CHAPLIN: Cooperative Hierarchical Actor Policy Learning with Imitation Networks

Blair Yang Charlotte Chen Ray Hung Scott Cui

1. Introduction

We present CHAPLIN, a novel RL framework that combines Proximal Policy Optimization (PPO), Dreamer's environment modelling, and an auxiliary imitation task for enhanced learning efficiency. CHAPLIN integrates imitation learning to refine policy-making, allowing the use of new and historical data for rapid adaptation and robust knowledge retention in dynamic environments. By re-exploring older samples and decoupling policy and world-model, CHAPLIN achieves a balance between exploration and exploitation, and leverages both new and historical data to build an unbiased world model. We examine CHAPLIN's efficacy in multiple environments and explore its significance for RL development.

2. Related Work

Our research extends the core of policy gradient methodologies, which are pivotal to the progression of advanced RL.

Proximal Policy Optimization (Schulman et al., 2017) improved policy gradient approaches by introducing a simple proxy loss function that allows the re-usage of proximal data, leading to a substantial boost in sample efficiency.

Dreamer (Hafner et al., 2020) capitalizes on reconstruction loss to refine the model's environmental comprehension, yielding marked enhancements in both sample efficiency and model interpretability.

Auxiliary Task in Policy Gradient (Cobbe et al., 2020) pioneered a phasic learning structure that incorporates auxiliary tasks and consistent value learning to mitigate policy network overfitting and amplify sample reuse.

3. Potential Methodology

3.1. Data Processing

Given the nature of RL, we generate samples by interacting with environments. We may employ the following environments: Atari and Mujoco (Brockman et al., 2016), UCLA Competitive RL Envs (Peng et al., 2022), Rajeswaran et al., 2018), and PyMARL (Samvelyan et al., 2019).

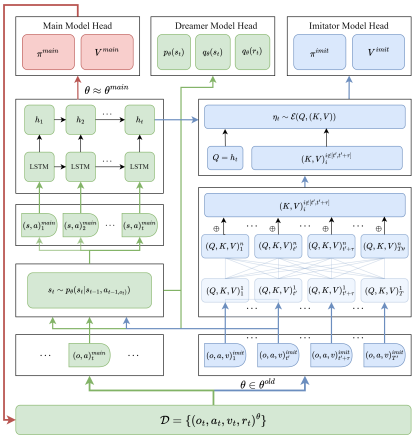

Figure 1. CHAPLIN Framework: Integrating Actor-Imitator Mechanisms. Experience samples are differentiated by freshness for dual-path processing. The Dreamer model evaluates all samples for latent state derivation. Fresh samples enhance policy/value decisions; older samples contribute to imitation loss. Color codes: green for universal application, red for policy/value updates, and blue for imitation analysis.

3.2. Architecture

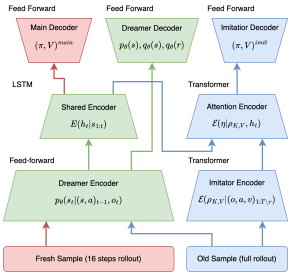

Figure 2. Our model. Samples are processed differently based on freshness: fresh samples for direct policy updates and older samples for imitation learning and representation refinement.

Our model leverages a combination of Transformers, feed-forward nets, and LSTMs, employing a priority queue to process samples based on recency. Fresh samples are encoded to update policies and values, while older samples

are encoded twice, passing through an additional encoder to capture historical policy styles for imitation learning, akin to an unsupervised learning task. All samples contribute to the Dreamer model's representation learning, encompassing both immediate policy optimization and historical policy integration. By utilizing this, we could decouple the policy training and dreamer's environment modeling.

3.3. Baseline Model

Our baseline model comprises Dreamer (Schulman et al., 2017), and a standard PPO implementation, both lacking the imitation component. This setup enables a direct comparison of CHAPLIN's performance enhancements attributed to the integrated imitation learning and enhanced sample efficiency. Models will undergo training across an identical number of episodes, with comparative analysis based on the final reward metrics.

3.4. Joint Imitation Learning

Our method can be formalized into the following equations:

$$\begin{cases} \text{Shared Encoder :} & E\left(h_t|s_{1:t}\right) \\ \text{Imitator Encoder :} & \mathcal{E}^{\text{imit}}\left(\eta_t|h_t,(o,a,v)^{\text{imit}}_{1:T\setminus\{t:t+\tau\}}\right) \\ \text{Imitator Policy Model :} & \pi^{\text{imit}}\left(a^{\text{imit}}_t|h_t,\eta_{1:t}\right) \\ \text{Imitator Value Model :} & V^{\text{imit}}\left(v^{\text{imit}}_t|h_t,\eta_{1:t}\right) \end{cases}$$

$$\begin{cases} \text{Policy Model :} & \pi\left(a_t|h_t\right) \\ \text{Value Model :} & V\left(h_t\right) \\ \text{Representation Model :} & p\left(s_t|s_{t-1},a_{t-1},o_t\right) \end{cases}$$

$$\begin{cases} \text{Transition Model :} & q\left(s_t|s_{t-1},a_{t-1}\right) \\ \text{Reward Model :} & q(r_t|s_t) \end{cases}$$

The loss for each model can be given by:

$$\mathcal{L}_{\text{imit}} = \begin{cases} \mathcal{L}^a_{\text{imit}}(\pi^{\text{imit}},a^{\text{main}}) & = \hat{\mathbb{E}}_t\left[-a^{\text{main}}_t\cdot\log(\pi^{\text{imit}}_t)\right] \\ \mathcal{L}^v_{\text{imit}}(v^{\text{imit}},v^{\text{main}}) & = \hat{\mathbb{E}}_t\left[\left(v^{\text{main}}_t - v^{\text{imit}}_t\right)^2\right] \end{cases}$$

$$\mathcal{L}_{\text{rep}} = \begin{cases} \mathcal{L}_{\text{KL}} = D_{\text{KL}}\left(p(s_{t+1}|s_t,a_t,o_{t+1})||q(s_{t+1}|s_t,a_t)\right) \\ \mathcal{L}_{\text{reward}} = \hat{\mathbb{E}}_t\left[\left(r_t-\hat{r}_t\right)^2\right] \end{cases}$$

$$\mathcal{L}_{\text{PPO}} = \begin{cases} \mathcal{L}_{\text{policy}} = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)A_t,\text{clip}(r_t(\theta),1\pm\epsilon)A_t)\right] \\ \mathcal{L}_{\text{value}} = \hat{\mathbb{E}}_t\left[\left(V_\theta(h_t)-\hat{V_t^{targ}}\right)^2\right] \\ \mathcal{H} = \hat{\mathbb{E}}_t\left[-\sum\pi_\theta(a_t|s_t)\log\pi_\theta(a_t|s_t)\right] \end{cases}$$

The total loss is given by:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{PPO}} + \mathcal{L}_{\text{rep}} + \mathcal{L}_{\text{imit}}$$

4. Ethical Considerations

Sustainability: By the nature of RL, we need to generate many samples for training, which is extremely resource-consuming. This also raises sustainability concerns.

Bias and Fairness: We don't employ human-generated datasets, so no bias will be present in our model.

5. Project Plan

5.1. Distribution of roles

Every team member will do experiments, analyze the results, and paper reading/writing. Besides that, each member has further roles.

- Blair: Designing models.
- Scott: Designing models, managing code base.
- Charlotte: Providing theoretical supports.
- Ray: Paper writing, scrum manager.

5.2. Internal Deadlines and Major Tasks

Our team will be scheduling either an in-person meeting or an online meeting each week, to discuss and assess the main tasks that need to be done this week. A detailed schedule is shown in Table 1.

Deadline	Task
March 3	Architecture Finalization
March 7	Environment Setup
March 14	Model Training & Experiments
April 7	Finalization + Writeup

Table 1. Tentative Schedule

6. Risk Register

- What if a member will not be contributing much to the project?

  A non-contributing member must document unfinished tasks and complete what they can by the deadline.

- What if the model cannot be trained within a reasonable amount of time?

  We have decent computational resources, which enables us to create a reasonably sized model and run it locally. We will try to find more resources if necessary.

- What if the project complexity turned out to be out of our expectations?

  We will do our best to achieve minimum expectations.

7. Project Repository

https://github.com/csc413-project/csc413-project

References

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Cobbe, K., Hilton, J., Klimov, O., and Schulman, J. Phasic policy gradient, 2020.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination, 2020.

Peng, Z., Hui, E., Zhang, Y., Ho, B., and Lam, J. Competitive rl environments. https://github.com/ucla-rlcourse/competitive-rl, 2022.

Rajeswaran, A., Kumar, V., Gupta, A., Vezzani, G., Schulman, J., Todorov, E., and Levine, S. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations, 2018.

Samvelyan, M., Rashid, T., de Witt, C. S., Farquhar, G., Nardelli, N., Rudner, T. G. J., Hung, C.-M., Torr, P. H. S., Foerster, J., and Whiteson, S. The StarCraft Multi-Agent Challenge. *CoRR*, abs/1902.04043, 2019.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.