THE FATE OF A FAILED TRAINING RUN: RECOVERY DYNAMICS AND REPLAY-BUFFER PATHOLOGY IN SOFT ACTOR-CRITIC

Anonymous authors
Paper under double-blind review

ABSTRACT

A failed reinforcement-learning run is usually reduced to one observation: its final score. We instead save the failed state and branch it repeatedly, turning post-failure recovery into a distribution over stochastic continuations. The resulting 720-run study covers 20 failed Soft Actor-Critic checkpoint branch points on two Meta-World tasks. Recovery is a multi-stage process: a first successful episode (a *spark*), optional crossing of a repeated-success marker (*consolidation*), and a final solved or unsolved evaluation; a consolidated failure is a *relapse*. No unsparked continuation solves (0/313), whereas 111/159 continuations that reach 50 successes solve. Critic-loss excursions are not terminal: 94/120 solved continuations cross the operational instability marker, and in 49/90 solved continuations containing both events the critic crosses first. Actor–critic reset raises the equal-checkpoint consolidation rate by 0.160 (95% checkpoint-blocked bootstrap interval $[0.051, 0.278]$), although its solve-rate gain is less certain and heterogeneous across checkpoints. Replay data supplies a sharper intervention. On checkpoint s18, reset with the failed run's buffer yields 0/10 solves; clearing that buffer yields 4/10; and replacing it with two compatible zero-success foreign buffers yields 9/10 and 7/10. The matched donor comparison shows that successful donor episodes and post-reset donor history are not necessary for rescue in this setting, while leaving the enabling property of the replacement data open. Finally, a checkpoint-aware stopping frontier quantifies the cost of acting on delayed recovery: an illustrative $+400$k no-spark rule saves 24.6% of post-branch compute while sacrificing 7/120 eventual solves. Replicated branches therefore expose both the dynamics of recovery and a data-state obstruction that a parameter reset alone can preserve.

1 INTRODUCTION

Training runs fail, but failure is rarely replicated. In reinforcement learning (RL), a run that reaches near-zero reward is normally abandoned, continued once, or summarized by its final return. None of these choices separates the state inherited at failure from the stochastic events that follow it. A successful restart may be luck; an unsuccessful continuation may have recovered under a different seed or after a longer delay.

This distinction matters for off-policy RL because a checkpoint contains two coupled states. Its networks and optimizer moments encode parameter state, while its replay buffer encodes data state. Scheduled resets, plasticity injection, dormant-neuron recycling, and reset-enabled replay ratios seek to prevent training collapse (Nikishin et al., 2022; 2023; Sokar et al., 2023; D'Oro et al., 2023). They do not establish whether an already failed state is recoverable, nor whether its parameters or retained experience obstruct that recovery.

We study that question by copying the same failed checkpoint into repeated continuations under a fixed intervention (Figure 1). The checkpoint holds constant the actor, critics, target networks, optimizer and entropy state, and replay data; the continuation seed supplies stochastic variation. A checkpoint–intervention cell therefore estimates an outcome distribution rather than reporting an anecdote. We apply this design to 720 completed Soft Actor-Critic (SAC) continuations at 20 failed checkpoint branch points on *pick-place-v3* and *push-v3* (Haarnoja et al., 2018a;b; Yu et al., 2020).

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.

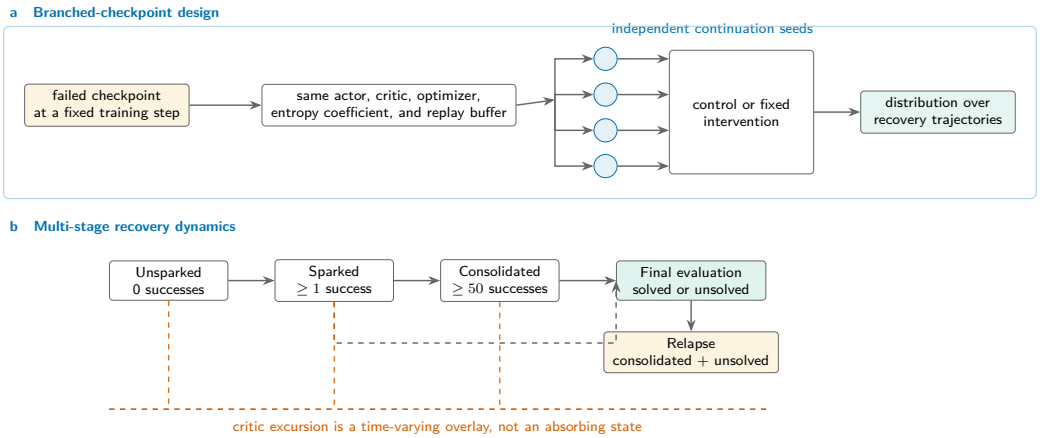

Figure 1: **Branched-checkpoint design and recovery states.** A failed checkpoint is copied into independent continuations under a fixed intervention, so each checkpoint–arm cell estimates a distribution. Recovery moves from unsparked to sparked and often to consolidated, followed by a solved or unsolved final evaluation. Some solves precede the operational consolidation marker; a consolidated failure is a relapse. A critic excursion may occur at any stage and is not an absorbing state.

The first answer is descriptive but consequential: recovery is multi-stage. A run first encounters a successful episode, may reproduce success often enough to consolidate it, and finally either solves or relapses. We call the first event a *spark*; the name identifies a first passage and does not commit us to an exploration mechanism. Likewise, a *critic excursion* is an operational critic-loss crossing, not an absorbing event. This fence is necessary because the data contradict the terminal reading: most solved continuations cross the marker, often before consolidation.

The second answer localizes an obstruction. Fresh actor and critic networks do not rescue one failed checkpoint while its own replay buffer is retained. Removing that buffer helps; replacing it with compatible foreign replay helps more. Two effective donors contain no successful episodes, and one was collected without a parameter reset. Thus neither successful donor experience nor post-reset donor history is necessary in the matched setting. The result does not yet identify the decisive property of replay content, but it changes the causal target: the failed run's data state can preserve failure through a network reset.

Our contributions are:

- a branched-checkpoint protocol and audited event table of 720 completed SAC continuations, converting post-failure outcomes into per-checkpoint distributions;
- a multi-state account of recovery in which critic instability is a time-varying overlay, together with equal-checkpoint effects, a two-stage checkpoint-blocked bootstrap, and leave-one-checkpoint-out analyses;
- a replay intervention ladder that separates retained, cleared, and foreign buffers, including a matched donor-history control; and
- a checkpoint-aware compute–recovery frontier for no-spark stopping, replacing a single retrospective threshold with an explicit operating trade-off.

2 Branched-checkpoint protocol

Training and branch points. We use SAC from Stable-Baselines3 (Raffin et al., 2021) with learning rate 3×10^{-4}, batch size 1024, replay capacity 10^6, $\gamma = 0.99$, $\tau = 0.005$, target entropy -4, and four gradient steps per environment step. Experiments use either one or four parallel environments; the early B5 batch uses one environment with four optimizer updates per collected

transition, while later batches use four environments with one update per collected transition (four gradient steps per environment step in both). Within-checkpoint control–reset pairs always share the same configuration, so the paired comparison never mixes the two regimes. Most checkpoint branches occur at 200k environment steps; one seed also contributes a 100k branch point. A checkpoint enters the study when its evaluation reward at the branch is at most 100—our operational definition of a failed checkpoint, not a latent type—and each continuation runs to one million total steps. The 20 branch points comprise 15 on pick-place and five on push. They were deliberately selected failed runs, not a random sample from training. We name a branch point by its training seed (s13, s18) and append the branch step for the one seed that contributes two (s6@100k, s6@200k).

Interventions. The principal comparison is control versus actor–critic reset. Control resumes the saved state unchanged. Actor–critic reset replaces the actor, critics, target networks, and their Adam moments with a fresh SAC initialization while retaining the replay buffer, entropy coefficient, and entropy optimizer. Other cells reset only the actor, reset the actor with temperature reinitialization, reset only the entropy optimizer, clear replay, or transplant a foreign buffer. Appendix A gives exact component-level definitions. Coverage is intentionally unbalanced because later interventions were chosen from earlier branch distributions; pooled arm rates are therefore descriptive, and Appendix C reports them in full. Five cohort cells use $n = 30$ (s13 control and actor–critic reset, s17 control, s18 control, and s20 control); the other 57 cells use $n = 10$. The five $n = 30$ cells expand replication on the primary s13 intervention pair and on the s17/s18/s20 controls to tighten the within-checkpoint comparison. These are follow-up replications within selected checkpoints, not additional independent branch points.

Operational events. Evaluations use one environment, occur every 10k steps, and average 20 deterministic episodes; the final evaluation reward is the mean of those 20 returns. A training episode counts as successful when its episode reward is at least 500, the same level used below for the final evaluation. A *spark* is the first successful training episode. *Consolidation* (*consol50*) occurs when cumulative successes reach 50; we also report the 100-success level descriptively. A *critic excursion* is the first logged per-rollout mean critic loss above 50 (SB3 emits one mean value per rollout to stdout; the marker is its first crossing), a threshold fixed before the later buffer experiments. A continuation is *solved* when its final evaluation reward is at least 500, and has *relapsed* when it consolidated but finishes unsolved. These markers make trajectories comparable; they are not latent states inferred by a model, and success events are not causal treatments.

Checkpoint-aware inference. Replication is nested within checkpoint branch points. For the reset effect, we use the 15 branch points containing both control and actor–critic reset cells. We first compute each cell's event rate, form reset-minus-control differences within checkpoint, and average checkpoints equally. A two-stage bootstrap resamples checkpoint branch points and then continuations within checkpoint–arm cells (20,000 draws; seed 20260811). We also remove each checkpoint in turn. This analysis addresses dependence among continuations from the same failed state; it does not turn the selected checkpoints into a population-random sample. For the small matched donor cell, we use an exact paired McNemar test over the ten shared recipient seeds. Appendix B states the estimator and the resampling scheme.

Frozen source of truth. All headline results come from one completion-filtered event table: 720 unique run rows joined one-to-one with 720 provenance records. The audit checks completion, identifiers, event ordering, outcome definitions, relapse consistency, and cohort coverage; it reports zero errors. Appendix F records the filename, digest, scripts, and claim-level cross-checks.

3 POST-FAILURE RECOVERY IS MULTI-STAGE

If a failed checkpoint were irrecoverable, early successes would be incidental: the continuation would fail whether or not it briefly reached a success state. The observed trajectories have a stricter structure. Across all 720 continuations, none of the 313 continuations with zero successful training episodes solves. Removing any one checkpoint leaves this fraction at zero. The converse is not deterministic. Of 407 continuations with at least one success, 120 solve (29.5%); of the 159 that reach 50 successes, 111 solve (69.8%); of the 112 that reach 100, 94 solve (83.9%). Forty-eight continuations consolidate and later finish unsolved, while nine solved continuations never reach the 50-success marker. Those

nine record only 9–38 training successes yet pass the deterministic final evaluation: several spark late (up to $+592$k post-branch), leaving too little run left to accumulate 50, and evaluation can pass while stochastic training still fails. The exception is a train–eval gap, not recovery without repetition. These counts support a progression from first passage to repetition to final evaluation, not a single recovery event.

Critic instability is not terminal. A competing terminal-event account predicts that a critic-loss crossing should exclude later recovery, or at least usually follow it. The data show the opposite. Of 120 solved continuations, 94 cross the critic-excursion marker. Among the 90 solved continuations containing both a critic excursion and consol50, 49 cross the critic marker first. Only 7 of the 94 solved continuations with an excursion cross it before their first successful episode, so the marker typically follows rather than precedes recovery, and the non-terminality claim concerns the post-reward trajectory rather than pre-spark prediction. The solved-run excursion fraction remains between 75.2% and 82.5% under leave-one-checkpoint-out deletion, and the fraction crossing before consol50 remains between 46.1% and 59.2%. Critic loss can therefore be transient, or coexist with behavior that later solves the task. We retain the marker because it describes optimization instability, but report it as a time-varying overlay rather than an absorbing state.

The same point is visible at the task boundary. Every control continuation on the five push checkpoints crosses the marker, yet 11/50 solve. On s13, by contrast, the control arm has neither excursions nor solves. A single threshold cannot mean terminal failure in both settings. Point-in-time critic loss may be useful for monitoring, but its interpretation is checkpoint- and trajectory-dependent.

Failed checkpoints carry different continuation distributions. Every branch point satisfies the same failed-checkpoint reward criterion, but their control solve rates range from 0 to 0.7. Pick-place s14 solves in 5/10 control continuations, s1 in 2/10, and s6@200k and s21 in 1/10 each, whereas s6@100k, s12, s13, s15, s17, s18, s19, s20, s22, s23, and s24 produce none in their observed control cells. Thus the checkpoint's branch reward does not determine its fate. Repeated branches reveal the quantity a single score suppresses: conditional recovery probability from the same saved state. Appendix D shows that the same branch criterion also hides very different distributions of *when* the critic marker is crossed.

4 Reset changes recovery stages heterogeneously

Actor–critic reset changes the event distribution, but not uniformly across stages or checkpoints (Figure 2). Across the 15 paired branch points, the equal-checkpoint reset-minus-control risk differences are $+0.113$ for sparking, $+0.160$ for consol50, $+0.022$ for critic excursion, and $+0.107$ for solving. Their checkpoint-blocked 95% intervals are $[-0.024, 0.267]$, $[0.051, 0.278]$, $[-0.153, 0.191]$, and $[-0.007, 0.227]$, respectively (Table 3). Consolidation is the clearest pooled effect; the solve interval narrowly includes zero, and the spark interval spans zero as well. The pooled solve effect is carried by pick-place (equal-checkpoint difference $+0.170$) with essentially no contribution from push (-0.020); the same split holds for sparking ($+0.210$ versus -0.080), while the consolidation gain is identical on both tasks ($+0.160$). All leave-one-checkpoint-out solve differences remain positive ($[0.071, 0.136]$), so no single checkpoint creates the sign, but magnitude and uncertainty remain substantial.

Checkpoint s13 shows why within-state replication is still informative. At $n = 30$, control produces 8 sparks, no consolidations, no critic excursions, and no solves. Actor–critic reset produces 27 sparks, 12 consolidations, 22 critic excursions, and 9 solves. Reset does not simply move a scalar return: it converts an inert failed state into a distribution containing both recovery and instability. The critic increase on s13 is large even though the checkpoint-averaged excursion effect is near zero, an example of the heterogeneity obscured by pooling.

Partial resets provide a local boundary. On s13, actor-only reset sparks in 6/10 continuations but yields no consolidation or solve, while actor reset with temperature reinitialization yields no spark in 10 trials. On s6@200k, resetting only the entropy optimizer yields 0/10 solves, compared with 7/10 for the full actor–critic reset. These partial-reset cells span several checkpoints (s1, s6, s12–s19) with different replication counts, so we report them as non-pooled boundary observations rather than a second pooled comparison: they do not isolate a unique parameter mechanism, but they show that a

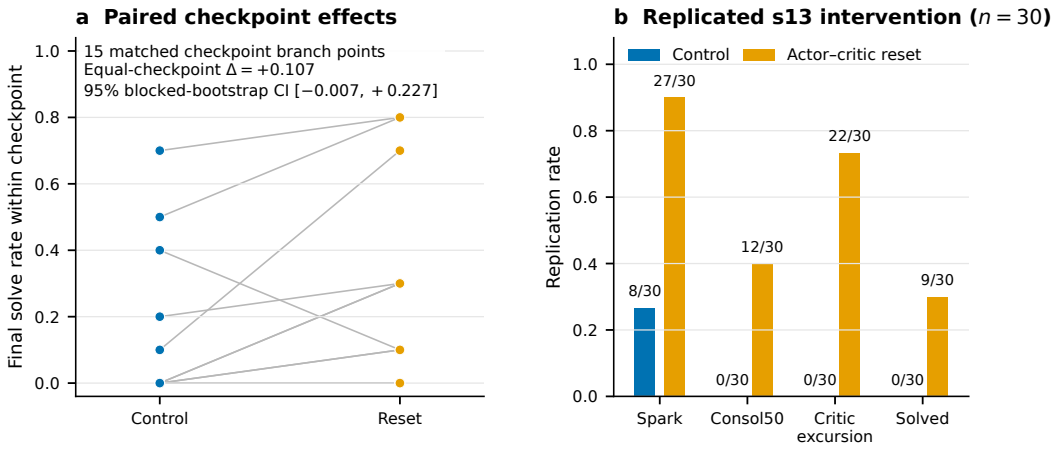

Figure 2: **Reset effects across and within checkpoints.** **a**, Solve rates at the 15 branch points containing both control and actor–critic reset continuations; lines join the same checkpoint. The displayed interval resamples checkpoints and then continuations. **b**, The replicated s13 intervention. Reset creates consolidation and solves, but also critic excursions.

first success and an optimizer-state change do not reproduce the full reset distribution. The replay experiments next show that even replacing all learned networks can leave a separate obstruction intact.

5 Replay data can carry the obstruction

Parameter reset should rescue checkpoint s18 if its actor and critic are the binding failure. It does not: fresh actor and critic networks trained with s18's retained buffer produce 0/10 solves and 10/10 critic excursions (Figure 3). The retained data state survives the parameter intervention.

5.1 An intervention ladder at s18

We next vary replay while applying the same actor–critic reset. Clearing the recipient buffer produces 4/10 solves. Replacing it with a compatible foreign buffer collected from a post-reset continuation produces 9/10 (two-sided Fisher exact $p = 0.057$ against clearing, so the donor-over-clearing gap is itself not significant at $n = 10$). That donor contains zero successful episodes at the dump point, so successful donor experience cannot be necessary for rescue. In transition units the recipient buffer holds 49,999 transitions while the matched donors hold 149,998, a threefold volume difference (both below the 10^6 capacity); the success-bearing donorC holds 659,998. A post-freeze pilot closes this volume confound (Table 2): the same donorB subsampled to recipient-matched 50k transitions with actor–critic reset solves 10/10 on s18 (minimum final evaluation 583; GPU, $n = 10$), matching the frozen full-volume 9/10. A CPU-device bridge of the same volume-matched cell solves 2/2 (final evaluations 4327 and 2646), consistent with a negligible device shift; we keep the GPU and CPU results separate in Table 2. Replacement distribution, rather than volume, is the remaining explanation supported by this comparison.

The obvious alternative is that the post-reset donor carries a special history even before it records a success. We test that account with a matched zero-success buffer from an *un-reset* continuation. Recipient checkpoint, actor–critic reset, number of parallel environments, gradient steps, update ratio, donor parent seed (s13), donor dump step (600k), donor success count, and ten recipient continuation seeds are held fixed. The donor checkpoint contents necessarily differ because donor reset history is the treatment (post-reset versus un-reset). The un-reset donor produces 7/10 solves, compared with 9/10 for the post-reset donor. Pairing recipient seeds gives three post-reset-only solves and one un-reset-only solve (two-sided exact McNemar $p = 0.625$). With ten pairs, this does not establish equivalence or exclude a moderate difference. It does rule out the proposed large binary mechanism in which post-reset donor history is required.

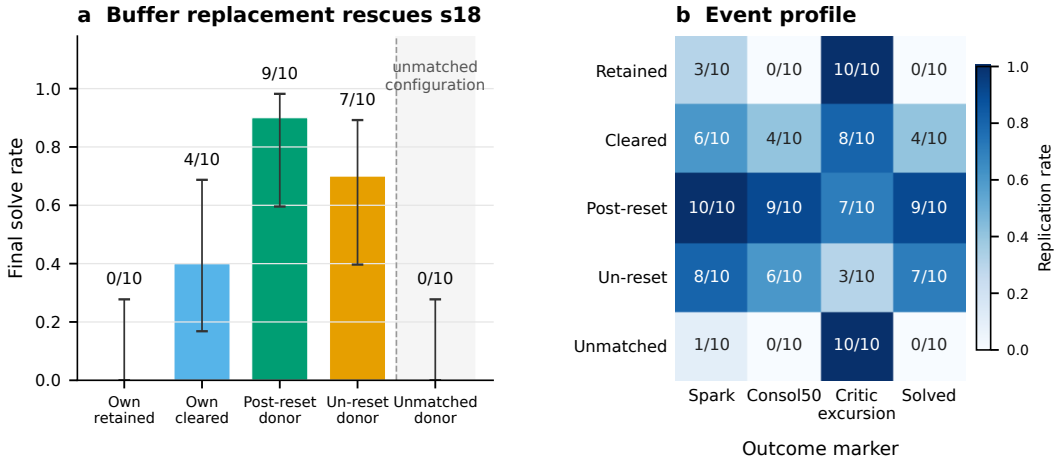

Figure 3: **Replay treatment at failed checkpoint s18. a**, Final solve rates; bars show descriptive 95% Wilson intervals over ten continuations. The first four cells use four environments and one optimizer update per collected transition. The fifth bar is the unmatched failed-checkpoint donor (one environment, four updates per collected transition, zero donor successes, and 0/10 solves). The separate success-bearing donor (70 donor successes and 3/10 solves) is reported in Table 1 but not plotted. **b**, Recovery-stage and critic-excursion rates. Compatible zero-success foreign buffers rescue with or without post-reset donor history.

A separate replay-clear cell at s13 yields 4/10 solves, compared with 9/30 for the retained actor–critic-reset cell. Because it is not part of the matched s18 ladder and has a different replication count, we report it as a non-pooled descriptive observation rather than independent confirmation of the s18 clearing effect.

The event profiles sharpen the comparison. Retained replay yields 3/10 sparks, no consolidation, and 10/10 excursions. Clearing raises these to 6/10 sparks and 4/10 consolidations. The post-reset and matched un-reset donors yield, respectively, 10/10 and 8/10 sparks, 9/10 and 6/10 consolidations, and 7/10 and 3/10 critic excursions. Rescue is therefore not a side effect of making the critic marker quiet: the stronger donor condition still crosses it in most continuations.

5.2 What the ladder establishes—and what it does not

The ordered interventions support a narrow causal statement: on s18, the recipient's retained replay buffer can prevent fresh networks from recovering; removing it helps, and compatible replacement data helps more. The experiments separately show that successful donor episodes and post-reset donor history are not necessary. They do not identify which distributional property makes a foreign buffer compatible.

Clearing also carries a refill-shock confound: the cleared cell restarts from a single retained transition with the warm-up period long elapsed, so full batch-1024 updates apply immediately to near-duplicate data. A post-freeze gated-clear pilot (warm-up 1024; GPU, $n = 10$) tests this alternative: it solves 7/10 against the frozen ungated 4/10. This shows that refill handling matters, but the two cells do not isolate its causal contribution; both remain below the volume-matched 10/10. The post-freeze s18 dose-response over retained recipient fraction is 0/10 at 100% (frozen), 0/10 at 50% (CPU), 0/10 at 25% (CPU), 4/10 frozen (7/10 gated post-freeze) at 0% cleared, and 9–10/10 with replaced volume-matched or full donor data; the only near-miss under retention is a consolidated relapse (41 successes, final evaluation 245) at 50%. The 50% and 25% cells are both 0/10, so this small ladder is flat over those two fractions and does not locate a retention threshold; replacement rescues at matched volume.

One negative cell illustrates the remaining confounding. A zero-success donor taken directly from another un-reset failed checkpoint yields 0/10 solves, but that cell uses a different donor identity, one rather than four parallel environments, and four rather than one updates per collected transition.

It is evidence of configuration sensitivity, not a clean provenance effect. A second unmatched donor containing 70 successful episodes yields only 3/10 solves on s18. Donor success count is therefore neither required nor an adequate scalar description of transfer, but the present design cannot distinguish state–action coverage, recency, reward composition, temporal correlations, or their interaction with update ratio.

5.3 THE REPLACEMENT EFFECT RECURS ACROSS RECIPIENTS

The compatible post-reset zero-success donor was also transplanted into s12 and s15, two checkpoints that never consolidate under control. It yields 8/10 and 9/10 solves, respectively, compared with 0/10 control and 0/10 or 1/10 actor–critic-reset solves with retained replay. Across s12, s15, and s18, the same donor condition solves 26/30 continuations. On s23, a post-freeze pilot replaces the mismatched donorC transplant with the same compatible full donorB under the matched ladder configuration (four environments, one update per collected transition; GPU, $n = 10$): it solves 9/10, with only rep0 unsolved (54 successes, final evaluation 7; Table 2). The frozen s23 0/10 (one consolidated relapse) therefore reads as a configuration mismatch—one environment and four updates per transition with a success-bearing donor—rather than a recipient boundary. Four responsive recipients establish recurrence across s12, s15, s18, and s23; coverage remains four checkpoints, so no population-wide claim follows.

6 STOPPING IS A COMPUTE–RECOVERY TRADE-OFF

The zero-spark association is cheap to monitor, but it does not select its own stopping time. We therefore evaluate six policies: stop a continuation at t steps after branching if it has not yet recorded a successful episode; otherwise let it finish. Figure 4 reports post-branch compute saved against the fraction of eventual solves sacrificed. We apply the policy retrospectively to every completed continuation and use the same two-stage checkpoint bootstrap for uncertainty.

Earlier stopping saves more compute by discarding more delayed recoveries. At $+50$k, the policy saves 78.0% of post-branch compute but loses 73/120 solves (60.8%). At $+200$k, it saves 45.2% and loses 28/120 (23.3%). An illustrative $+400$k point saves 24.6% (95% checkpoint-blocked interval $[18.1, 29.9]$) and loses 7/120 solves, or 5.8% ($[0.8, 12.8]$). Waiting to $+600$k sacrifices no observed solve and saves 11.5% ($[8.4, 14.1]$). Table 5 in Appendix E lists all six operating points with their intervals.

The curve changes the practical claim. There is no untuned $+400$k law: the threshold is a choice made after observing this dataset, and its desirability depends on the cost of a missed recovery relative to training compute. Disaggregated, the illustrative $+400$k point saves 30.2% of control compute while losing 1/20 control solves, against 14.5% saved and 4/41 lost under reset: pooled savings mix arms with different recovery timing. Because the threshold was selected on these same 120 solves, both numbers are in-sample, and the $+600$k zero-loss column is an observed zero with a degenerate interval rather than a guarantee. The zero-spark result supplies a monitor and this study supplies an empirical frontier. A deployment should choose an operating point on held-out checkpoints under its own exchange rate, recalibrating the threshold conditionally on its own arm mixture rather than adopting $+400$k.

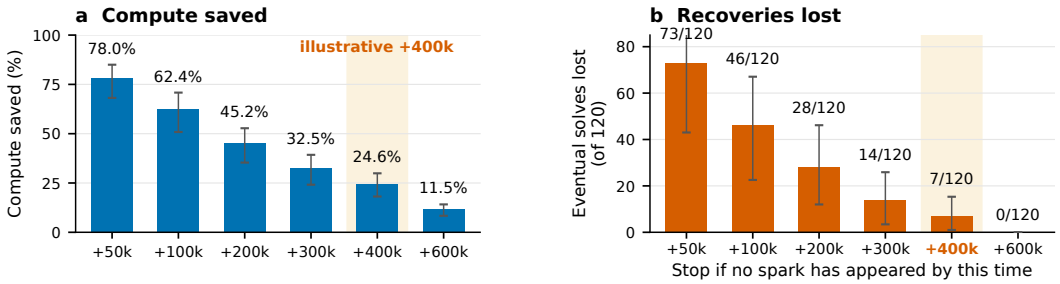

Figure 4: **No-spark stopping frontier.** Each policy stops a continuation that has recorded no successful episode by the labeled post-branch time. Whiskers are 95% checkpoint-blocked intervals. The shaded column marks an illustrative operating point, not a universal threshold.

7 Related Work

Resets and plasticity. Primacy bias, dormant neurons, and plasticity loss motivate resetting or regularizing agents during training (Nikishin et al., 2022; Sokar et al., 2023; Nikishin et al., 2023; Dohare et al., 2024; Klein et al., 2024). Optimizer resets isolate the role of accumulated moments (Asadi et al., 2023), while selective reset methods seek to retain useful parameters (Falzari & Sabatelli, 2025). These methods are principally preventive or scheduled. We intervene after an operationally failed checkpoint and replicate the same starting state, which exposes an outcome distribution and allows retained parameter and data state to be separated.

Replay and high update ratios. High replay ratios can improve sample efficiency while amplifying value divergence, overestimation, overfitting, and loss of plasticity (D'Oro et al., 2023; Hussing et al., 2024; Nauman et al., 2024). Replay-buffer design also changes the empirical training distribution. Our result is complementary: after failure has occurred, resetting networks while keeping the same replay may preserve the obstruction, whereas clearing or compatible replacement can restore recovery. The experiment localizes a data state but does not yet identify its sufficient statistic.

Checkpoint reuse and instability measurement. Reincarnating RL reuses trained agents to accelerate subsequent learning (Agarwal et al., 2022); we reuse a failed checkpoint as an experimental control. Work on training instability often compares independent runs or injects perturbations into otherwise healthy training (Zhang et al., 2026). The branched unit instead holds the pre-branch state fixed and estimates post-failure event rates. This design, rather than a new RL algorithm, is the paper's methodological contribution.

8 Scope and Conclusion

This study uses SAC, two tasks from one benchmark family, continuations that run to one million total steps, and 20 deliberately selected failed checkpoint branch points. The main interventions have dense within-cell replication but sparse and adaptive coverage across checkpoints. Checkpoint-blocked inference accounts for nested continuations; it cannot support a claim about all SAC failures or other algorithms. A hierarchical logistic model would add stronger modeling assumptions under only 20 clusters, sparse arm coverage, and complete separation for zero-spark solving, so we use blocked resampling and leave-one-checkpoint-out sensitivity instead.

The operational markers also have limits. Spark and consolidation describe observed success counts, not exploration or causal phases. The critic threshold was fixed prospectively for later batches but is scale-dependent and, as the results show, not a failure certificate. The solve threshold is a final evaluation convention. The stopping frontier is retrospective and requires validation on unseen checkpoints before deployment.

The buffer result is causal for the treatment ladder on s18 and recurs under the compatible zero-success donor on s12, s15, and the post-freeze s23 pilot. The post-freeze cells also close the volume comparison and sample 25% and 50% retention, but they are small, do not log the full recovery

markers, and sit outside the frozen 720-row table. They therefore narrow the confounds without identifying the replay statistic that matters or establishing a population-level threshold. A decisive next experiment should hold recipient, donor, environment count, update ratio, and continuation seeds fixed while varying a prespecified buffer property and validating the resulting ordering on held-out failed checkpoints.

Within these bounds, the branched view changes what a failed run is. It is not a scalar ending but a distribution over continuations. That distribution separates first passage from consolidation, prevents transient critic instability from being mislabeled as terminal, and shows that failure can be carried by retained replay data even after the networks are replaced.

AI USE STATEMENT

In this work, we used generative AI tools for proposing and refining hypotheses, designing and providing feedback on research methodology and experiments, implementing methods, and interpreting results. Specifically, they helped choose which follow-up interventions to run and in what order, implement the intervention and event-extraction code, audit experiment bookkeeping, run the analyses, and propose readings of the results. We have not used generative AI tools for generating synthetic data sets, cleaning and reformatting the dataset, supporting qualitative and thematic data analysis, or assisting with translation, and developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools for suggesting experimental parameters, creating and editing software code, creating and modifying scientific figures, creation of artifacts, drafting parts of the research paper, editing the paper to improve readability, and suggesting a structure for the paper. No data were synthesized: every number reported here comes from a completed training run. We have reviewed all AI-assisted work: the authors set the research question and the study design, inspected the underlying run artifacts, verified the frozen event table and every reported statistic against it, and checked each citation against its source. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The anonymous supplement contains the frozen 720-row event table and its SHA-256 digest, a row-level provenance table, cohort manifest, audit report, checkpoint-aware analysis and tests, stopping-policy outputs, and figure generation sources. The post-freeze rebuttal pilot cells are reported separately in Appendix F and do not alter the frozen estimates. Appendix F maps the reported claims to these artifacts and records the intervention implementation.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Reincarnating reinforcement learning: Reusing prior computation to accelerate progress. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Kavosh Asadi, Rasool Fakoor, and Shoham Sabach. Resetting the optimizer in deep RL: An empirical study. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632: 768–774, 2024. doi: 10.1038/s41586-024-07711-7.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G. Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Massimiliano Falzari and Matthia Sabatelli. Fisher-guided selective forgetting: Mitigating The primacy bias in deep reinforcement learning. *arXiv preprint arXiv:2502.00802*, 2025.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Marcel Hussing, Claas A. Voelcker, Igor Gilitschenski, Amir-massoud Farahmand, and Eric Eaton. Dissecting deep RL with high update ratios: Combatting value divergence. *Reinforcement Learning Journal*, 2:995–1018, 2024.

Timo Klein, Christoph Luther, Manus McAuliffe, Lukas Miklautz, Claudia Plant, and Sebastian Tschiatschek. Plasticity loss in deep reinforcement learning: A survey. *arXiv preprint arXiv:2411.04832*, 2024.

Michal Nauman, Michał Bortkiewicz, Piotr Miłoś, Tomasz Trzciński, Mateusz Ostaszewski, and Marek Cygan. Overestimation, overfitting, and plasticity in actor-critic: the bitter lesson of reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 37342–37364. PMLR, 2024.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16828–16847. PMLR, 2022.

Evgenii Nikishin, Junhyuk Oh, Georg Ostrovski, Clare Lyle, Razvan Pascanu, Will Dabney, and André Barreto. Deep reinforcement learning with plasticity injection. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `https://jmlr.org/papers/v22/20-1364.html`.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32145–32168. PMLR, 2023.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020.

Zhipeng Zhang, Zhenjie Yao, Kai Li, and Lei Yang. Training instability in deep learning follows low-dimensional dynamical principles. *arXiv preprint arXiv:2601.13160*, 2026.

A Intervention implementation

Every checkpoint stores actor and critic networks, target networks, the replay buffer, entropy coefficient α (represented by $\log \alpha$), and the Adam state associated with each optimized component. Components not named by an intervention remain exactly as saved.

Actor–critic reset. Instantiate a fresh SAC model with identical hyperparameters. Copy the fresh actor weights and Adam moments, critic weights and Adam moments, and target critic networks into the branching model. Retain the replay buffer, $\log \alpha$, and the entropy optimizer.

Partial reset arms. Actor-only reset replaces the actor weights and its Adam state. Actor reset with temperature reinitialization additionally sets $\log \alpha \leftarrow \log(0.2)$ but does not reset the critic or entropy optimizer. Optimizer-only reset reinitializes the Adam state backing $\log \alpha$ while keeping all network weights, the value of $\log \alpha$, and replay unchanged.

Replay clear. Apply actor–critic reset and replace the retained buffer with the minimum buffer permitted by the training loop. The implementation's `keep_replay_fraction=0.0` retains one transition because at least one stored item is required before batch sampling. This is an implementation minimum, not a claim that one transition is optimal.

Replay transplant. Apply actor–critic reset and replace the recipient buffer in full with a buffer dumped from a separate donor continuation. No recipient transition is retained. The buffer's vectorized-environment shape must agree with the recipient, which is why the main matched comparison fixes parallel environment count. Table 1 records the donor conditions used at s18.

Table 1: **s18 replay-intervention cells.** U/T is optimizer updates per collected transition. All cells apply actor–critic reset. The first four are matched on four environments and U/T $= 1$; the final two are deliberately reported as unmatched configurations.

Replay treatment	Donor history	Donor successes	Envs	U/T	Solved
Recipient retained	recipient failure	–	4	1	0/10
Recipient cleared	–	–	4	1	4/10
Foreign donor	post-reset continuation	0	4	1	9/10
Foreign donor	un-reset continuation	0	4	1	7/10
Foreign donor	post-reset continuation	70	1	4	3/10
Foreign donor	un-reset failed checkpoint	0	1	4	0/10

Table 2: **Post-freeze rebuttal replay cells.** All cells apply actor–critic reset and sit outside the frozen 720-row table, whose SHA-256 digest is unchanged. Donor transitions are 49,999 (recipient-matched), 149,998 (matched full donor), or 659,998 (donorC, frozen reference only). Spark and consolidation markers were not logged for these pilots, so only solve counts are reported. GPU cells use $n = 10$; the CPU bridge uses $n = 2$.

Cell	Recipient	Replay treatment	Donor trans.	Device	n	Solved
B1 volmatch	s18	donorB-down50k	49,999	GPU	10	10/10
B1 bridge	s18	donorB-down50k	49,999	CPU	2	2/2
B2 gated clear	s18	cleared, warm-up 1024	–	GPU	10	7/10
B3 compat. donor	s23	full donorB	149,998	GPU	10	9/10
B5 half	s18	50% recipient retained	–	CPU	10	0/10
B5 quarter	s18	25% recipient retained	–	CPU	10	0/10

B CHECKPOINT-AWARE STATISTICAL DETAILS

Let r_{ca} be the empirical event rate for checkpoint branch point c and arm a. For an event Y, the paired equal-checkpoint reset effect is

$$\widehat{\Delta}_Y = \frac{1}{C} \sum_{c=1}^{C} \left(r_{c,\text{reset}} - r_{c,\text{control}} \right), \qquad C = 15.$$

The bootstrap samples C branch points with replacement. Within each sampled branch point, it independently samples with replacement the observed continuations in each of the two cells, recomputes their rates, and then recomputes $\widehat{\Delta}_Y$. Percentile intervals use 20,000 draws with seed 20260811. Leave-one-checkpoint-out (LOCO) ranges repeat the point calculation after deleting each c. The estimand weights observed checkpoint branch points equally; it is not a replication-weighted pooled rate.

Table 3: **Paired actor–critic reset effect.** Risk differences are reset minus control across 15 checkpoint branch points.

Event	Equal-checkpoint difference	95% blocked CI	LOCO range
Spark	$+0.113$	$[-0.024, +0.267]$	$[+0.076, +0.136]$
Consol50	$+0.160$	$[+0.051, +0.278]$	$[+0.136, +0.179]$
Critic excursion	$+0.022$	$[-0.153, +0.191]$	$[-0.029, +0.081]$
Solved	$+0.107$	$[-0.007, +0.227]$	$[+0.071, +0.136]$

A hierarchical logistic model was considered but is not the primary analysis. There are only 20 checkpoint branch points, arm coverage is sparse, and the zero-spark outcome is completely separated. Under these conditions, prior and link choices would contribute more apparent precision than the design supports. Blocked resampling makes the checkpoint unit explicit with fewer modeling choices.

The exact McNemar calculation for the matched s18 donor comparison conditions on discordant pairs. Of ten shared recipient seeds, three solve only with the post-reset donor and one only with the un-reset donor. Under a null probability of $1/2$ for either direction, the two-sided exact probability is $2\Pr\{\mathrm{Binomial}(4, 1/2) \leq 1\} = 0.625$.

C Full pooled event counts

Table 4 reports replication-level counts to make the released table easy to cross-check. These rows should not be interpreted as a balanced comparison: arms cover different checkpoint sets, and later arms target earlier observed failures.

Table 4: **Pooled counts by intervention.** Critic excursion is an overlay and may coincide with every other outcome. Relapse means consol50 followed by an unsolved final evaluation.

Arm	n	Spark	Consol50	Critic	Solved	Relapse
Control	280	129	27	186	20	12
Actor–critic reset	170	133	59	139	41	19
Actor-only reset	50	31	8	26	4	4
Actor reset + temperature reinit.	100	37	10	63	7	4
Entropy-optimizer reset	30	18	7	17	4	3
Replay clear + reset	20	13	8	17	8	0
Foreign donor (success-bearing)	20	7	5	20	3	2
Foreign zero-success post-reset donor	30	30	29	23	26	3
Unmatched failed-checkpoint donor	10	1	0	10	0	0
Matched zero-success un-reset donor	10	8	6	3	7	1
All	720	407	159	504	120	48

The state counts form the intended partition. Of 720 continuations, 313 never spark and 407 spark. Of the latter, 159 reach consol50. Among the consolidated set, 111 solve and 48 relapse; another nine solve without reaching consol50. Critic excursion is not part of this partition: it occurs in 504 trajectories, including 94 solved ones.

D Additional temporal heterogeneity

Replicated branches also expose checkpoint-specific excursion timing. Under control, pick-place s17 crosses the critic marker in 29/30 continuations. Twenty cross within 60k–76k post-branch steps; the median is 70k, the interquartile range is 16k, and a minority tail extends to 692k. At s18, 19/30 cross over a much wider 168k–726k range; the median is 348k and the interquartile range is 264k. These are descriptive conditional distributions, not proposed failure types. Their purpose is to show that identical point-in-time branch criteria can conceal different time-to-event structure.

The entropy coefficient does not repair the terminal interpretation. Its branch value is low at both recoverable and unrecoverable checkpoints, and within excursion trajectories it can rise after critic

loss has already begun to increase. We therefore do not use α or critic-loss time as a prospective checkpoint classifier in the main analysis.

E COMPLETE STOPPING FRONTIER

For a threshold t, a run is stopped at branch$+t$ only if its spark step is absent or later than that point. Saved compute is the removed portion of the observed post-branch budget, divided by total observed post-branch compute. Solve loss is measured against all 120 eventual solves, not only stopped runs.

Table 5: **No-spark stopping frontier with checkpoint-blocked 95% intervals.**

Stop if no spark by	Post-branch compute saved	Eventual solves lost
$+50$k	78.0% [68.1, 85.0]	73/120; 60.8% [35.8, 81.3]
$+100$k	62.4% [50.9, 70.8]	46/120; 38.3% [18.8, 55.9]
$+200$k	45.2% [35.3, 52.8]	28/120; 23.3% [10.0, 38.5]
$+300$k	32.5% [24.1, 39.3]	14/120; 11.7% [2.9, 21.6]
$+400$k	24.6% [18.1, 29.9]	7/120; 5.8% [0.8, 12.8]
$+600$k	11.5% [8.4, 14.1]	0/120; 0.0% [0.0, 0.0]

F FROZEN ARTIFACTS AND CLAIM REPRODUCTION

The anonymous package contains the frozen 720-row event table, row-level provenance, a 62-cohort manifest, and the audit report. It also contains the tested checkpoint-aware analysis and figure-generation code. Together these scripts reproduce the paired effects, blocked bootstrap, LOCO analyses, donor pairing, stopping frontier, and every quantitative figure.

The audit verifies 720 unique run keys, a one-to-one provenance join, event ordering, completed outcomes, and all marker definitions. It also checks the headline identities used in the paper: 0/313 zero-spark solves, 94/120 solved runs with a critic excursion, 49/90 critic-first solved runs containing both events, nine solved runs without consol50, the $+400$k stopping result, and the 7/10 matched un-reset donor result. All frozen-data checks pass. The post-freeze pilot-cell counts in Table 2 are summarized independently in the artifact's rebuttal-pilot CSV; their raw training logs are not part of the frozen package. Exact software versions, commands, and the dataset digest are included in the artifact README. The frozen CSV's SHA-256 digest is `5034e20a268c23b7cb23fec10c98ae97f205337e8fe2234b269057840fb28fd7`.