JEPA-GUIDED SOFT GROUPING FOR STEP-LEVEL CREDIT ASSIGNMENT IN LONG-HORIZON LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon LLM agents trained with outcome rewards face a central credit-assignment challenge: final success or failure does not reveal which intermediate decisions contributed to the outcome. Step-level grouping provides finer-grained supervision by comparing the subsequent returns of intermediate steps across trajectories, making group construction central to effective credit assignment. However, constructing groups through exact observation text matching excludes comparisons between states that share task-relevant context, such as holding the same object for the same goal, but differ in their descriptions of the surroundings. We introduce JEPA-guided soft grouping to recover such comparisons in a learned feature space while retaining the original exact text groups. A task-conditioned JEPA state encoder is trained through future-latent prediction, and current-state feature similarity is used to retrieve neighbors from other trajectories of the same task, forming soft comparison sets. Differences between each step's observed discounted return and those of its neighbors yield supplementary step-level credit. Because feature similarity alone does not guarantee reliable comparisons, a selective gate controls the admission of this credit and a bounded residual limits its magnitude, while preserving the original episode-level and exact step-level advantages. On ALFWorld with Qwen2.5-1.5B-Instruct, our method achieves 92.97% success—the highest among the configurations in our main comparison—versus 87.76% for the reproduced exact-grouping baseline (+5.21 percentage points). These results support supplementing exact text grouping with learned feature space comparisons for step-level credit assignment.

1 INTRODUCTION

Long-horizon language-model agents receive sparse outcome rewards that do not identify which intermediate actions advanced the task. Group-relative optimization estimates advantages from sampled outcomes without a value network (Shao et al., 2024). A single trajectory-level signal, however, cannot distinguish locally useful decisions from mistakes within the same episode.

GiGPO addresses this limitation with a two-level comparison structure (Feng et al., 2025). It computes an episode-level relative advantage among trajectories sampled for the same task, then constructs exact step-level groups from repeated anchor observations. The second level supplies local comparative credit without a learned value model or additional counterfactual rollouts. Group construction determines which decisions can be compared and which return differences become policy-learning signals. Requiring identical observation text excludes states that share task-relevant context, such as the same held object and destination, but differ in their descriptions of the surroundings.

Naively replacing equality with semantic similarity is unsafe. Similar-looking observations can correspond to different task stages, inventories, remaining subgoals, or admissible actions. A false comparison can therefore push the policy in an incorrect direction rather than merely withholding supervision. This creates a reliability–coverage trade-off: exact groups are conservative but incomplete, whereas approximate groups are broader but can contaminate credit.

We study whether a predictive state representation can supply additional step-level comparisons while preserving GiGPO's original groups. Our method trains a lightweight, action-conditioned JEPA (Joint-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

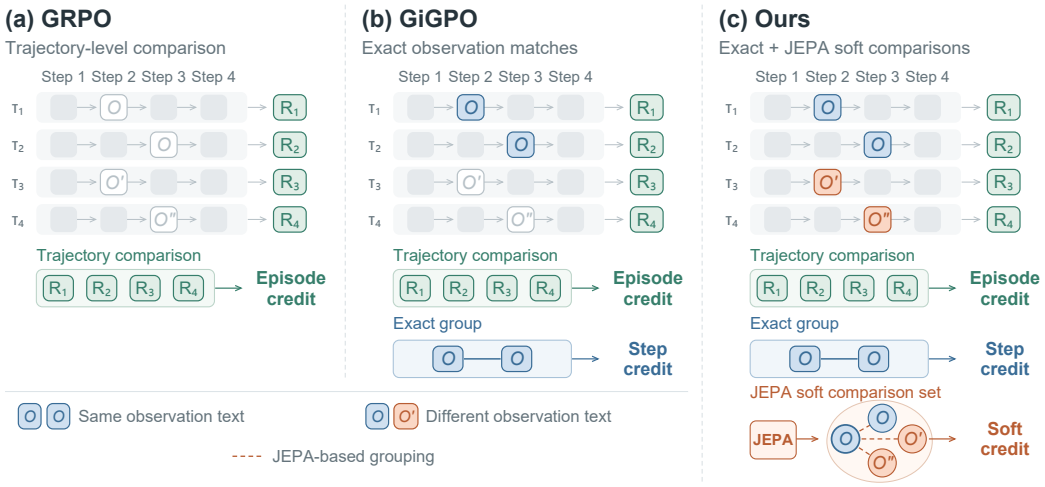


Figure 1: **Grouping for credit assignment.** All panels show the same four illustrative trajectories from one task; τ_i and R_i denote a trajectory and its outcome. (a) GRPO compares trajectory outcomes to obtain episode-level credit. (b) GiGPO additionally groups identical observations (matching O labels) to obtain step-level credit. (c) Ours retains both computations and uses JEPA embedding similarity to form a separate cross-trajectory soft comparison set, which can include textually different observations (O', O''). Query–neighbor discounted-return contrasts yield supplementary soft credit. Dashed orange edges denote selected query–neighbor relations, not state transitions; latent positions are schematic.

Embedding Predictive Architecture) on the trajectories already collected for policy optimization. Its task-conditioned state encoder produces representations whose similarity is used to retrieve same-task states from other trajectories into a separate soft comparison set, without changing membership in an exact group. Observed environment returns—not the JEPA prediction loss—determine the candidate credit. A selective gate admits this signal when the exact GiGPO signal is weak and, unless near zero, directionally compatible, after which a bounded residual limits its influence. Figure 1 summarizes this extension.

We make three contributions:

- Unlike exact text grouping, which restricts comparisons to textually identical observations, we use JEPA's predictive state representations to form supplementary comparison sets across trajectories of the same task. This extends step-level comparisons beyond textual identity while leaving the original exact groups intact.
- To control the risks of approximate comparisons, we separate proposing comparisons from admitting their resulting credit into policy optimization. A selective gate admits soft credit only when exact credit is weak, requiring sign compatibility outside a near-zero region, while a bounded residual limits its magnitude. This enables broader step-level supervision without allowing approximate comparisons to override strong exact credit.
- On ALFWorld (Shridhar et al., 2021) with Qwen2.5-1.5B-Instruct (Qwen Team, 2024), our method achieves 92.97% success—the highest among the configurations in our main comparison— versus 87.76% for reproduced GiGPO (+5.21 percentage points). Component studies examine the roles of predictive representations and conservative credit integration.

2 Related Work

Credit assignment under delayed feedback. Delayed rewards obscure the contribution of individual decisions. RUDDER redistributes a trajectory return using contribution analysis to produce more immediate learning signals (Arjona-Medina et al., 2019). For language-model reasoning, VinePPO instead estimates intermediate-state values from additional Monte Carlo continuations (Kazemnejad

et al., 2025), while SSVPO assigns step credit from marginal contributions computed over alternative reasoning chains (Li et al., 2026). HiPER partitions trajectories into subgoal segments and assigns planning and execution credit through hierarchical advantage estimation with learned value baselines, rather than group-relative return comparisons (Peng et al., 2026). These methods refine the reward or advantage assigned to a step. Our method addresses a complementary question: which already collected interaction steps should form the comparison set from which a relative advantage is estimated. It neither predicts rewards nor requires additional counterfactual rollouts.

Group-relative policy optimization for agents. GRPO removes the learned critic by normalizing outcomes within groups of sampled responses (Shao et al., 2024). GiGPO extends this principle to interactive agents by combining episode-level groups with exact anchor-state groups for local credit (Feng et al., 2025). HGPO further constructs hierarchical groups according to historical-context consistency and adaptively aggregates their advantages (He et al., 2026). Our method preserves GiGPO's episode and exact step advantages but targets a different failure mode: exact equality can omit useful cross-trajectory comparisons. A predictive representation proposes additional neighbors, and a selectively gated bounded residual prevents those proposals from replacing or reversing the direction of non-negligible exact credit.

Predictive representation learning for control. Joint-embedding predictive architectures learn representations by predicting targets in latent rather than observation space. I-JEPA demonstrated this principle for self-supervised image representation learning (Assran et al., 2023), and V-JEPA 2 extended action-conditioned latent prediction to robotic planning (Assran et al., 2025). We adopt the predictive principle but use it for neither visual reconstruction nor planning. Our action-conditioned, multi-horizon predictor is trained on online language-agent interactions; its online state embeddings only propose soft comparison groups, while observed environment returns determine the resulting credit. A conservative residual retains exact credit as the primary step-level signal.

3 METHOD

We introduce JEPA-guided soft grouping as a supplementary credit-assignment mechanism for GiGPO. We first define GiGPO's original advantages, then describe the task-conditioned representation, predictive learning, and construction of separate soft comparison sets. A selective gate and bounded residual control how the resulting credit supplements the unchanged GiGPO advantages. We conclude with the training order and inference procedure. Figure 2 summarizes the workflow.

3.1 PRELIMINARIES

For a task instruction u, the policy π_θ generates $N = 8$ environment trajectories. A trajectory $\tau = (s_0, a_0, r_0, \ldots, s_T)$ contains states, language actions, and environment rewards. Let d_i denote the rollout-group identifier associated with the task prompt of step i. We use G for return quantities and reserve A, E, and J for advantage signals. We keep the raw and penalized returns explicit because the exact and soft-group branches consume the return at different stages of the implemented training loop. The discounted return from step t is

$$G^{\text{env}}_{\tau,t} = \sum_{k=t}^{T_\tau - 1} \gamma^{k-t} r_{\tau,k}, \qquad \gamma = 0.95. \tag{1}$$

The implementation then applies a current-step invalid-action penalty,

$$\widetilde{G}_{\tau,t} = G^{\text{env}}_{\tau,t} - \lambda_{\text{inv}} q_{\tau,t}, \qquad \lambda_{\text{inv}} = 0.1, \tag{2}$$

where $q_{\tau,t} = 1$ indicates an invalid action.

GiGPO assigns each action an episode-level advantage A^{epi}_i and an exact step-level advantage E_i. The exact group $\mathcal{E}(i)$ contains steps with the same rollout-group identifier whose anchor observations are identical:

$$\mathcal{E}(i) = \{j : \; d_j = d_i \; \wedge \; o_j = o_i\}. \tag{3}$$

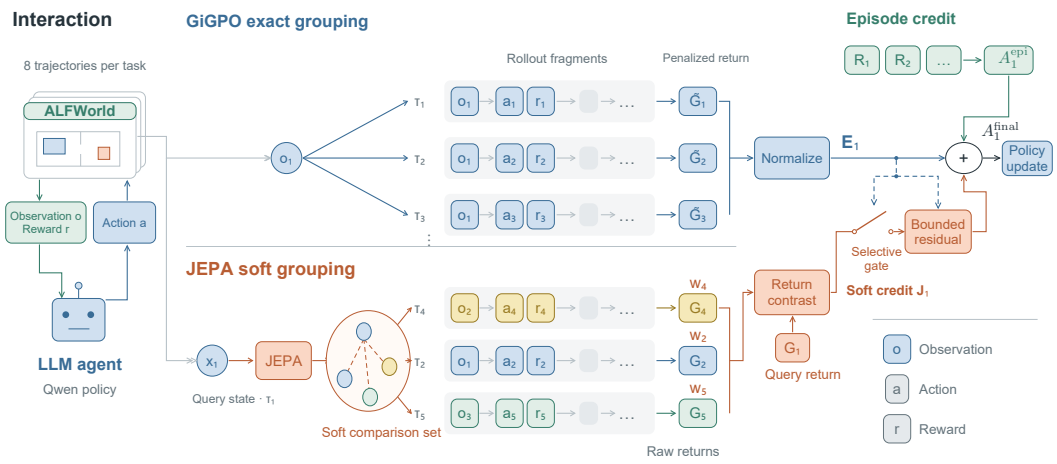


Figure 2: **Overview of JEPA-guided soft grouping.** Exact text grouping (top) yields GiGPO credit E_1. JEPA retrieves a separate soft comparison set from other trajectories of the same task (bottom); observed discounted returns yield J_1. The gate g_1 and bounded residual Δ_1, both conditioned on E_1, control its contribution to $A_1^{\text{final}} = A_1^{\text{epi}} + E_1 + \beta_k g_1 \Delta_1$. Orange dashed links mark retrieved neighbors; blue dashed arrows denote conditioning, and the green path carries episode-level credit. Group memberships are schematic.

This definition matches the implemented key: inventory and admissible actions do not enter the exact group. With mean–standard-deviation normalization,

$$E_i = \frac{\widetilde{G}_i - \mu_{\mathcal{E}(i)}}{\sigma_{\mathcal{E}(i)} + \epsilon}. \tag{4}$$

Singleton groups receive $E_i = 0$. The GiGPO advantage used in our comparison is

$$A_i^{\text{GiGPO}} = A_i^{\text{epi}} + E_i. \tag{5}$$

Our method leaves both terms unchanged and constructs an additional candidate comparison for selected steps.

3.2 Task-conditioned state representation

For every interaction step, we serialize the task instruction u, anchor observation o_t, inventory v_t, and an order-normalized summary m_t of the admissible actions:

$$x_t = \text{Serialize}(u, o_t, v_t, m_t). \tag{6}$$

The fields are separated explicitly before tokenization. The evaluated method uses the current-state retrieval representation. It does not include an explicit interaction history, completed-subgoal counter, or future action in the retrieval key. Importantly, x_t is an input to the JEPA encoder only; it is not the input to the exact grouping in Equation 3.

3.3 Action-conditioned multi-horizon JEPA

The JEPA module is separate from the Qwen policy and is trained only to learn a predictive state representation. Following the joint-embedding predictive principle (Assran et al., 2023), an online state encoder produces an L_2-normalized embedding

$$z_t = f_\eta(x_t), \qquad \|z_t\|_2 = 1. \tag{7}$$

A two-layer Transformer encodes states, and a two-layer MLP predicts future embeddings; architectural details are in Appendix A. Given the current action and a horizon embedding, it estimates a future-state embedding:

$$\widehat{z}_{t+h} = p_\phi\big(z_t, e_\psi(a_t), e_h(h)\big), \qquad h \in \{1, 4, 8\}. \tag{8}$$

The target is produced by a stop-gradient exponential-moving-average encoder,

$$z_{t+h}^{\mathrm{tar}} = \mathrm{sg}[f_\xi(x_{t+h})]\,, \tag{9}$$

whose momentum is 0.996. The training loss averages cosine prediction losses over the three horizons and adds a variance regularizer:

$$\mathcal{L}_{\mathrm{JEPA}} = \frac{1}{3}\sum_{h\in\{1,4,8\}} \left[1 - \cos\big(\widehat{z}_{t+h}, z_{t+h}^{\mathrm{tar}}\big)\right] + 0.05\mathcal{L}_{\mathrm{var}}. \tag{10}$$

Here, $\mathcal{L}_{\mathrm{var}}$ averages $\max(0, 0.05 - \sqrt{\mathrm{Var}_{\mathrm{pop}}(z_{:,\ell}) + 10^{-4}})$ over latent dimensions, with population variance over the JEPA batch. No reward or value prediction is used.

3.4 Constrained soft grouping

Retrieval uses online state embeddings, not predicted future embeddings. Eligible neighbors share the query's task-prompt rollout group but come from a different trajectory:

$$d_j = d_i, \qquad \tau_j \neq \tau_i, \qquad \cos(z_i, z_j) \geq 0.995. \tag{11}$$

Candidates are ranked by cosine similarity. The frozen configuration uses $K = 3$ and requires three neighbors. It additionally requires at least two first-token action categories across the query and its selected neighbors. If the initial top-three neighbors all share the query's action token and a later candidate differs, the last selected neighbor is replaced. This rule encourages action diversity but is not a semantic action parser or a counterfactual equivalence test.

For an accepted neighbor set $\mathcal{N}(i)$, similarity weights are

$$w_{ij} = \exp\left(\frac{\cos(z_i, z_j) - 1}{\tau_w}\right), \qquad \tau_w = 0.1. \tag{12}$$

The weighted return moments are

$$\mu_i^J = \frac{\sum_{j\in\mathcal{N}(i)} w_{ij} G_j^{\mathrm{env}}}{\sum_{j\in\mathcal{N}(i)} w_{ij}}, \tag{13}$$

$$(\sigma_i^J)^2 = \frac{\sum_{j\in\mathcal{N}(i)} w_{ij} (G_j^{\mathrm{env}} - \mu_i^J)^2}{\sum_{j\in\mathcal{N}(i)} w_{ij}}. \tag{14}$$

To avoid amplifying return differences when the three-neighbor variance is small, we lower-bound the normalization scale using the standard deviation s_{d_i} of raw discounted returns over all rollout steps in the query's task-prompt group. The candidate soft-group credit is

$$J_i = \mathrm{clip}\left(\frac{G_i^{\mathrm{env}} - \mu_i^J}{\max(\sigma_i^J, \lambda_k s_{d_i}) + \epsilon}, -10, 10\right). \tag{15}$$

Here s_{d_i} is an unweighted population standard deviation, including the query; λ_k controls the scale floor, active in iterations 51–150. Appendix A.1 specifies the schedule and reference setting. Thus, the representation determines which steps are compared, whereas observed environment returns determine the credit. In the evaluated implementation, J_i uses the raw discounted return because it is computed before the invalid-action penalty; E_i uses the penalized return in Equation 2. This asymmetry is an implementation choice rather than a theoretical requirement of soft grouping.

3.5 Selectively gated bounded residual

Retrieved neighbors are candidate comparisons, not guaranteed equivalent decision states. We therefore separate two safeguards: the gate decides whether to admit soft credit, and the residual cap limits its magnitude. Neither operation changes GiGPO's episode-level or exact step-level term (Figure 3).

Let c_i indicate that query i has a valid three-neighbor soft group. The selective gate is

$$g_i = \mathbb{I}[c_i]\mathbb{I}[|J_i| > 0.10]\mathbb{I}[|E_i| \leq 0.75]\mathbb{I}[|E_i| \leq 0.05 \vee \mathrm{sign}(E_i) = \mathrm{sign}(J_i)]\,. \tag{16}$$

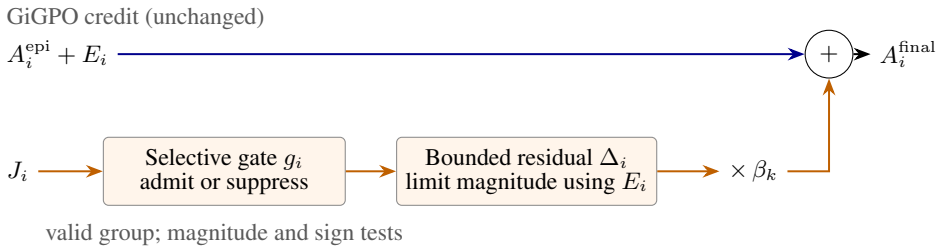


Figure 3: **Controlled supplementation of GiGPO credit.** The original episode-level advantage A_i^{epi} and exact step-level advantage E_i bypass the soft-credit branch unchanged. Candidate soft credit J_i is admitted or suppressed by the gate g_i; an E_i-dependent cap bounds the residual Δ_i. Only the scaled contribution $\beta_k g_i \Delta_i$ is added to the original advantages to form A_i^{final}. Equations 16–19 define the operations.

We cap the candidate residual relative to the exact signal:

$$C(E_i) = \begin{cases} |E_i|, & |E_i| > 0.05, \\ 1, & |E_i| \le 0.05, \end{cases} \tag{17}$$

$$\Delta_i = \operatorname{sign}(J_i) \min(|J_i|, C(E_i)) \,. \tag{18}$$

The final advantage is

$$A_i^{\text{final}} = A_i^{\text{epi}} + E_i + \beta_k g_i \Delta_i. \tag{19}$$

Let $k = 1, \ldots, 150$ index policy iterations. The residual coefficient follows

$$\beta_k = \begin{cases} 0, & k \le 11, \\ 0.25 \min\left(1, \frac{k-11}{20}\right), & k > 11. \end{cases} \tag{20}$$

Iterations 1–11 therefore use the GiGPO advantage exactly; β_k rises linearly from 0.0125 at iteration 12 to 0.25 at iteration 31. This coefficient scales admitted residuals, not the fraction of states or total credit supplied by JEPA. Only steps with valid neighbors that pass every gate receive the additional term.

3.6 Training order and inference

Each policy iteration follows six operations: (1) collect trajectories from the Qwen policy; (2) compute raw discounted returns; (3) use the pre-update online JEPA encoder to retrieve neighbors and compute J_i; (4) apply the invalid-action penalty and construct Equation 19; (5) update the policy with PPO; and (6) add current transitions to replay, optimize JEPA, and update its target encoder. This ordering prevents the current rollout from first training the representation and then immediately determining its own neighbor assignment. The policy loss does not backpropagate into JEPA. At evaluation time, only the trained Qwen policy is loaded; JEPA, replay, neighbor retrieval, and residual gating are training-time components.

4 Experiments

4.1 Experimental setup

We evaluate on ALFWorld, a text-based interactive environment aligned with the household tasks in ALFRED (Shridhar et al., 2021). Its six task families are pick-and-place, examine-under-light, clean-and-place, cool-and-place, heat-and-place, and two-object placement. Each episode is limited to 50 actions. Success is the environment's binary `won` indicator. Successful episodes receive environment reward 10; failures receive 0. Training additionally penalizes invalid actions by 0.1. We use Qwen2.5-1.5B-Instruct as the policy and no learned reward model.

The policy-training budget is 150 iterations, with 16 tasks and eight trajectories per task in each batch. The method uses the existing rollouts for both policy optimization and representation learning,

without expert-trajectory SFT or additional environment queries. Appendix A specifies optimization, training stages, initialization, and runtime settings; Appendix C documents model development and selection.

We evaluate fixed step-150 policies on 384 episodes across three sampling seeds. Overall success weights episodes equally; each task-family result is the mean of its three seed-specific success rates. Reported uncertainty is the sample standard deviation across evaluation seeds. The main comparison uses the same ordered task panel as reproduced GiGPO. Appendix B gives the sampling protocol and task-order audit.

4.2 Main results

Table 1 compares group-based and related structured credit assignment methods on ALFWorld with Qwen2.5-1.5B-Instruct. JEPA-guided soft grouping achieves 357/384 successes, compared with 337/384 for reproduced GiGPO, a gain of 5.21 percentage points. The largest task-family improvement is on two-object placement: 95.24% versus 79.46%. Our method achieves the highest overall success among the configurations listed in Table 1, at 92.97%. The HGPO row uses its legacy-framework result; literature results retain their original protocols (Appendix B.1).

Table 1: ALFWorld success rates (%) for group-based and related structured credit assignment with Qwen2.5-1.5B-Instruct. GRPO and GiGPO w/ std are reported by Feng et al. (2025); BEACON and GraphGPO are from Wang et al. (2026) and Cheng et al. (2026). HGPO ($K = 4$, legacy) is the In-Success result from He et al. (2026, Table 7); dashes indicate unreported task breakdowns. Task columns report mean success; Avg. includes sample standard deviation across three evaluation seeds for our checkpoints and reported uncertainty for literature results. Per-task uncertainty for our checkpoints is in Appendix B. Avg. is overall success, not an unweighted average of task-family means. Boldface highlights our method.

Method	Pick	Clean	Heat	Cool	Look	Pick Two	Avg.
GRPO	85.3	84.5	78.2	59.7	53.7	53.5	72.8 ± 3.6
GiGPO w/ std	94.4	94.8	94.4	79.8	67.5	76.4	86.7 ± 1.7
Reproduced GiGPO	90.19	86.90	98.41	89.91	80.68	79.46	87.76 ± 4.96
BEACON	100.0	86.7	100.0	78.9	88.2	92.9	91.4
GraphGPO	95.15	100.00	96.30	85.26	85.71	93.65	92.71 ± 1.32
HGPO ($K = 4$, legacy)	–	–	–	–	–	–	92.45 ± 0.81
Ours	**94.94**	**93.17**	**98.41**	**89.17**	**83.71**	**95.24**	$\mathbf{92.97 \pm 2.07}$

HGPO refines exact-state groups using historical context (He et al., 2026), whereas GraphGPO derives credit from a graph of aggregated states and transitions (Cheng et al., 2026). Our method instead retrieves non-identical observations as supplementary comparisons without replacing GiGPO's advantages. Its overall mean is 0.52 percentage points above legacy HGPO and 0.26 points above GraphGPO's main configuration.

4.3 Does soft grouping expand local credit?

In the reference-configuration diagnostics, valid soft groups cover an additional 3.03% of steps outside non-singleton exact groups, raising mean union coverage from 95.06% to 98.08%. This strict supplement excludes overlap with exact grouping and averages iterations 31–150. Appendix F provides the definitions, coverage curves, and provenance. These measurements show that soft grouping supplies comparisons unavailable to observation equality.

Which additional comparisons does JEPA retrieve? Figure 4 illustrates three retrieved pairs: different wording for an open cabinet while holding the same soapbar, corresponding bowl instances, and an additional object beside the same dishsponge. Observation equality excludes these comparisons. JEPA selects them into separate soft comparison sets, preserving the original exact groups.

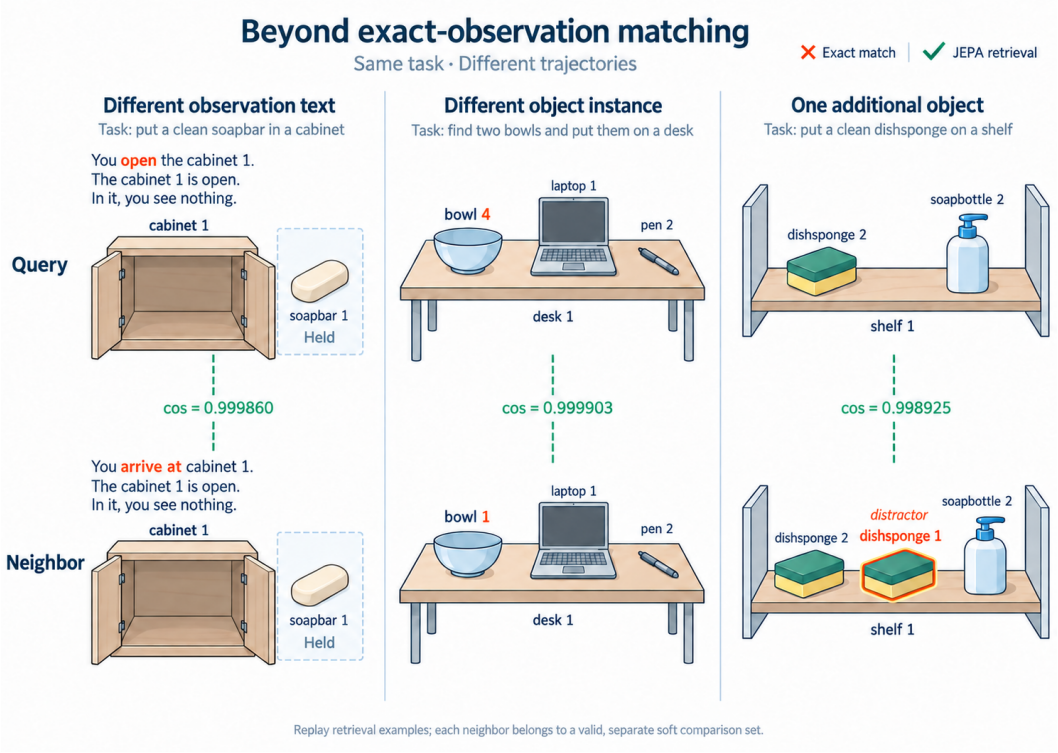


Figure 4: **Non-exact observations selected by reference JEPA retrieval.** Each column pairs a query (top) with a retrieved neighbor (bottom) from another trajectory of the same task. Orange highlights observation differences; green dashed lines denote retrieval relations, not state transitions, and "cos" reports JEPA embedding cosine similarity. Left: different wording describes the same open cabinet and held soapbar. Middle: the bowl instance changes at the same desk. Right: an additional dishsponge appears while the recorded next action remains "take dishsponge 2 from shelf 1"; "distractor" describes this local action context. Drawings summarize checkpoint-replay observations and inventory, not simulator screenshots. Each displayed neighbor belongs to a valid three-neighbor soft comparison set satisfying the full retrieval constraints; exact groups are unchanged. Extraction details are in Appendix H.

4.4 Gate, Residual, and Strength Studies

Component studies use reference JEPA (92.45%), rather than the final configuration with protected normalization (92.97%; Appendix A.2). Removing the gate's magnitude and sign tests retains the valid-neighbor mask, residual cap, and β schedule (91.41%). Direct fusion also removes the cap, adding βJ_i whenever neighbors are valid (91.93%). Reference JEPA has the highest observed mean of the three (Table 2).

Table 3 reports reference-configuration sensitivity to the residual coefficient. The default $\beta_{\max} = 0.25$ scores 92.45%, compared with 85.94% at 0.10 and 91.67% at 0.50. Increasing residual strength does not monotonically improve success, supporting 0.25 among the tested configurations.

Table 2: Gate and residual study with reference JEPA. Success (%), mean ± sample SD. Protocols: Appendix A.2.

Method	Gate	Cap	Success
No gate	–	Yes	91.41 ± 2.07
Direct fusion	–	–	91.93 ± 0.90
Reference	Yes	Yes	$\mathbf{92.45 \pm 1.97}$

Table 3: Reference JEPA: residual-strength sensitivity. The default 0.25 has the highest observed mean success (%).

$\beta_{\max}$	123	456	789	Mean
0.10	86.72	85.94	85.16	85.94
0.25	90.63	92.19	94.53	**92.45**
0.50	90.63	94.53	89.84	91.67

4.5 REPRESENTATION-LEARNING OBJECTIVES

We compare reference JEPA with a SimCSE-style current-state contrastive objective (Gao et al., 2021) and a CPC-style future-state contrastive objective (van den Oord et al., 2018). Both alternatives retain the state-encoder architecture, task-restricted retrieval, selective gate, bounded residual, and 150-iteration policy-training budget. Reference JEPA has the highest success among these configurations (Table 4), supporting its use as the default representation learner. SimCSE-style scores 90.10% and CPC-style 80.99%.

Table 4: Representation-learning configurations on ALFWorld, using reference JEPA for the JEPA row. Task columns report mean success (%); Avg. reports overall mean ± sample SD across three evaluation seeds. Best values in each column are bold. Configuration and evaluation details: Appendix D.

Representation	Pick	Clean	Heat	Cool	Look	Pick Two	Avg.
JEPA	93.01	**97.28**	**100.00**	**92.31**	**93.94**	79.96	$\mathbf{92.45 \pm 1.97}$
SimCSE-style	**97.10**	93.51	**100.00**	84.87	83.57	**81.20**	90.10 ± 1.19
CPC-style	92.74	89.45	93.87	73.66	69.37	65.32	80.99 ± 1.19

Higher retrieval coverage alone does not imply higher policy success. At iteration 150, CPC-style forms valid soft groups for 37.9% of steps versus 5.7% for SimCSE-style, with gate-admission rates of 10.1% versus 2.2%. Their mean absolute advantage changes are 0.013 and 0.003, respectively. These final-iteration observations highlight the importance of useful comparisons, rather than simply increasing the amount of supplementary credit.

Training and inference cost. Representation learning and neighbor retrieval add training computation but require no additional environment rollouts. At inference time, the method uses only the trained Qwen policy: no JEPA module, replay, or retrieval is required. Appendix G reports measured reference-run costs and distinguishes wall-clock overhead from intermediate threshold-reaching time.

5 LIMITATIONS

Our evaluation is limited to ALFWorld and a single policy-training run for the final configuration; multi-seed evaluation therefore does not establish robustness across independent training runs. Task instances recur across evaluation seeds, and the benchmark split informed development.

Learned similarity and action diversity guide comparison proposals rather than guarantee decision-state equivalence; gating and bounded residuals control their influence. The evidence supports the complete procedure, while isolating the contribution of predictive semantics and the raw-versus-penalized return asymmetry remains future work. Evaluation across independent training runs and additional environments would further establish generality.

6 CONCLUSION

We studied how to extend step-level credit assignment beyond exact textual grouping in long-horizon language-model agents. JEPA-guided soft grouping uses predictive state representations to propose supplementary cross-trajectory comparisons while preserving exact groups as the primary credit structure. Protected return normalization, selective gating, and bounded residuals control the resulting soft credit. On ALFWorld with Qwen2.5-1.5B-Instruct, the final configuration achieves 92.97% success—the highest among the configurations in our main comparison—versus 87.76% for reproduced GiGPO, including 95.24% versus 79.46% on two-object placement. These results support a broader principle: learned feature space comparisons can extend step-level supervision beyond textual identity without replacing exact groups or their credit signals.

AI USE STATEMENT

Generative AI tools assisted with text improvement, code writing, and related work discovery. The authors are responsible for verifying AI-assisted content and for the manuscript, code, results, and claims.

REPRODUCIBILITY STATEMENT

The method specification and experimental protocol are described in Sections 3 and 4. The appendix records the training stages, hyperparameters, evaluation protocol, and provenance of the main result and component studies. An anonymized code and artifact package will accompany the submission.

REFERENCES

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/16105fb9cc614fc29e1bda00dab60d41-Abstract.html.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture, 2023. URL https://arxiv.org/abs/2301.08243.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Xin Cheng, Shuo He, Lang Feng, HaiYang Xu, Ming Yan, Lei Feng, and Bo An. Beyond trajectory-level attribution: Graph-based credit assignment for agentic reinforcement learning, 2026. URL https://arxiv.org/abs/2605.26684.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training, 2025. URL https://arxiv.org/abs/2505.10978.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.552. URL https://aclanthology.org/2021.emnlp-main.552/.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=T8Dev99qnz.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *International Conference on Machine Learning*, 2025. URL https://proceedings.mlr.press/v267/kazemnejad25a.html.

Yugu Li, Zehong Cao, Jianglin Qiao, and Siyi Hu. SSVPO: Effective step-level credit assignment for RL training of language models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/4da4dd613cc9e63932d643b9c6e8f86f-Abstract-Conference.html.

Jiangweizhi Peng, Yuanxin Liu, Ruida Zhou, Charles Fleming, Zhaoran Wang, Alfredo Garcia, and Mingyi Hong. HiPER: Hierarchical reinforcement learning with explicit credit assignment for large language model agents, 2026. URL https://arxiv.org/abs/2602.16165.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning, 2021. URL https://arxiv.org/abs/2010.03768.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018. URL https://arxiv.org/abs/1807.03748.

Zixuan Wang, Yuchen Yan, Hongxing Li, Teng Pan, Dingming Li, Ruiqing Zhang, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Milestone-guided policy learning for long-horizon language agents, 2026. URL https://arxiv.org/abs/2605.06078.

A CONFIGURATION AND IMPLEMENTATION

Table 5: Settings for the main and reference JEPA configurations, shared unless marked otherwise. Credit-scale schedules are specified below.

Component	Value
Policy	Qwen2.5-1.5B-Instruct
Initialization	same immutable base checkpoint; no expert SFT
Environment	ALFWorld
Policy iterations	150
Training task batch	16
Rollouts per task	8
Maximum episode length	50 actions
Policy learning rate	10^{-6}
Policy optimizer at iteration 51	Reinitialized (final only)
Discount factor	0.95
Invalid-action penalty	0.1
JEPA retrieval embedding	current-state embedding
State encoder	2 layers, 4 heads, width 256, FFN 1,024
Embedding vocabulary / dropout	32,768 hashed entries / 0.0
Maximum state / action tokens	192 / 32
Prediction horizons	1, 4, 8
Target EMA momentum	0.996
Prediction-loss variance weight	0.05
Replay capacity / JEPA batch	100,000 / 256
JEPA updates per iteration	at most 30
Similarity threshold / temperature	0.995 / 0.1
Neighbors	exactly 3 when valid
Action diversity	enabled, first-token categories
Exact fallback in JEPA branch	disabled
History filter	disabled
β schedule	zero through 11; linear ramp to 0.25 at 31
PPO mini-batch / per-GPU micro-batch	256 / 8
Policy KL loss coefficient	0.01

Architecture. The state encoder has two Transformer layers, four attention heads, hidden and latent widths of 256, and feed-forward width 1,024. Qwen tokenizer IDs map to a 32,768-entry embedding table through modular hashing; state and action inputs are capped at 192 and 32 tokens. The action encoder shares the online token-embedding table. The predictor is a two-layer MLP with a 1,024-unit hidden layer and GELU activation.

Implementation record. The reference runtime uses Python 3.10.18, PyTorch 2.6.0 with CUDA 12.4, Ray 2.43.0, vLLM 0.8.2, and Transformers 4.51.3. State serialization uses explicit task, observation, inventory, and admissible-action fields. Exact grouping is keyed only by the task-prompt rollout group and anchor observation. State fields are assembled before whole-text tokenization and truncation at 192 tokens. The main and reference runs retain the legacy action-derived inventory tracker, including its known failure to confirm every take/put update against environment feedback. Corrected-inventory and field-budget experiments are separate configurations, not the source of the main result. No Progress filter, task resampling, or policy-context augmentation is used in the final main configuration.

A.1 Main configuration and training stages

The main result is the fixed step-150 checkpoint of a training-seed-0 run. Protected return normalization is disabled for iterations 1–50 and enabled for iterations 51–150:

$$\lambda_k = \begin{cases} 0, & k \leq 50, \\ 0.5, & 51 \leq k \leq 150. \end{cases} \tag{21}$$

The task-scale pool contains all valid rollout steps associated with the original task group, including the query. It is not restricted to selected neighbors, actions, or progress stages. The standard deviation uses the population denominator. The numerical constant in credit normalization is $\epsilon = 10^{-6}$.

The implementation evaluates β before incrementing a zero-initialized JEPA iteration counter after PPO. At policy iteration k, this counter is $k - 1$; Equation 20 uses the one-based policy index.

Training diagnostics showed limited participation of JEPA soft credit during the first 50 iterations, motivating activation of protected normalization from iteration 51. Policy weights, JEPA states, and replay are retained across the phase transition; the policy optimizer setting is recorded in Table 5.

Validation and checkpointing occur every five policy iterations; the main result uses the pre-specified terminal iteration 150 rather than the best intermediate validation score. Equation 21 records the evaluated procedure, not an independently validated optimal switch point.

A.2 Reference configuration and component-study provenance

We use *reference JEPA* to denote the original configuration with $\lambda_k = 0$ throughout training. Its step-150 checkpoint scores $92.45 \pm 1.97\%$. The gate/direct-fusion study, residual-strength sensitivity, and JEPA representation-control row retain this reference result rather than substituting the final $92.97 \pm 2.07\%$ checkpoint. Frozen-replay cases, online coverage, paired-outcome statistics, and historical cost measurements also refer to this reference checkpoint or run unless stated otherwise.

The final configuration uses the scale-floor schedule and phase-transition optimizer setting specified above. The main comparison therefore evaluates complete trained systems; it does not isolate the causal effect of the scale floor. No component or result from a separate JEPA-only credit-replacement project is used in this manuscript. Artifact identifiers and hashes are retained in the reproducibility record.

B Evaluation protocol and task-order verification

All main evaluations use ALFWorld's in-distribution seen split (`valid_seen`), not `valid_unseen`. The environment mode is `eval_in_distribution`; each seed samples 128 episodes from the 140-game seen evaluation pool.

Each fixed checkpoint is evaluated with sampling seeds 123, 456, and 789, 128 episodes per seed, sampling temperature 0.4, and a 50-action limit. Overall mean success equals pooled success because each seed has the same episode count. For each task family, we average the three seed-specific rates; we do not average over the six families to obtain overall success. All reported $\pm$ values for our evaluations are sample standard deviations across evaluation seeds, not independent training runs.

The main run freezes the historical evaluation-directory order. We reconstruct its task assignment with the installed TextWorld sampling routine and actual evaluation commands, then compare (seed, episode index, task key) with the historical JEPA and GiGPO outcome records. All 384 assignments match, including order and repeated occurrences. This is a deterministic reconstruction, not a newly saved per-episode task manifest. The final run retained aggregate success metrics but not individual success labels; consequently we do not report a new paired repair/regression count or task-clustered confidence interval for its improvement.

The seeds contain 85, 84, and 77 distinct task keys, respectively, and 130 in their union. For reference JEPA versus GiGPO, saved individual outcomes give 327 joint successes, 19 joint failures, 28 repairs, and 10 regressions. A 50,000-replicate paired bootstrap resamples the 130 task-key clusters while preserving all occurrences of each selected key. Its 95% percentile interval is $[0.00, 9.65]$ percentage points for the reference $+4.69$-point difference. These counts and this interval do not apply to the final configuration.

Table 6: Main evaluation panel and seed-specific success. Counts preserve repeated task instances; total successes are out of 128 for each seed.

Quantity	123	456	789	Total
Pick episodes	35	32	34	101
Clean episodes	27	25	24	76
Heat episodes	13	12	21	46
Cool episodes	26	30	18	74
Look episodes	11	8	10	29
Pick Two episodes	16	21	21	58
GiGPO successes	105	116	116	337
Reference JEPA successes	116	118	121	355
Final JEPA successes	116	120	121	357

Table 7: Per-task success (%), mean ± sample standard deviation across evaluation seeds, on the historical task panel.

Task	Reproduced GiGPO	Reference JEPA	Final JEPA
Pick	90.19 ± 3.09	93.01 ± 2.07	94.94 ± 3.74
Clean	86.90 ± 1.50	97.28 ± 2.36	93.17 ± 6.33
Heat	98.41 ± 2.75	100.00 ± 0.00	98.41 ± 2.75
Cool	89.91 ± 14.68	92.31 ± 13.32	89.17 ± 14.21
Look	80.68 ± 23.48	93.94 ± 10.50	83.71 ± 18.48
Pick Two	79.46 ± 2.84	79.96 ± 8.08	95.24 ± 4.76
Overall	87.76 ± 4.96	92.45 ± 1.97	92.97 ± 2.07

B.1 Scope of literature comparisons

Table 1 focuses on group-based and related structured credit assignment: HGPO uses hierarchical state groups, GraphGPO uses state aggregation, and BEACON uses milestone-aligned segment comparisons. HiPER uses actor-critic hierarchical advantage estimation instead of group-relative return comparisons and is discussed in related work.

Our source lineage is the pre-major-update `verl-agent` commit `f974dc5` (June 1, 2025). We therefore use HGPO's legacy-framework $K = 4$ In-Success of $92.45 \pm 0.81\%$ from its Table 7 as the closer framework reference (He et al., 2026). This does not establish identical commits or evaluation samples. HGPO also reports $94.85 \pm 0.92\%$ In-Success with the updated framework in Table 1. GraphGPO's main setting scores $92.71 \pm 1.32\%$, while its $\omega = 0.4$ sensitivity setting scores $92.97 \pm 0.63\%$ (Cheng et al., 2026). The ranking in our table therefore concerns the listed configurations, not an all-configuration state-of-the-art claim.

C Development and implementation details

A replay scan held out task identities within collected training replay and compared representation mixtures, similarity thresholds, and neighborhood sizes. A later frozen-replay diagnostic compared candidate gates and residual scales. The current-state representation, conservative gate, warm-up, ramp, and first-token diversity rule were engineering choices frozen through policy-training pilots. These diagnostics did not select the complete configuration through a single scalar criterion. The benchmark split was visible during validation and development, and the final main configuration was selected after examining exploratory results. It is therefore a development-visible evaluation, not an untouched test set. The final $+0.52$-point difference from reference JEPA is two net successes; it does not establish reproducibility across training seeds. The main result has higher Pick Two success but lower Clean, Heat, Cool, and Look means than reference JEPA.

JEPA uses AdamW with learning rate 3×10^{-4} and weight decay 10^{-4}. Each policy iteration performs at most 30 updates of batch size 256 when replay contains samples at all prediction horizons. The replay capacity is 100,000. The cosine threshold is 0.995 and neighbor temperature is 0.1.

D REPRESENTATION CONTROLS AND DIRECT-FUSION EVALUATION

The SimCSE-style and CPC-style controls start from the base Qwen2.5-1.5B-Instruct policy and use training environment seed 0, 150 policy iterations, the original task sampler, and the same small state-encoder architecture. Both use replay capacity 100,000, auxiliary batch size 256, 30 auxiliary updates per policy iteration, and AdamW learning rate 3×10^{-4}. Retrieval uses the current-state encoder, cosine threshold 0.995, three cross-trajectory neighbors from the same task, and action diversity; the selective gate and bounded residual are retained. All representation controls use the reference normalization without the task-scale floor.

SimCSE-style uses two independent dropout views of the current state, dropout probability 0.1, a symmetric in-batch contrastive loss, and temperature 0.05. CPC-style uses action-conditioned predictions at horizons $\{1, 4, 8\}$ with an in-batch InfoNCE loss at temperature 0.1 and online future-state targets. Neither contrastive control uses the JEPA EMA-target regression or variance regularizer. Identical tokenized inputs are masked as negatives. These are objective adaptations to our encoder, not reproductions of the original pretrained SimCSE or CPC systems.

All reported checkpoints are fixed at iteration 150 and evaluated using sampling seeds 123, 456, and 789, 128 episodes per seed, temperature 0.4, and a 50-action limit. SimCSE-style and CPC-style have identical realized task lists and ordering. The JEPA row reuses the reference-run evaluation, whose realized task list differs. The two new controls use two A800 policy GPUs and one auxiliary A800, whereas the historical JEPA run uses four RTX 4090 policy GPUs and one auxiliary RTX 4090. Thus the table compares tested complete configurations rather than isolating the learning objective under identical training and evaluation realizations. The common cosine threshold is fixed across objectives, not separately calibrated to each representation. Reported standard deviations describe evaluation-seed variation, not independent training replications.

Direct fusion removes both gate conditions and the residual cap, but retains valid-neighbor eligibility and the original β schedule. Its seed-0 step-150 success rates are 92.97%, 91.41%, and 91.41%, respectively ($91.93 \pm 0.90\%$). The no-gate variant retains the cap and scores 89.84%, 93.75%, and 90.63% ($91.41 \pm 2.07\%$). Reference JEPA scores 90.63%, 92.19%, and 94.53% ($92.45 \pm 1.97\%$). Direct fusion uses the new-server evaluation task lists rather than the historical reference list; its mean comparison is descriptive and does not isolate a gate-only effect. The gate study's direction varies across evaluation seeds. The residual-strength experiments retain their original results; hardware and policy world size also differ between the non-default runs and the historical default, so this sensitivity study is not a fully isolated estimate of the coefficient's effect.

E FROZEN-REPLAY DIAGNOSTICS

A stratified sample of 6,000 transitions from reference JEPA's final 100,000-transition replay has 66.1% soft-group coverage and mean neighbor cosine similarity 0.9982. Among retrieved neighbors, 16.3% share exactly the same anchor observation; 83.7% are non-exact textual matches. Accepted groups average 3.04 distinct action strings and 2.02 first-token action categories; 15.6% contain more than one episode outcome. This sampling procedure differs from the online diagnostics, so its coverage is not directly comparable to Table 8. These statistics describe retrieved groups and do not establish decision-state equivalence. A separate frozen-replay shuffle diagnostic did not show superiority over coverage-matched random representations under its proxy; it does not replace the policy-training comparisons in Section 4.5.

F REFERENCE-RUN ONLINE CREDIT COVERAGE

This reference-run analysis distinguishes comparisons already available to exact grouping from steps newly covered by valid JEPA neighbor sets. Define

$$\mathcal{C}_{\text{exact}} = \{i : |\mathcal{E}(i)| \geq 2\}, \qquad \mathcal{C}_{\text{soft}} = \{i : c_i = 1\}, \qquad \mathcal{C}_{\text{supp}} = \mathcal{C}_{\text{soft}} \setminus \mathcal{C}_{\text{exact}}. \tag{22}$$

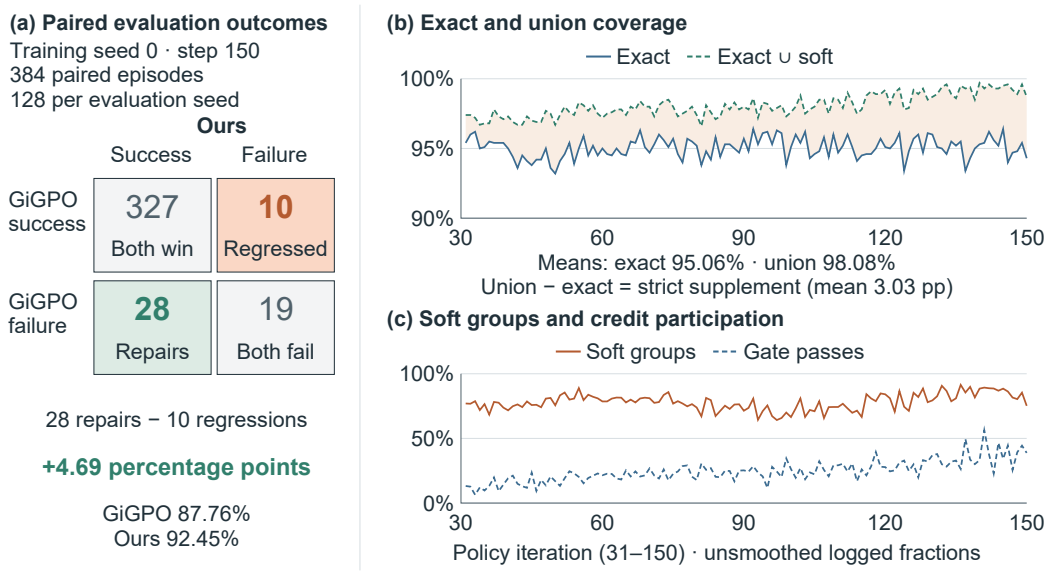


Figure 5: **Reference JEPA: paired outcomes and online credit coverage.** (a) The 384 paired episodes yield 28 repairs and 10 regressions. (b,c) Unsmoothed fractions over iterations 31–150. Exact coverage counts non-singleton groups; soft coverage counts valid JEPA neighbor sets. Strict supplement excludes exact-covered steps. The shaded exact-to-union gap averages 3.03 points; it is not a confidence interval. These are reference-run diagnostics, not outcomes of the final configuration. Gate passes measure residual admission. Means weight iterations equally; axes span 90–100% in (b) and 0–100% in (c).

Exact coverage counts steps in non-singleton GiGPO groups; soft-group coverage counts steps with valid cross-trajectory JEPA neighbor sets. Only the strict supplement $\mathcal{C}_{\text{supp}}$ identifies newly covered steps.

Table 8: Reference-run online coverage diagnostics (%). Iterations 31–150 are averaged equally rather than pooled over generated steps. Strict supplement excludes exact-covered steps.

Scope	Exact	Soft group	Strict supplement	Union
Iterations 31–150	95.06	78.09	3.03	98.08
Iteration 150	94.3	75.2	4.4	98.7

After the β ramp, exact coverage averages 95.06% and soft coverage 78.09%; their 3.03% strict supplement raises union coverage to 98.08%, using unrounded per-iteration values. At iteration 150, the overlap is 70.8%, and the 4.4% strict supplement raises coverage from 94.3% to 98.7%. Figure 5 shows the full curves alongside the paired evaluation outcomes.

At iteration 150, 38.8% of samples pass the selective gate, and 14.1% cross the 0.05 step-credit magnitude threshold from $|E_i| \leq 0.05$ after the residual is applied. The mean absolute change over valid response tokens is 0.054; all advantages remain finite. Thus, a measurable subset of retrieved groups changes the advantages consumed by PPO. These diagnostics do not isolate JEPA semantics as the unique cause of the policy gain.

G Reference-run training cost

Across common iterations 51–150, reproduced GiGPO takes 777 seconds per iteration on average and reference JEPA takes 868 seconds, an observed increase of 11.7%. The interval requires 21.59 wall-clock hours and 86.35 allocated GPU-hours for GiGPO, compared with 24.12 hours and 120.61

allocated GPU-hours for reference JEPA. Different generated trajectory lengths also affect runtime, so the full difference is not attributable to auxiliary computation alone. These measurements are not the cost of the final A800 run.

Measured from iteration 50 to the first internal validation success of at least 87.5%, JEPA-guided training takes 16.20 hours and 80.98 allocated GPU-hours, versus 20.56 hours and 82.26 GPU-hours for GiGPO. These reference-run measurements are a first-passage observation for the reported seed-0 runs, not evidence of uniformly faster convergence. This comparison excludes the first 50 iterations and does not estimate end-to-end compute savings.

H SOFT-GROUP CASE EXTRACTION

We apply the production retrieval function to a reproducible sample of 6,000 transitions from the reference checkpoint's 100,000-transition replay buffer. The extraction yields 3,419 valid three-neighbor groups, of which 3,361 contain at least one neighbor with a different observation from the query. In total, 8,162 of the 10,257 selected query–neighbor pairs are non-exact observation matches. These are replay-sample statistics; Table 8 reports coverage measured directly during online training.

Figure 4 shows three selected examples. The observation excerpts and next actions are taken from the stored records; object diagrams summarize those observations and the recorded inventory field. Each displayed neighbor belongs to a valid group under the full retrieval constraints. The exact group is unchanged, and the retrieved neighbor enters a separate soft comparison set. The cases illustrate the local basis for comparison: shared inventory and next action, corresponding object instances, and a shared next action in the presence of an additional object.

I REPRESENTATION LEARNING AND RETRIEVAL

Figure 6 details the predictive model and neighbor selection.

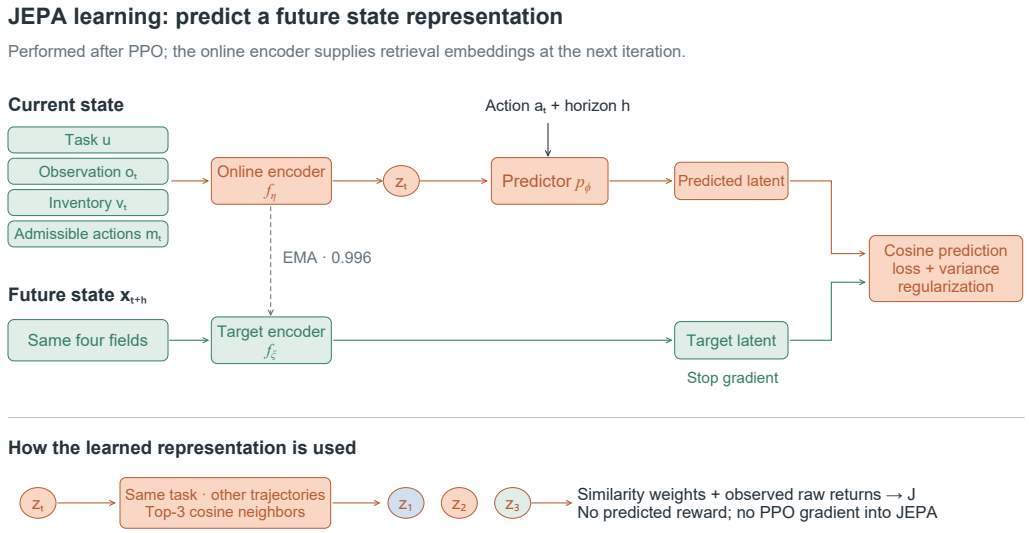


Figure 6: **Predictive representation learning and constrained retrieval.** The upper diagram shows predictive learning. The two-layer, four-head online encoder produces a 256-dimensional embedding. An action-conditioned MLP predicts $\widehat{z}_{t+h}$ for $h \in \{1, 4, 8\}$; the stop-gradient EMA encoder produces z^{tar}_{t+h} with momentum 0.996. The action encoder shares the online token table. The loss averages horizon cosine losses and adds $0.05\mathcal{L}_{\text{var}}$ as defined in Method. The lower diagram shows retrieval, which uses online state embeddings from the current rollout, not predicted embeddings or replay neighbors. Candidates must share the task-prompt rollout group, belong to a different trajectory, and have cosine similarity at least 0.995. Exactly three neighbors are selected, with at least two first-token action categories across query and neighbors; the last neighbor may be replaced to meet this constraint. Neighbors need not come from three distinct trajectories. Similarity-weighted raw returns define μ^J_i, σ^J_i; Equation 15 forms the credit J_i using the rollout-group scale floor. JEPA predictions are not rewards.

J GATE AND RESIDUAL DETAILS

Figure 7 makes the conjunction in Equation 16 explicit and summarizes the residual cap and schedule.

Selective gate: all conditions required

$$c_i = 1 \ \wedge \ |J_i| > 0.10 \ \wedge \ |E_i| \le 0.75$$
$$\wedge \ \big(|E_i| \le 0.05 \ \vee \ \operatorname{sign}(E_i) = \operatorname{sign}(J_i)\big)$$

Bounded residual

$$C(E_i) = \begin{cases} |E_i|, & |E_i| > 0.05, \\ 1, & \text{otherwise}, \end{cases} \qquad \Delta_i = \operatorname{sign}(J_i)\min\big(|J_i|, C(E_i)\big)$$

Final advantage

$$A_i^{\text{final}} = A_i^{\text{epi}} + E_i + \beta_k g_i \Delta_i$$

Schedule and contribution bounds

$$\beta_k = \begin{cases} 0, & k \le 11, \\ 0.25\min\big(1, (k-11)/20\big), & k > 11, \end{cases}$$

$$|\beta_k g_i \Delta_i| \le \begin{cases} 0.25|E_i|, & |E_i| > 0.05, \\ 0.25, & |E_i| \le 0.05. \end{cases}$$

Figure 7: **Selective admission and bounded residual.** $c_i = 1$ denotes a valid three-neighbor group. The two displayed gate rows form one conjunction: the second row's near-zero-or-sign-agreement test is parenthesized and joined to the first row by AND, matching Equation 16. The cap $C(E_i)$ bounds the candidate residual before scaling by β_k, where k is the one-based policy-iteration index. The displayed bounds constrain the added step-credit term; they do not guarantee policy improvement. The PPO loss does not backpropagate into JEPA.