde{A}_t = A_t + \xi_t U_t$
for *epoch* $= 1$ to K **do**
  for minibatch $\mathcal{B} \subset \mathcal{D}_k$ **do**
    $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_k}(a_t|s_t)$
    $\mathcal{L}^{\text{shaped}}(\pi_\theta) = \mathbb{E}\left[\min(r_t, 1\pm\varepsilon_k)\tilde{A}_t\right]$
    $\theta \leftarrow \theta + \alpha_\theta \nabla_\theta \mathcal{L}^{\text{shaped}}(\pi_\theta)$
  end for
end for

We incorporate memory-derived utility into the policy update by augmenting the standard advantage term. The advantage function quantifies how favorable an action is relative to the average action at state, reinforcing those with higher-than-expected returns and suppressing those that fall short. However, during early training the critic is poorly calibrated due to limited exploration, often producing near-uniform or noisy value estimates (Henderson et al. 2018). As a result, the computed advantages provide weak learning signals, even when the behavior is directed toward the task, leading to inefficient or unstable updates. This issue is more pronounced in sparse-reward or delayed-feedback settings, where A_t tends to be close to zero for most initial training timesteps. To address this, we define a *shaped advantage* $\tilde{A}_t = A_t + \xi_t U_t \quad$ where $0 < \xi_t \leq 1$. When critic feedback is weak, U_t supplies additional direction aligned with task objectives, compensating for flat or noisy gradients and accelerating learning. As training progresses and A_t becomes more reliable, the contribution of U_t naturally becomes negligible. Because the utility enters additively and does not modify the reward signal, the optimization dynamics of PPO are preserved, along with its convergence guarantees. More broadly, this mechanism is compatible with policy gradient methods relying on advantage estimation, providing a general way to incorporate structured guidance into RL.

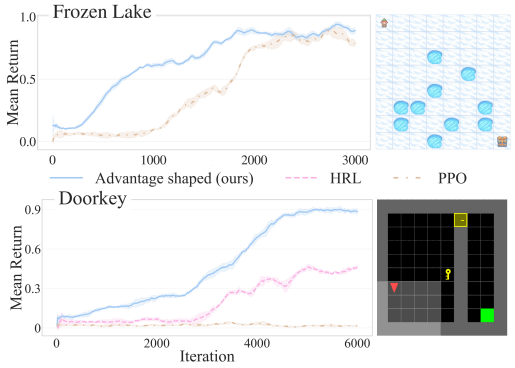

Figure 2: Mean return on FROZENLAKE-8X8 (top) and DOORKEY (bottom). Our method has the highest return.

Experimental Results

Experimental Setup We evaluate our method in two environments: FROZENLAKE (Gymnasium) and DOORKEY (MiniGrid suit), selected to probe early-stage exploration effects and long-horizon, partially observable planning, respectively. We compare against three baselines: PPO, trained tabula rasa from rewards, Hierarchical RL (HRL) (Bhambri et al. 2024), which uses LLM-derived option policies for temporal abstraction, and LLM4Teach (Zhou et al. 2023), a state-of-the-art method where a pre-trained LLM distills guidance into a student RL agent.

Figure 2 reports mean return across environments. In FROZENLAKE, shaped advantages achieve faster convergence, but baselines eventually reach the same asymptotic return with sufficient training. In DOORKEY, however, the shaped agent sustains a clear advantage, achieving both higher sample efficiency and stronger final performance. While HRL improves over PPO, it trails our method by nearly a factor of two. Table 1 reports test-time evaluation on unseen seeds for DOORKEY. Here, our method matches the state-of-the-art LLM4Teach in both mean return and success rate, with no statistically significant difference at the 95% confidence level. Unlike teacher-based approaches, however, these gains are achieved without continuous LLM supervision, relying only on limited offline and occasional online queries. These results indicate that memory-derived utility provides meaningful guidance in sparse and partially observable environments while minimizing the reliance on expensive LLM queries.

Method	Mean Return	Success Rate
LLM4teach	0.912 ± 0.075	0.970 ± 0.004
Our method	0.898 ± 0.093	0.953 ± 0.043

Table 1: Performance comparison on unseen seeds.

References

Bhambri, S.; Bhattacharjee, A.; Kalwar, D.; Guan, L.; Liu, H.; and Kambhampati, S. 2024. Extracting Heuristics from Large Language Models for Reward Shaping in Reinforcement Learning. *arXiv preprint arXiv:2405.15194*.

Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; and Meger, D. 2018. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Liu, J.-C.; Chang, C.-H.; Sun, S.-H.; and Yu, T.-L. 2024. Integrating planning and deep reinforcement learning via automatic induction of task substructures. In *The Twelfth International Conference on Learning Representations*.

Nourzad, N.; Coleman, J.; Zhao, Z.; Krishnamachari, B.; Verma, G.; and Segarra, S. 2024. Actor-Twin Framework for Task Graph Scheduling. In *The Seventeenth Workshop on Adaptive and Learning Agents*.

Schoepp, S.; Jafaripour, M.; Cao, Y.; Yang, T.; Abdollahi, F.; Golestan, S.; Sufiyan, Z.; Zaiane, O. R.; and Taylor, M. E. 2025. The evolving landscape of llm-and vlm-integrated reinforcement learning. *arXiv preprint arXiv:2502.15214*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. In *arXiv preprint arXiv:1707.06347*.

Zhou, Z.; Hu, B.; Zhao, C.; Zhang, P.; and Liu, B. 2023. Large language model as a policy teacher for training reinforcement learning agents. *arXiv preprint arXiv:2311.13373*.