





Enhancing Decision-Making of Large Language Models via Actor-Critic

Heng Dong*[1], Kefei Duan*[2], Chongjie Zhang[2]

¹Tsinghua University, ²Washington University in St. Louis

Motivation

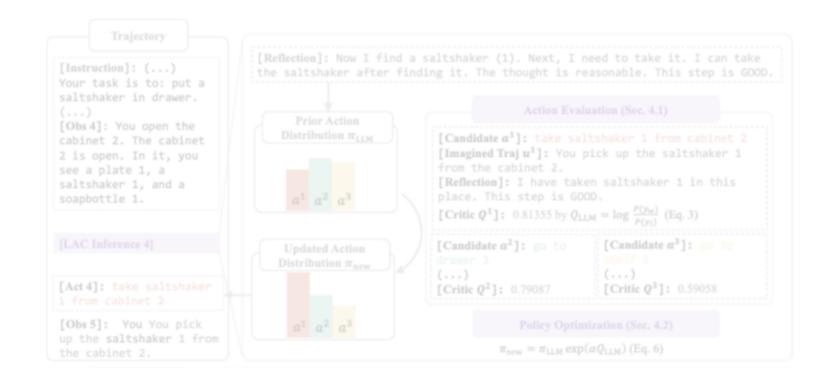
• LLMs are powerful in NLP but struggle in complex decision-making tasks.

- Challenges:
 - Limited long-term reasoning (auto-regressive bias).
 - Fragile rollout-based planning.
- Goal: Enable robust and scalable decision-making with LLMs using reinforcement learning insights.

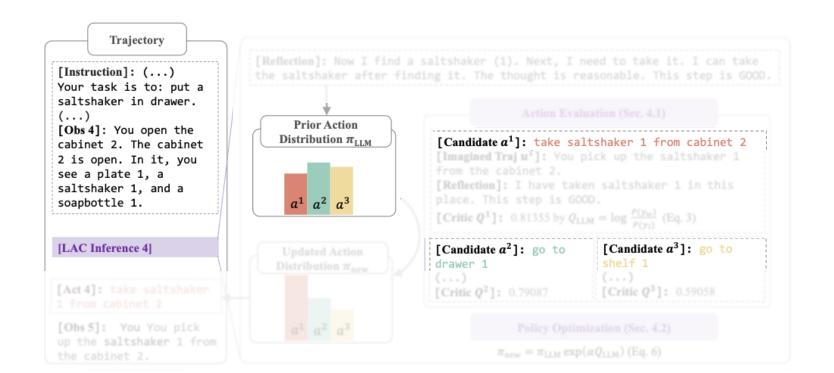
Key Idea — LAC Framework

- LAC: LLM-based Actor-Critic
 - Combines **LLM prior policy** (actor) with **action evaluation** (critic).
 - Critic uses logits of positive/negative tokens to compute Q-values.
 - Optimizes policy via gradient-free KL-constrained update.

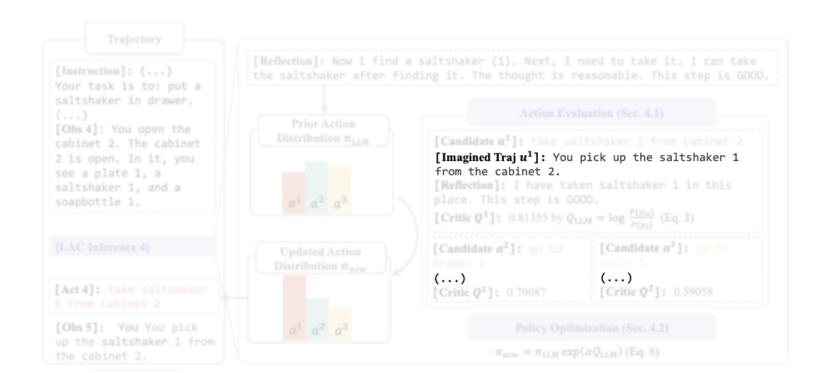
Trajectory Algorithm 1 LAC: LLM-based Actor-Critic algorithm. [Reflection]: Now I find a saltshaker (1). Next, I need to take it. I can take [Instruction]: (...) the saltshaker after finding it. The thought is reasonable. This step is GOOD. 1: **Input:** current task goal g, history h_t , actor π_{LLM} , for-Your task is to: put a ward model f_{LLM} , value-based critic Q_{LLM} , hyperpasaltshaker in drawer. Action Evaluation (Sec. 4.1) (...) rameter α , candidate action size n. Prior Action [Obs 4]: You open the Distribution π_{IIM} [Candidate a^1]: take saltshaker 1 from cabinet 2 2: Output: selected action a_t^* cabinet 2. The cabinet [Imagined Trai u^1]: You pick up the saltshaker 1 2 is open. In it, you 3: $\{a_t^i\}_{i=1}^n \sim \pi_{\text{LLM}}(\cdot|g,h_t);$ from the cabinet 2. see a plate 1, a 4: for i = 1 to n do [Reflection]: I have taken saltshaker 1 in this saltshaker 1, and a place. This step is GOOD. soapbottle 1. 5: $u_t^i \leftarrow f_{\text{LLM}}(g, h_t, a_t^i);$ > predict future trajectory [Critic Q^1]: 0.81355 by $Q_{LLM} = \log \frac{P(y_w)}{P(y_v)}$ (Eq. 3) 6: $\mathcal{Q}_{\text{LLM}}(g, h_t, a_t^i, u_t^i) \leftarrow \log \frac{P(y_w|g, h_t, a_t^i, u_t^i)}{P(y_t|g, h_t, a_t^i, u_t^i)}; \triangleright \text{action}$ [LAC Inference 4] [Candidate a^2]: go to [Candidate a^3]: go to Updated Action evaluation (Sec. 4.1) drawer 1 Distribution π_{new} 7: end for (···) [Act 4]: take saltshaker [Critic Q³]: 0.59058 [Critic Q²]: 0.79087 8: $\pi_{\text{new}}(a_t^i|g, h_t) \leftarrow \pi_{\text{LLM}}(a_t^i|g, h_t) \exp(\alpha \mathcal{Q}_{\text{LLM}}(g, h_t))$ 1 from cabinet 2 ▶ policy optimization (Sec. 4.2) [Obs 5]: You You pick Policy Optimization (Sec. 4.2) 9: $a_t^* \leftarrow \arg\max_{a_t^i} \pi_{\text{new}}(a_t^i|g, h_t)$ up the saltshaker 1 from $\pi_{\text{new}} = \pi_{\text{LLM}} \exp(\alpha Q_{\text{LLM}})$ (Eq. 6) the cabinet 2.



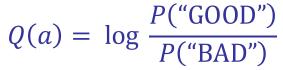
1. Sample actions from LLM (π_{LLM}) .

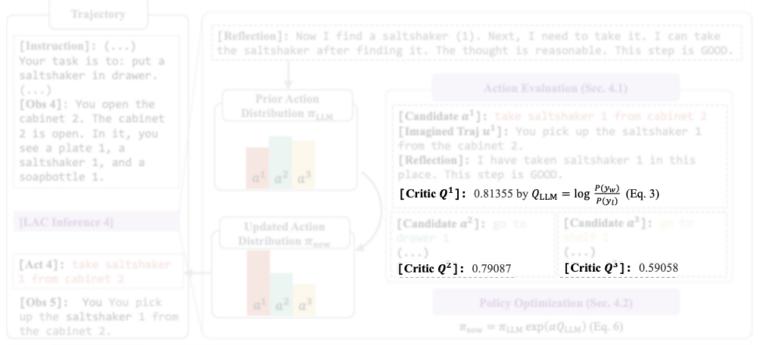


2. Predict future trajectories (via f_{LLM}).



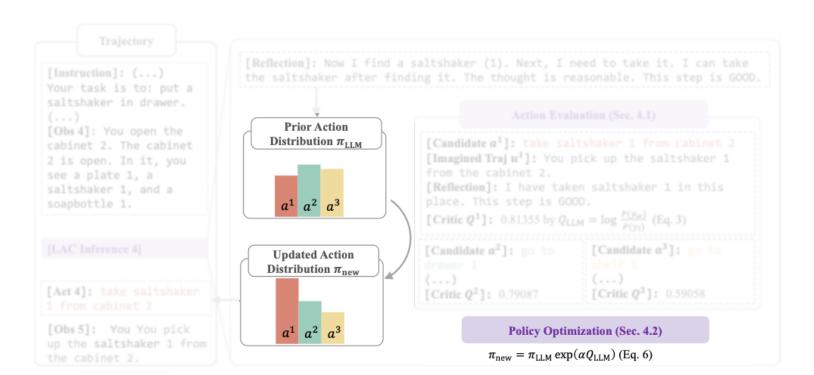
3. Evaluate each action:





4. Update policy:

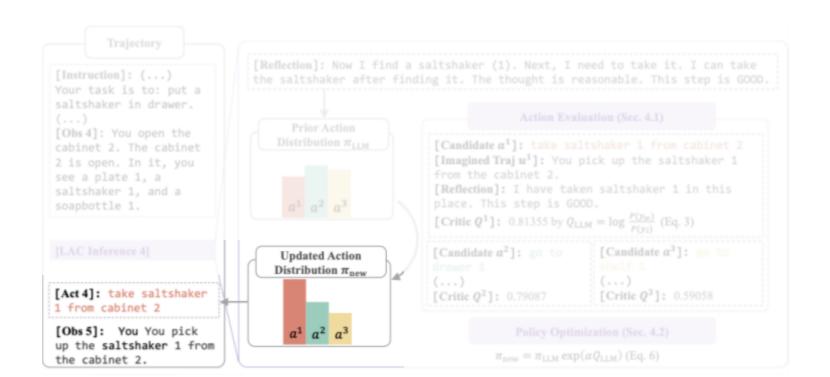
$$\pi_{new} \propto \pi_{LLM} \cdot \exp(\alpha Q)$$



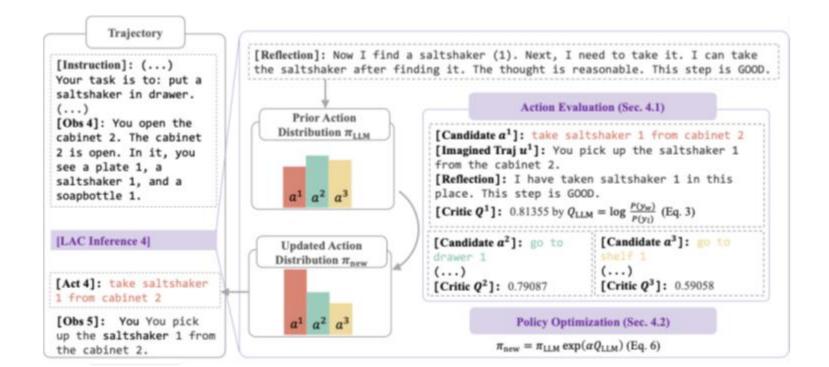
Reflections: LLM also generates self-judgments ("This step is GOOD") to aid evaluation.



• Sample an action from π_{new} and then execute it in the environment



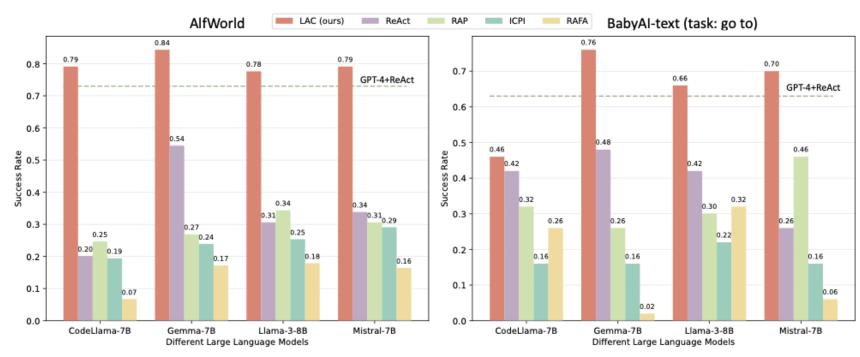
• This is the full algorithm process.



Results & Benchmarks

Benchmarks

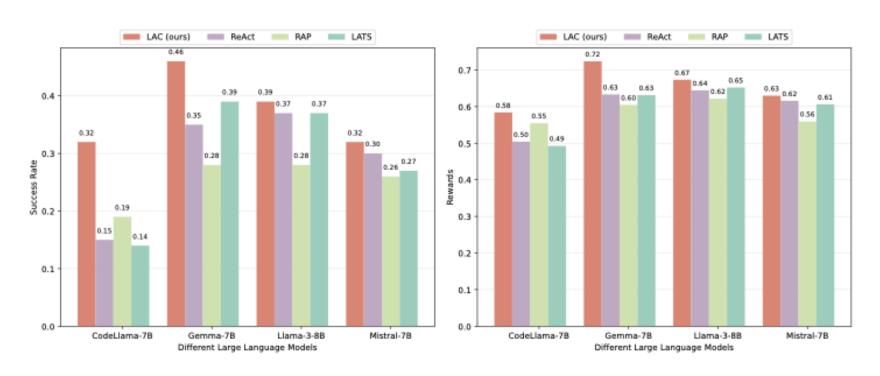
ALFWorld (high-level), BabyAI-Text (low-level), WebShop (infinite actions)



Results & Benchmarks

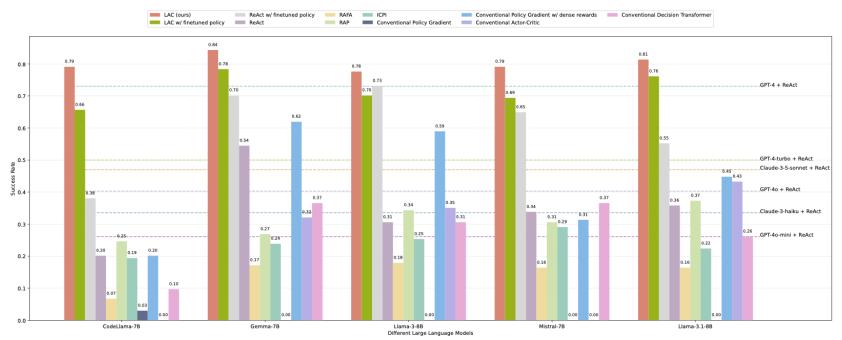
Benchmarks

ALFWorld (high-level), BabyAI-Text (low-level), WebShop (infinite actions)



Results & Benchmarks

- Highlights
 - Outperforms ReAct, RAP, ICPI, even GPT-4+ReAct in many tasks
 - Works well across multiple open-source LLMs (e.g., CodeLlama, Mistral)



Summary & Takeaways

- Summary
 - New way to combine LLM priors with evaluated planning
 - Works with lightweight LLMs
 - Gradient-free, interpretable, and generalizable
- **Future**
 - Deeper planning with trees, adaptation to continuous rewards.

Our code: https://github.com/drdh/LAC







Thanks for Your Listening