BUDGET ALLOCATION FOR PPO TEXT-GAME AGENTS: LONG RUNS, CANDIDATE SELECTION, AND SUCCESSIVE HALVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement-learning post-training of language-model agents leaves a practical choice: spend a delivery budget on one long run, or train several shorter candidates and select one by validation. More candidates provide selection opportunities, but repeated validation and operational costs reduce the training available to each. The delivered model's score alone cannot distinguish a weak candidate set from a poor pick. We study this choice under a common reference budget for training, validation and specified operational costs, holding the model and PPO recipe fixed within each instance. Sealing validation choices before independently evaluating every fixed-splitting candidate lets us measure the empirical best-candidate gap and selection shortfall separately. For two small language models trained as text-game agents, one long run outperforms validation selection among four shorter runs by more than ten percentage points. Even the empirical best of each four-candidate set on the independent pool stays below its paired long run; a registered held-out check confirms the delivered deficit. Continuing a selected candidate after one halving recovers much of the deficit, and a doubled budget improves two-candidate splitting's relative performance. At the reference budget, selection gains over the average candidate do not compensate for four-way splitting: the quality of the available candidates also matters.

1 INTRODUCTION

Reinforcement learning adapts language models to interactive tasks by training them on the consequences of their actions (Carta et al., 2023; Wang & Ammanabrolu, 2025; Chen et al., 2025). Producing a useful agent requires deciding both how to train it and which trained model to deliver. With a fixed budget and a chosen training recipe, should one continue a single run for longer, or train several shorter candidates from the same checkpoint and select one by validation? This paper studies that decision for language-model agents trained with proximal policy optimization (PPO; Schulman et al., 2017).

More candidates and more training offer different opportunities for improving the delivered agent. Variation across seeds in deep RL and language-model fine-tuning makes selection potentially valuable (Henderson et al., 2018; Dodge et al., 2020). Yet each additional candidate consumes budget for training, validation and repeated operational costs, leaving fewer updates for each trajectory. Extending one trajectory is not guaranteed to help either: multi-turn agent training can be unstable (Wang et al., 2025; 2026; Kim et al., 2026), and entropy collapse can limit further exploration (Petrenko et al., 2026). The outcome therefore depends on the quality of the candidates produced by each allocation as well as on the ability to select among them.

Recent language-model RL research examines compute scaling (Khatri et al., 2026), sampling and update allocation within a run (Cheng et al., 2026), allocation across model size, search, learning and feedback (Wilhelm & Kao, 2026), and search across training configurations and strategies (Chen et al., 2026; Fang et al., 2026). More broadly, Hyperband and bet-and-run formalize the trade-off between candidate count and resources per candidate (Li et al., 2018; Lissovoi et al., 2017). Supervised language-model fine-tuning already uses multiple starts, early stopping and continuation under

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.

a fixed training budget (Dodge et al., 2020), while gradient-descent studies examine run lengths and restart counts for target accuracy and affordable compute (Poli & Yilmaz, 2026). These results establish the allocation problem, but do not directly answer the delivery question studied here: with model, supervised start and PPO recipe held fixed, how does validation selection among shorter independent candidates compare with one longer run after charging both training and the costs of producing and choosing the model?

Answering this question requires measuring more than the selected model's score. Validation selection can overstate the winner's performance (Smith & Winkler, 2006; Cawley & Talbot, 2010; de la Peña et al., 2026), and evaluation noise complicates comparisons between language-model training methods (Hochlehnert et al., 2025). Independent evaluation of the winner measures delivery performance, but does not reveal whether a deficit could have been avoided by selecting another candidate from the same set. We therefore evaluate every fixed-splitting candidate after sealing the validation choice. Comparing the set's empirical best member with the long run, and the validation pick with that best member, separates the observed candidate-set deficit from the empirical selection shortfall. This asks whether better selection alone could close the observed gap.

To account for repeated per-candidate costs, we compare allocations under a common reference delivery budget that charges training, validation and specified operational costs. Following the principle of an explicit benchmark cost boundary (Dahl et al., 2023), we calibrate those costs and fix batch counts before the runs they govern. More candidates pay repeated startup, validation and checkpoint costs, so equal reference budgets need not buy equal total training. The resulting policies are compared on independent-game success rates; actual elapsed time and expenditure are reported separately. This accounting defines the decision faced by the practitioner, while the full candidate evaluation explains where the delivery difference lies.

We study Qwen2.5-1.5B and SmolLM2-1.7B under one PPO text-game recipe, with a common supervised checkpoint within each instance. To assess run-to-run variability, we compare long runs and fixed splitting across six paired training blocks and examine sensitivity to single-block deletion. A registered reserved-pool check tests whether the main delivery result persists on new games. Successive halving tests whether screening and continuation reduce the deficit from fixed splitting; a doubled budget tests sensitivity to available resources. We make three contributions:

1. We compare one long PPO run with validation selection among shorter independent candidates at a common reference delivery budget B. In both text-game model instances, fixed splitting into two or four candidates lowers delivered performance, by more than ten percentage points with four candidates; the registered reserved-pool check confirms the four-candidate deficit (Section 4.1).
2. We independently evaluate every fixed-splitting candidate and show that the same-pool empirical best of four stays below the paired long run in every block. Validation selection adds a smaller shortfall, locating most of the observed deficit in the available candidate sets (Section 4.2).
3. We show that continuation and budget size change this comparison: one successive-halving allocation recovers much of the four-way deficit, and doubling the budget improves two-candidate splitting's relative performance in both instances (Sections 4.3 and 4.4).

Scope. The evidence concerns two model instances, one PPO recipe and one text-game family. Section 2 relates the delivery question to prior work; Section 3 specifies the protocol and cost rule; Section 4 reports delivery, candidate-set and budget comparisons, with protocol exceptions in Section 4.5; Section 5 discusses implications for training and selecting language-model agents and the limits of inference beyond these runs.

2 Related work

Language-model reinforcement learning and compute. Online PPO grounding and multi-turn training recipes adapt language models to interactive tasks (Carta et al., 2023; Wang & Ammanabrolu, 2025), including text environments such as TextWorld and ALFWorld (Côté et al., 2018; Shridhar et al., 2021). Training dynamics, long-horizon credit assignment and entropy collapse affect the benefit of additional updates (Wang et al., 2025; 2026; Kim et al., 2026; Petrenko et al.,

2026). Khatri et al. (2026) study RL compute scaling; Cheng et al. (2026) allocate sampling compute among rollouts per problem, problems per batch and update steps, while Wilhelm & Kao (2026) examine model size, rollout search, learning and reward feedback under a fixed compute budget. These studies concern how a policy is trained. We examine how many independent candidates to train with a fixed recipe before delivering one. Chen et al. (2025) report best-run performance beside a three-run mean; our paired long-run comparison asks whether generating extra candidates is worth the training each gives up under a common reference budget.

Search across RL training configurations. Joint Fidelity Hyperparameter Optimization (JF-HPO; Chen et al., 2026) searches configurations under a total time budget, jointly varying model size and training-step fidelity with early stopping and checkpoint reuse. LLMZero (Fang et al., 2026) searches multi-stage RL training strategies using checkpoint continuation, configuration changes and early stopping. Both pursue efficient search over training configurations and trajectories. Our comparison holds the model and PPO recipe fixed within each instance and measures the delivery trade-off between one longer run and validation selection among independent shorter runs, including evaluation of every fixed-splitting candidate. In offline-to-online RL, Bozkurt et al. (2026) jointly select and fine-tune policies under an interaction budget charging training and evaluation. Their candidates span offline algorithms and hyperparameters; our candidates share a supervised start and one PPO recipe.

Restarts and stochastic fine-tuning. Multi-start search, successive halving and Hyperband formalize resource allocation across candidates (György & Kocsis, 2011; Jamieson & Talwalkar, 2016; Li et al., 2018); bet-and-run explicitly connects a single long run, multiple shorter runs and continuation of selected runs (Lissovoi et al., 2017). Dodge et al. (2020) study seed variation in supervised BERT fine-tuning and allocate a fixed training budget through multiple starts, early stopping and continuation. Poli & Yilmaz (2026) jointly analyze run length and restart count for reaching target accuracy, and also select models by achievable accuracy at a given gradient-step budget. These establish the trade-off between training depth and candidate selection; we study its delivery outcome for PPO text-game agents with explicit selection and operational costs. Our successive-halving arm uses this established family of methods to test whether screening and continuation improve on fixed splitting.

Evaluation and selection. Validation selection can make the winning score optimistic (Smith & Winkler, 2006; Cawley & Talbot, 2010; de la Peña et al., 2026); evaluating more policies can also change algorithm rankings (Kurenkov & Kolesnikov, 2022). Konyushkova et al. (2021) allocate an evaluation budget across given policies and measure simple regret relative to the best policy's value. We instead generate candidates within a reference delivery budget and independently evaluate the fixed-splitting sets. The empirical best-candidate gap and selection shortfall distinguish the observed set's deficit from the improvement available by changing the selected candidate. The finite-pool maximum is an optimistic retrospective diagnostic, not a known policy value. Budget-dependent reporting (Dodge et al., 2019; Tang et al., 2020) and empirical RL uncertainty (Agarwal et al., 2021; Patterson et al., 2024) motivate reporting paired observed contrasts, game-sampling intervals and training-block variation separately. Weight averaging combines trained candidates (Wortsman et al., 2022); our policies select or continue one candidate.

3 Experimental design and budget accounting

3.1 Setup and delivery policies

The design compares the models the allocation policies deliver under a common accounting rule: candidates start from a common supervised checkpoint with batch counts fixed in advance by the rule in Section 3.2, validation selects the delivered candidate before independent evaluation, independent evaluation covers the delivered model and every fixed-splitting candidate, and a reserved pool provides the registered confirmation.

We train PPO text-game agents on 64 TextWorld games (Côté et al., 2018), with reference solutions of eight actions and an episode cap of 24 actions. M1 uses Qwen2.5-1.5B-Instruct (Qwen Team, 2024); M2 uses SmolLM2-1.7B-Instruct (Ben Allal et al., 2025). Each starts from supervised train-

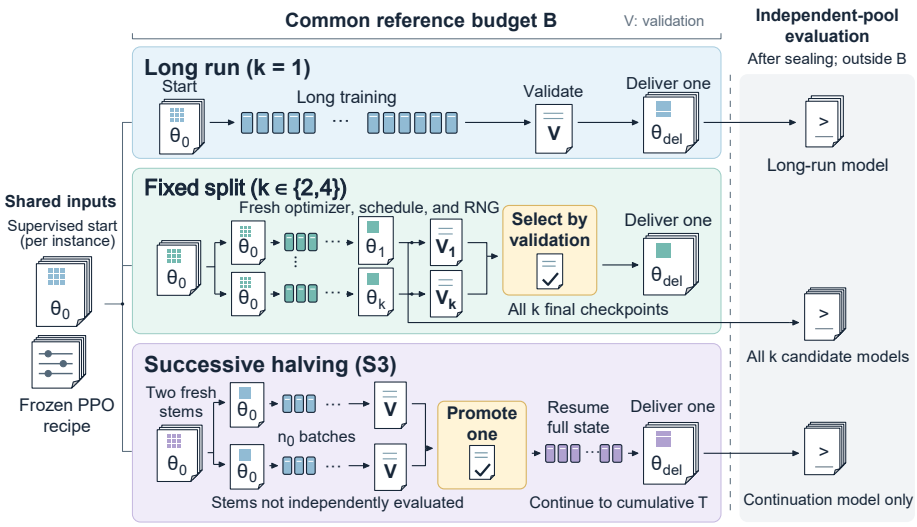

Figure 1: **Delivery policies under a common reference budget.** Fixed splitting selects among final checkpoints; S3 promotes a stem and resumes its full training state. Independent evaluation follows sealing and is outside B: all fixed-splitting candidates, but only the S3 continuation, are evaluated. Tiles are not to scale.

ing on the same isolated demonstrations, disjoint from all PPO and evaluation games. We use six seeded blocks per instance and report the instances separately. Each batch contains eight distinct games from a seeded cycle through the training pool. PPO uses one epoch per batch, a terminal 0/1 reward with a separate KL penalty, and two initial critic-only batches; complete hyperparameters are in Appendix H.

Figure 1 lays the delivery policies out side by side under the same reference budget B. The long run ($k = 1$) trains a single candidate to the end of its budget, validates it and delivers it. Fixed splitting ($k \in \{2, 4\}$) starts k candidates with fresh optimizer, schedule and random state, trains each for its own share of the budget, validates all k final checkpoints and delivers the validation winner. Successive halving (S3) trains two stems for n_0 batches, promotes the stem with the better validation count and resumes its full training state to the cumulative horizon T. Validation lies inside the budget; independent-pool evaluation follows sealing, lies outside it, and covers the long-run model, every fixed-splitting candidate and the S3 continuation only, which is what makes the candidate-set diagnostic of Section 4.2 possible.

Each fixed-splitting candidate's cosine schedule spans its own allocated batch count, and only its final checkpoint can be selected. The PPO recipe (Schulman et al., 2017) uses GAE (Schulman et al., 2016) and runs on verl (Sheng et al., 2025), following Wang & Ammanabrolu (2025).

Resuming the promoted stem restores its parameters, optimizer, RNG and absolute batch order. Each stem's schedule spans T from the start; continuation does not repeat the initial critic-only updates.

Three disjoint pools each contain 100 games with four repeats. Validation selects the candidate with the most successes, breaking ties by lowest index; its identity is sealed before independent evaluation. The reserved pool is opened once for the registered comparison and descriptive companion results.

3.2 Reference budget and executed allocations

The budget rule converts a common reference budget into policy-specific batch counts. It specifies which selection and operational costs are charged alongside training, and thereby determines every executed allocation.

Table 1: **Executed allocations and evaluation coverage.** Counts are batches per candidate. $2 \times n_0 \to T$ denotes two stems and one continuation to cumulative T. Each executed cell has six blocks; all candidates use final-checkpoint validation, except S3, which validates stems only. Delivery means the selected model only. Parent short runs and supervised starts are auxiliary controls.

Budget	Policy	M1 batches	M2 batches	Independent	Reserved
B	Long run	270	242	Delivery	Delivery
	Fixed $k = 2$	117	103	All 2	Delivery
	Fixed $k = 4$	40	33	All 4	Delivery
	S3	$2 \times 58 \to 155$	$2 \times 51 \to 140$	Continuation	Continuation
$2B$	Long run	635	564	Delivery	M2 only
	Fixed $k = 2$	299	264	All 2	M2 delivery
	Fixed $k = 4$	Not run	114	M2: all 4	M2 delivery
	S3	Not run	Not run	Not applicable	Not applicable

All policies use single-delivery accounting: each candidate pays training, validation and per-run overhead, but only the delivered model pays recovery. For k fixed runs,

$$N(k) = \left\lfloor \frac{(B - F)/k - c_1 - q}{u} \right\rfloor, \qquad q = I + S + R + d,$$

where u is calibrated time per batch, c_1 validation, I startup, S save, R reload verification, d deployment share and F delivery recovery. At $B = 3600$ s, this permits $k = 4$ but not $k = 8$ with the registered 16-batch minimum. Calibration, alternative accounting and physical costs are in Appendices A and B.

Table 1 lists the executed allocations. S3 charges both stems and the continuation's resume and delivery, so its total training is $n_0 + T$ batches. Counts are frozen before the stage they govern; S3's were fixed after measuring resume cost, before any S3 run. The B long runs are reused unchanged from the parent batches. B is a reference delivery budget, not equal realized wall-clock time or cash expenditure; independent research evaluation is charged separately.

Calibration uses 16 training batches; the registered u and c_1 are 1.25 times the measured batch and 400-episode validation durations, a margin fixed before use. Recovery is charged once even when several candidates are trained, which is more permissive to splitting than charging it per candidate. No production batch count is adjusted for observed performance, elapsed time or expenditure.

3.3 Measurements and inference

This section defines the paired delivery contrast, two candidate-set diagnostics, the game-cluster bootstrap intervals and the leave-one-block-out check that the results use.

For block b, the primary contrast is $D_b = p_J - p_L$, the delivered model's success rate minus its paired same-budget long run. We report six-block means in percentage points (pp). The candidate diagnostic compares $\max_i p_i - p_L$ and $\max_i p_i - p_J$ on the independent pool; these empirical maxima are noisy, optimistic post-hoc quantities, not deployable selection rules.

Intervals use 10,000 paired game-cluster bootstrap resamples with nearest-rank endpoints. They condition on the trained models; six-block scatter and leave-one-block-out checks describe training-run variation (Agarwal et al., 2021; Patterson et al., 2024). Because seed and implementation variance can dominate reported differences in deep RL, in supervised NLP training and in fine-tuning (Henderson et al., 2018; Engstrom et al., 2020; Reimers & Gurevych, 2018; Mosbach et al., 2021), and the number of independent trials affects the reliability of a comparison (Colas et al., 2018; Bouthillier et al., 2021), every contrast is paired within blocks and reported with its deletion sensitivity. We avoid normal-approximation intervals (Bowyer et al., 2025). All analyses are descriptive except the registered reserved-pool procedure in Section 4.1; an interval covering zero is not evidence of equivalence.

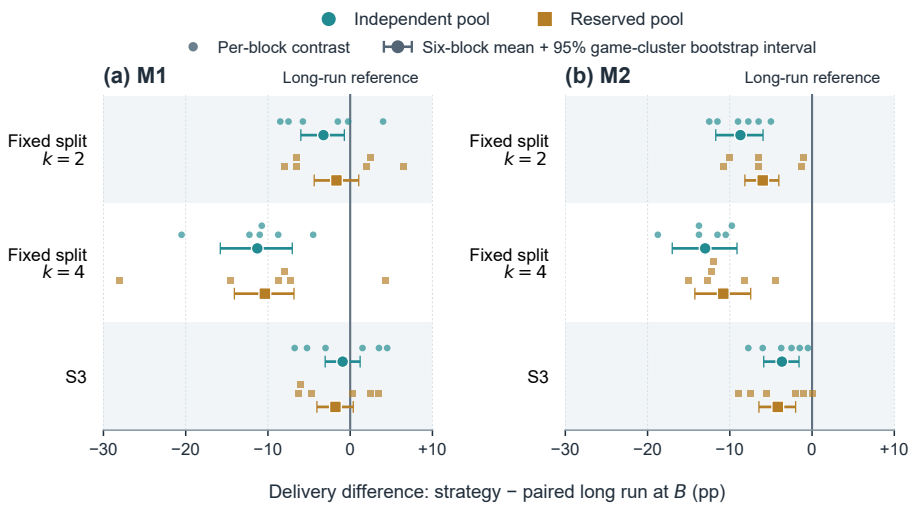

Figure 2: **Delivery differences at B.** Contrasts are delivered strategy minus paired long run, in percentage points. Small markers show six blocks; large markers and bars show their mean and 95% game-cluster bootstrap interval conditional on the trained models. Circles denote independent and squares reserved evaluation. The reserved $k = 4$ comparison is the registered target; other contrasts are descriptive. Vertical jitter only separates markers.

Every draw resamples the same games for all compared models. Direct S3–fixed and B–$2B$ comparisons are computed within blocks before averaging, and the same-budget long reference at $2B$ is the new long run at that budget, not the reused B long run.

We also record the candidate-mean gap $C_b = k^{-1}\sum_i p_i - p_L$ and the selection gain $G_b = p_J - k^{-1}\sum_i p_i$, so $D_b = C_b + G_b$, a reporting identity rather than a causal decomposition; complete candidate scores and delivered identities are in Appendix E.

Leave-one-block-out analysis repeats each headline comparison after each of six block deletions. We call a contrast stable when every deletion preserves its sign and excludes zero, and fragile when its full interval excludes zero but a deletion makes it cover zero. Contrasts whose full interval already covers zero are recorded separately; the six-block result remains the reported estimate in every case.

4 Results

We report the results in the order of the questions they answer: fixed splitting against one long run at the reference budget and its reserved-pool confirmation (Section 4.1), whether a better pick from the same candidates would change the answer (Section 4.2), whether continuing a selected candidate or a doubled budget does (Sections 4.3 and 4.4), and which findings persist after any single-block deletion (Section 4.5). Contrasts are the delivered model minus its paired long run, in percentage points, unless stated otherwise.

4.1 Delivery at the reference budget

We begin with the central comparison at the reference budget B: in each of the six blocks per instance, the model delivered by the $k = 2$ and $k = 4$ policies against its paired long run, all trained under the accounting of Section 3.2. Figure 2 shows the block-level contrasts and their means on both pools. On the independent pool, fixed splitting falls below the long run in 23 of 24 block-level comparisons. The mean differences for M1 are -3.25 pp at $k = 2$ and -11.29 pp at $k = 4$; for M2 they are -8.71 pp and -13.00 pp. At $k = 4$, both instances' intervals exclude zero on both pools, and this ordering survives every single-block deletion.

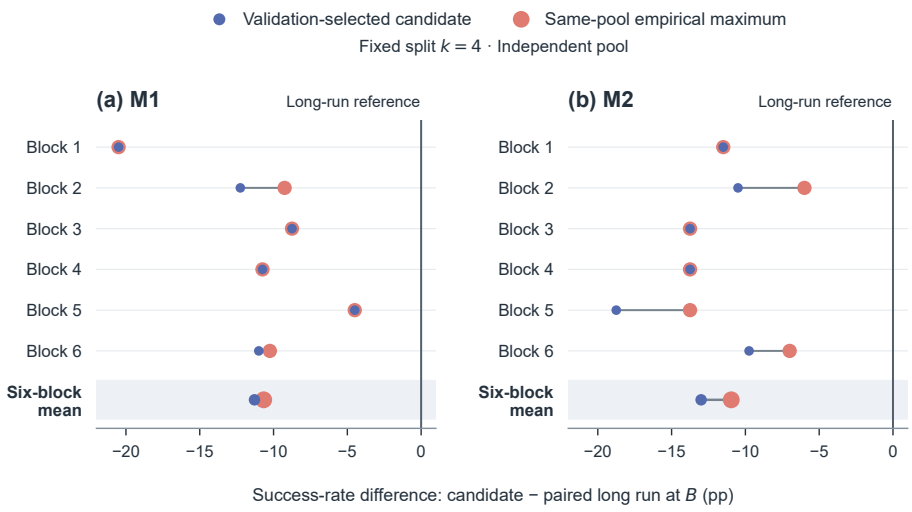

Figure 3: **Candidate-set diagnostic at $k = 4$, independent pool.** Blue markers are validation-selected candidates; coral markers are same-pool empirical maxima, each relative to the paired long run. Segments show empirical selection shortfall, and concentric markers indicate coincidence, including ties. The last row averages six blocks. Coral markers are larger for visibility, not uncertainty; the same-pool maximum is an optimistic post-hoc diagnostic.

The reserved pool confirms the four-candidate deficit. The registered target is the reserved-pool $k = 4$ delivery minus the paired B long run, separately for each instance, with a one-sided rule and Holm correction over two instances. Mean differences are -10.38 pp $[-14.08, -6.83]$ for M1 and -10.79 pp $[-14.25, -7.46]$ for M2. No game resample reaches zero in either instance, and both meet the registered rule. The registered bootstrap tail proportion is not a calibrated finite-sample p-value; the null-shifted sensitivity retains the direction. The rule predates independent evaluation and reserved-pool access; Appendix I gives the registration timeline, including one smoke-run validation count that already existed, and the statistical details.

4.2 Candidate-set diagnostic

A delivery deficit can arise because the candidates were weak or because validation picked the wrong one. Because every fixed-splitting candidate was evaluated on the independent pool, the two can be separated: we compare the best candidate of each observed four-candidate set, judged on that pool itself, with the paired long run, and the validation pick with that best candidate. Figure 3 shows both quantities per block. Even the empirical maximum falls below the long run in every block of both instances. Its mean deficit is 10.67 pp for M1 and 10.96 pp for M2; validation selection adds a further 0.63 pp and 2.04 pp of mean shortfall, respectively.

Selection does improve on the average candidate: at $k = 4$, the mean selection gain G is 9.6 pp for M1 and 1.0 pp for M2. A substantial selection gain can therefore coexist with a delivery deficit, because the candidate mean and the paired long run are different reference points. The largest observed selection shortfalls within a block are 3.0 pp and 5.0 pp, respectively; the diagnostic measures this loss separately from the gap that remains even at the empirical maximum. Training length, schedules and stochastic trajectories differ together between the arms, so the diagnostic locates the shortfall in the observed candidate sets without isolating its cause.

4.3 Adaptive continuation

Fixed splitting never continues a candidate once it is trained. The successive-halving arm S3 does: it trains two stems, promotes one on validation and continues it, so it tests whether the deficit of Section 4.1 belongs to starting several candidates as such or to fixed splitting in particular. Continuing a selected stem recovers much of the fixed-splitting deficit. S3 improves over $k = 4$ by 10.38 pp

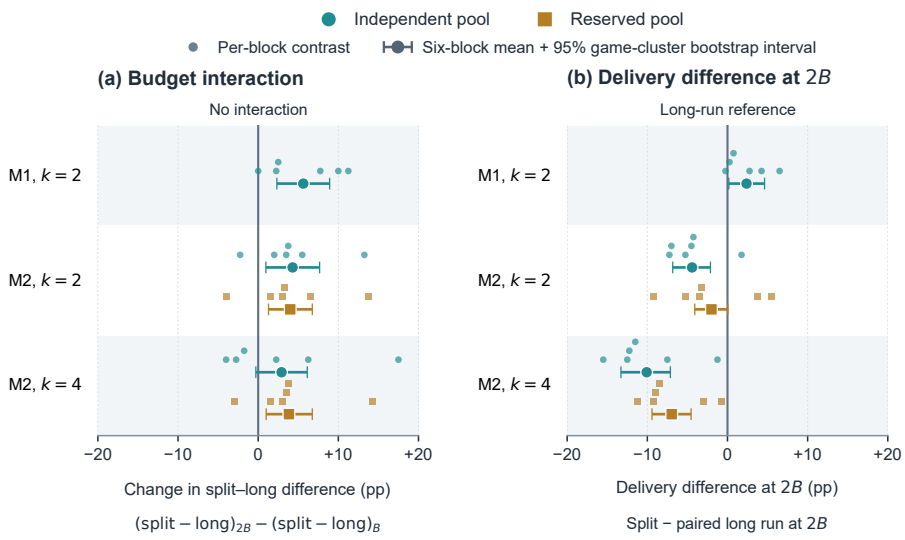

Figure 4: **Relative budget response and final delivery at $2B$.** Panel (a) is $(p_{k,2B} - p_{1,2B}) - (p_{k,B} - p_{1,B})$; panel (b) is $p_{k,2B} - p_{1,2B}$. Here $p_{k,B}$ denotes delivered success at candidate count k and budget B. Small markers are six block contrasts; large markers and bars are means and conditional 95% game-cluster intervals. Pools follow the circle/square encoding. All comparisons are descriptive. M1 covers $k \in \{1, 2\}$ independently only; M2 covers $k \in \{1, 2, 4\}$ on both pools.

$[7.08, 14.00]$ (M1) and 9.33 pp $[6.08, 12.79]$ (M2) independently, and by 8.58 pp and 6.63 pp on the reserved pool; all four direct paired intervals exclude zero under every single-block deletion. Validation promotes the second stem in three of six blocks in each instance, so the continuation is not determined by a fixed preference for the first stem.

Relative to the long run, S3 is -0.92 pp for M1, with an interval covering zero, and -3.67 pp for M2 on the independent pool. Against $k = 2$, S3 is ahead in M2 on both pools and close to it in M1, at $+2.33$ pp independently and -0.13 pp on the reserved pool. Full paired estimates and intervals are in Appendix D.

4.4 Doubling the budget

The comparisons so far hold the budget at B. To see whether the deficit is tied to that budget, the fixed-splitting comparison was repeated at $2B$ against new long runs trained at that budget, for $k = 2$ in both instances and $k = 4$ in M2 (Table 1). Figure 4 keeps two quantities apart: the budget interaction, which measures how much the split–long gap changes when the budget doubles, and the delivery difference at $2B$ itself. At $k = 2$, the independent-pool budget interaction is $+5.63$ pp $[2.33, 8.92]$ for M1 and $+4.29$ pp $[0.96, 7.67]$ for M2. More budget therefore narrows the split–long gap in both six-block estimates. At $2B$, however, M1's delivery difference is $+2.38$ pp $[0.17, 4.63]$, while M2's remains -4.42 pp $[-6.83, -2.13]$. M2's four-candidate delivery is also 3.33–4.54 pp below the cheaper long run at B across the two pools. S3 at $2B$ and M1's $k = 4$ at $2B$ were not run, so these findings concern the tested fixed-splitting policies.

4.5 Robustness and protocol coverage

We retain the full six-block estimates and use leave-one-block-out analysis to assess dependence on individual blocks. The $k = 4$ deficit at B and S3's advantage over $k = 4$ retain their direction and exclude zero after every deletion in both instances and pools, and M1's $k = 2$ budget interaction does so as well. Three results are more sensitive: the M1 $k = 2$ estimate at B has an interval that covers zero under some deletions, M1's positive $2B$ ordering is sensitive to deleting a block, and M2's $k = 2$ interaction intervals cover zero without block 5. Complete checks are in Appendix G.

The intervals and the block scatter answer different questions: the game-cluster intervals describe evaluation-pool sampling conditional on the six trained models of each arm, whereas the spread of a contrast across blocks describes training-block variation and, with six blocks, is descriptive only. Table 16 in Appendix G lists both for every headline contrast. On the independent pool, the $k = 4$ deficit and S3's gain over $k = 4$ have all six blocks on the side of their mean, with block standard deviations between 1.1 and 5.3 pp; the two-candidate $2B$ delivery contrasts and the $k = 2$ budget interaction have block standard deviations of the same order as their means and, except for M1's interaction, at least one block on the other side of zero.

Three protocol failures were retried after partial validation, so the results condition on protocol completion and the effect of that selection filter is not bounded. Incident-block exclusions and the retained retry records are reported in Appendix I.

5 Discussion and limitations

For the language-model agents studied here, selecting among four independent PPO candidates improved on the average candidate, but did not outperform one longer run at the reference budget B. Evaluating the discarded candidates is what makes that observation informative: on the independent pool, even the best member of each observed set remained below its paired long run, so no alternative pick from those candidates would have closed the observed independent-pool gap to the long run. A multiple-run policy should therefore be assessed on its candidate set as well as on its selection rule.

The allocation policy, and not the number of candidates alone, determines how a budget is used: continuing a selected stem improves on four-way fixed splitting, and more budget improves two-way splitting's relative gap. For practitioners in a comparable setting, a paired long run is therefore the useful reference for a proposed multi-candidate allocation; an improvement over another split policy does not by itself show a better delivered model.

Limitations. The evidence covers one text-game family, one PPO recipe, two model instances and six blocks per instance. Training duration, schedules and trajectories vary together, so the comparisons do not identify a causal effect of duration or architecture. Our intervals concern game sampling conditional on the trained models, and the same-pool empirical maximum is not a deployable rule. Different operational costs would yield different batch allocations; actual campaign cost is separate from the reference budget. The protocol-completion filter and one-time use of the reserved pool further limit inference beyond the observed runs, and the study does not identify a universally optimal number of runs. The fixed-splitting policies select only final checkpoints; selection among intermediate checkpoints, with its validation charged, is untested.

Conclusion. Training and selecting language-model agents should be assessed together under a delivery budget. In both studied PPO text-game instances, one long run outperformed validation selection among four shorter runs by more than ten percentage points. On the independent pool, even each set's empirical best stayed below the paired long run, so changing the selection alone could not close that observed gap. Continuing a selected stem and adding budget changed the comparison. These results show why the value of additional training candidates should be measured against extending one run, with candidate quality and selection shortfall evaluated separately.

Reproducibility statement

The main statistical results and completed-work costs are recomputed by six scripts from saved evaluation and run records. The commands, inputs and cost-table cutoff are given in Appendix J, and the derived JSON snapshots are content-hashed so that a published table cannot change when later units are added. The supplementary archive contains the scripts, the saved evaluation records and the snapshots needed to recompute the paper's tables without a GPU; training and evaluation themselves are not re-run. Two test modules in the archive check the tabulated cells against the snapshots. The cost model of Section 3.2 is arithmetic on the measured components reported in Appendix A. Appendix C states exactly which verification each run carries and which stronger check it does not.

ETHICS STATEMENT

This work trains small language-model agents on synthetic text games; it involves no human subjects, no personal data and no release of models trained on sensitive material. The main externality is compute: Appendix B reports the measured Pod-hours and cost of this sprint, including the fraction spent on idle time, waiting and failed attempts, so that the cost of this kind of allocation study is visible rather than hidden. That figure is the incremental cost of this study and excludes the parent batches it builds on, as stated there.

AI USE STATEMENT

We used generative AI tools to assist with research design, implementation, analysis, and manuscript preparation. Research assistance included refining the experimental comparisons, developing the budget-accounting and candidate-set diagnostic framework, drafting experimental protocols, and interpreting results. AI coding assistants helped implement and test the data-preparation, training, evaluation, and analysis pipelines, operate experiments, and assemble the supplementary materials.

We also used AI tools to assist with literature searches and citation checking, draft and revise the manuscript, and develop figure-generation scripts. A second AI system assisted with reviewing the experimental plans, recomputing analyses, and checking claims against the saved records. Verification included regression tests, numerical consistency checks, and comparisons with the underlying experimental outputs and cited sources. We reviewed the AI-assisted materials incorporated into this work and take responsibility for the final text, methods, results, and artifacts.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021. arXiv:2108.13264.

Loubna Ben Allal et al. SmolLM2: When smol goes big – data-centric training of a small language model, 2025. arXiv:2502.02737.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Naz Sepah, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Dmitriy Serdyuk, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 3, 2021. arXiv:2103.03098.

Sam Bowyer, Laurence Aitchison, and Desi R. Ivanova. Position: Don't use the CLT in LLM evals with fewer than a few hundred datapoints. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. arXiv:2503.01747.

Alper Kamil Bozkurt, Xiaoan Xu, Shangtong Zhang, Miroslav Pajic, and Yuichi Motai. Adaptive policy selection and fine-tuning under interaction budgets for offline-to-online reinforcement learning. In Gaurav Sukhatme, Lars Lindemann, Stephen Tu, Adam Wierman, and Nikolay Atanasov (eds.), *Proceedings of The 8th Annual Learning for Dynamics and Control Conference*, volume 331 of *Proceedings of Machine Learning Research*, pp. 1804–1817, 2026. arXiv:2605.05123.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 3676–3713, 2023. arXiv:2302.02662.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107, 2010.

Kevin Chen, Marco Cusumano-Towner, Brody Huval, Aleksei Petrenko, Jackson Hamburger, Vladlen Koltun, and Philipp Krähenbühl. Reinforcement learning for long-horizon interactive LLM agents, 2025. arXiv:2502.01600.

Minping Chen, Bowen Xiao, Du Liang, Chuxuan Zeng, and Zeyi Wen. Efficient hyperparameter optimization for LLM reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27540–27552, 2026. doi: 10.18653/v1/2026.acl-long.1271.

Zhoujun Cheng, Yutao Xie, Yuxiao Qu, Amrith Setlur, Shibo Hao, Varad Pimpalkhute, Tongtong Liang, Feng Yao, Zhengzhong Liu, Eric Xing, Virginia Smith, Ruslan Salakhutdinov, Zhiting Hu, Taylor Killian, and Aviral Kumar. IsoCompute playbook: Optimally scaling sampling compute for LLM RL, 2026. arXiv:2603.12151.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. How many random seeds? Statistical power analysis in deep reinforcement learning experiments, 2018. arXiv:1806.08295.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. TextWorld: A learning environment for text-based games, 2018. arXiv:1806.11532; also in Computer Games (CGW 2018), Springer CCIS 1017, 2019.

George E. Dahl, Frank Schneider, Zachary Nado, Naman Agarwal, et al. Benchmarking neural network training algorithms, 2023. arXiv:2306.07179.

Victor H. de la Peña, Fangyuan Lin, and Victor K. de la Peña. A selection premium decomposition for the expected maximum of random walks, 2026. arXiv:2602.19481.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2185–2194, 2019. arXiv:1909.03004.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping, 2020. arXiv:2002.06305.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on PPO and TRPO. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:2005.12729.

Haoyang Fang, Wei Zhu, Boran Han, Alex Zhang, Zhenyu Pan, Shuo Yang, Shuai Zhang, Jiading Gai, Peng Tang, Cuixiong Hu, Xuan Zhu, Huzefa Rangwala, George Karypis, and Bernie Wang. LLMZero: Discovering adaptive training strategies for RL post-training via LLM agents, 2026. arXiv:2606.18388.

András György and Levente Kocsis. Efficient multi-start strategies for local search algorithms. *Journal of Artificial Intelligence Research*, 41:407–444, 2011.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. arXiv:1709.06560.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. In *Conference on Language Modeling (COLM)*, 2025. arXiv:2504.07086.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 51 of *Proceedings of Machine Learning Research*, pp. 240–248, 2016. arXiv:1502.07943.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S. Dhillon, David Brandfonbrener, and Rishabh Agarwal. The art of scaling reinforcement learning compute for LLMs. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.13786.

Sunghwan Kim, Junhee Cho, Beong-woo Kwak, Taeyoon Kwon, Liang Wang, Nan Yang, Xingxing Zhang, Furu Wei, and Jinyoung Yeo. On training large language models for long-horizon tasks: An empirical study of horizon length. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. arXiv:2605.02572.

Ksenia Konyushkova, Yutian Chen, Tom Le Paine, Caglar Gulcehre, Cosmin Paduraru, Daniel J. Mankowitz, Misha Denil, and Nando de Freitas. Active offline policy selection. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24631–24644, 2021. arXiv:2106.10251.

Vladislav Kurenkov and Sergey Kolesnikov. Showing your offline reinforcement learning work: Online evaluation budget matters. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11729–11752, 2022. URL `https://proceedings.mlr.press/v162/kurenkov22a.html`.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018. arXiv:1603.06560.

Andrei Lissovoi, Dirk Sudholt, Markus Wagner, and Christine Zarges. Theoretical results on bet-and-run as an initialisation strategy. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 857–864, 2017. doi: 10.1145/3071178.3071329.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2006.04884.

Andrew Patterson, Samuel Neumann, Martha White, and Adam White. Empirical design in reinforcement learning. *Journal of Machine Learning Research*, 25(318):1–63, 2024. arXiv:2304.01315.

Aleksei Petrenko, Ben Lipkin, Kevin Chen, Erik Wijmans, Marco Cusumano-Towner, Raja Giryes, and Philipp Krähenbühl. Entropy-preserving reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2603.11682; OpenReview record.

Riccardo Poli and Ahmet Yilmaz. Gradient-descent steps to success over mean accuracy: A paradigm shift for ML, 2026. arXiv:2606.22053.

Qwen Team. Qwen2.5 technical report, 2024. arXiv:2412.15115.

Nils Reimers and Iryna Gurevych. Why comparing single performance scores does not allow to draw conclusions about machine learning approaches, 2018. arXiv:1803.09578.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016. arXiv:1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. arXiv:1707.06347.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *European Conference on Computer Systems (EuroSys)*, 2025. arXiv:2409.19256.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.03768.

James E. Smith and Robert L. Winkler. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006. doi: 10.1287/mnsc.1050.0451.

Raphael Tang, Jaejun Lee, Ji Xin, Xinyu Liu, Yaoliang Yu, and Jimmy Lin. Showing your work doesn't always work. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 2766–2772, 2020. arXiv:2004.13705.

Ruiyi Wang and Prithviraj Ammanabrolu. A practitioner's guide to multi-turn agentic reinforcement learning, 2025. arXiv:2510.01132.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Fei-Fei Li, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning, 2025. arXiv:2504.20073.

Zihan Wang, Chi Gui, Xing Jin, Qineng Wang, Licheng Liu, Kangrui Wang, Shiqi Chen, Linjie Li, Zhengyuan Yang, Pingyue Zhang, Yiping Lu, Jiajun Wu, Fei-Fei Li, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN-2: Reasoning collapse in agentic RL, 2026. arXiv:2604.06268.

Patrick Wilhelm and Odej Kao. Where should RL post-training compute go? Model size, search, learning, and feedback, 2026. arXiv:2607.13389.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, 2022. arXiv:2203.05482.

A REFERENCE-BUDGET ALLOCATION

Table 2 lists the calibrated cost components and Table 3 the batch counts they imply under both accountings. Accounting A charges the delivery recovery F for every candidate; accounting B, the one executed, charges it only for the delivered model (Appendix H.3).

Table 2: Calibrated cost components, from the frozen configurations. All values in seconds.

instance	u	c_1	I	S	R	F	d
M1 (Qwen2.5-1.5B)	9.865	157.87	78.46	31.80	29.32	566.79	64.64
M2 (SmolLM2-1.7B)	11.183	173.42	54.13	38.19	31.31	487.09	103.53

Table 3: Batches per candidate. **Bold**: arms executed and reported here; the M1 $2B$ unit covers $k = 1$ and $k = 2$ only. Every other entry is the count the accounting leaves, which is not the same as feasible: entries below the registered floor of 16 batches, and the negative ones under accounting A, are infeasible by construction. M1 at $2B$, $k = 4$ is feasible at 131 batches but was not executed. At B, accounting A cannot afford $k = 4$ at all.

	M1 at B		M2 at B		M1 at $2B$	M2 at $2B$
k	A	B	A	B	B	B
1	270	**270**	242	**242**	**635**	**564**
2	88	**117**	81	**103**	**299**	**264**
3	27	65	27	56	187	164
4	−3	**40**	1	**33**	131	**114**
8	−49	1	−40	−2	47	39

Figure 5 visualizes the batch counts derived in Section 3.2; it reports accounting quantities, not learning outcomes.

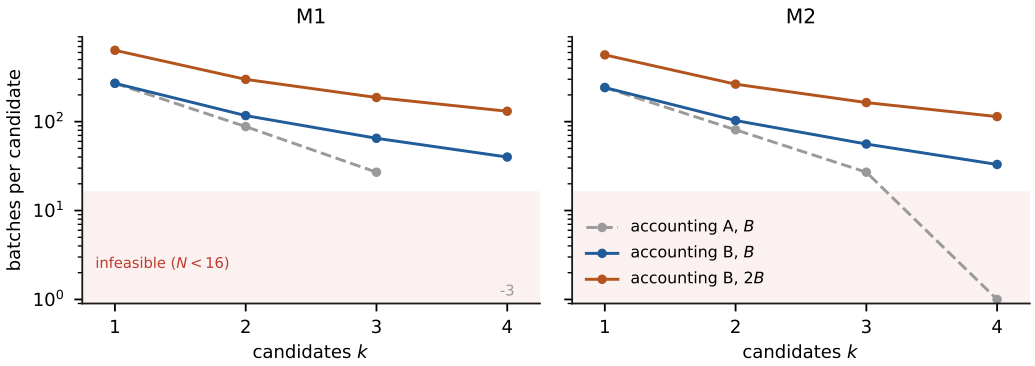

Figure 5: The accounting, not the learning, decides how many candidates are affordable. Batches per candidate against k under the two accountings; the shaded band is the registered feasibility floor of 16 batches. Under accounting A at budget B (grey) M1 has no budget left to train with at $k = 4$ (the value is -3, printed rather than drawn) and M2 has one batch. Under accounting B (blue), which is the one we run, $k = 4$ trains 40 and 33 batches. $k = 8$ is omitted here and given in Table 3.

B PHYSICAL COST

This section reports the **incremental** cost of the sprint described in this paper, up to a fixed cutoff. It does *not* include the cost of producing the parent batches that supply every block's long run and short reference run, nor the supervised starts or the earlier calibrations, all of which this work consumes

as inputs and whose cost is recorded in those batches' own reports. A total for the whole research programme would have to add those ledgers; we do not treat them as zero.

Measured Pod wall-clock of completed work, priced at each Pod's rate (3.25 USD/h for an H100 NVL including disk), with a hard cutoff so that the table is reproducible. The cutoff predates the $2B$ unit of the first instance, whose 18 runs and 18 evaluations are therefore *not* in the figures below; they are reported as a separate increment at the end of this section. Within the cutoff: 150 training-run timings, 75.80 h, 246.34 USD; 126 independent evaluations (group-shared), 17.93 h, 58.28 USD; 80 reserved evaluations, 8.42 h, 27.36 USD; 36 successive-halving preflights, 0.91 h, 2.96 USD. Subtotal **103.06 h, 334.94 USD**.

The rows are not homogeneous: an ordinary run's marker covers its preflight, training, verification and model retention, whereas a successive-halving row takes the training command from the stage timings and charges its preflight separately. Rental-window cost to the same cutoff is **404.80 USD** across three Pods from their rental starts (124.55 Pod-hours); the **69.86 USD** difference is deployment, idle and waiting time, the three failed attempts and technical test runs. We have not itemized it; it is an arithmetic reconciliation against rental start times rather than a vendor invoice. The reference delivery budget B of Section 3.2 is a rule for fixing batch counts, not this number.

The unit after the cutoff. Running the same script with no cutoff adds exactly the 36 rows of that unit: its 18 training runs are 24.75 h and 80.42 USD, its 18 independent evaluations 2.77 h and 9.00 USD, an increment of **27.52 h, 89.42 USD**. Completed-phase work through that unit is then 130.57 h and **424.36 USD** at the same rates, which discharges the registration's undertaking to report that unit's measured per-run cost. Both figures are accounted work, not a final rental total: the 404.80 USD above is a rental-window figure to the earlier cutoff only, so the two cannot be differenced, and backup, waiting, the aborted start and the idle time before shutdown are in neither. We keep the frozen cutoff table as the published one and report this increment beside it.

C Operational verification and resume details

What is verified. Each run's inputs hash to the frozen plan's binding; the training record must show completion with the registered per-game coverage and exactly $N - 2$ actor and N critic updates, where N counts the batches *that run* executes—for a successive-halving stem that is n_0. A continuation inherits its stem's counters and is accepted against the *cumulative* totals at T, not against the $T - n_0$ updates it adds: M1's continuations add 97 and are accepted at 153 actor and 155 critic updates, M2's add 89 and are accepted at 138 and 140.

For ordinary runs and stems the final validation summary must be a complete 400-episode record for that run id; the 12 continuations are not validated again (Section 3.1) and are checked instead through their resume binding and their final independent evaluation. The delivered model directory is hash-inventoried file-by-file before and after it is moved into place. Independent and reserved evaluation then loads the model from files whose fingerprints match the sealed inventory and audits every request against the request plan. The stronger per-tensor check—that every exported tensor equals the tensor of the reloaded model—was run in the *parent* batches; the runs reported here record `model_reload_verified: false`, so that check is not part of their evidence. Section 4.5 reports the three runs that were stopped and re-run.

The resume path was accepted before production, from two separate kinds of evidence that we keep apart. A three-leg GPU test compared a continuous run against a save-exit-resume run on a small horizon; its acceptance criteria were the step sequence, the learning-rate trajectory (max $|\Delta \mathrm{LR}| = 0.0$) and the training coverage order. Parameter and optimizer loading and RNG restoration are supported instead by the production checkpoint bindings and run logs. Neither is a claim of bitwise-identical trajectories, and we do not make one.

D DETAILED RESULT TABLES

The tables below retain the full numerical results underlying the main-text comparisons. Table 4 reports the same-pool candidate diagnostic at each arm's own budget; Table 5 also includes S3 versus the paired long run; Table 6 includes both same-budget and explicitly labeled cross-budget comparisons; Table 7 gives the descriptive reserved-pool levels. All values are copied from the analysis outputs listed in Appendix J. Here x1, x2 and x4 denote the long run and the two- and four-candidate policies at $2B$; long, k=2, k=4 and S3 are the policies at B.

Table 4: Empirical best candidate and empirical selection shortfall, independent pool, six blocks. Each arm is compared with the $k = 1$ delivery at its own budget. Differences are in percentage points.

instance / arm	reference	best − ref.	shortfall (mean)	shortfall (worst)	blocks best < ref.
M1 $k = 2$	long at B	−3.25	0.00	0.00	5/6
M1 $k = 4$	long at B	−10.67	+0.63	+3.00	**6/6**
M2 $k = 2$	long at B	−8.50	+0.21	+1.25	6/6
M2 $k = 4$	long at B	−10.96	+2.04	+5.00	**6/6**
M2 x2 ($2B$)	x1 at $2B$	−3.29	+1.13	+2.75	5/6
M2 x4 ($2B$)	x1 at $2B$	−9.17	+0.92	+2.75	6/6
M1 x2 ($2B$)	x1 at $2B$	+2.58	+0.21	+1.25	**1/6**

Table 5: Successive halving (S3) against the other arms, paired per block, both pools (pp, 95 % game-cluster intervals).

contrast	M1 indep.	M1 res.	M2 indep.	M2 res.
S3 − k=4	+10.38 [7.08, 14.00]	+8.58 [5.75, 11.63]	+9.33 [6.08, 12.79]	+6.63 [4.17, 9.21]
S3 − k=2	+2.33 [0.04, 4.63]	−0.13 [−2.46, 2.25]	+5.04 [3.13, 7.21]	+1.83 [0.25, 3.46]
S3 − long	−0.92 [−3.04, 1.21]	−1.79 [−4.04, 0.38]	−3.67 [−5.88, −1.58]	−4.17 [−6.46, −2.00]

Table 6: Levels and interactions at $2B$ (pp, 95 % game-cluster intervals). The M1 unit covers $k \in \{1, 2\}$ on the independent pool only; "—" marks a cell that was never run, not one awaiting results.

contrast	M2 independent	M2 reserved	M1 independent
x2 − x1	−4.42 [−6.83, −2.13]	−2.00 [−4.08, +0.04]	+2.38 [+0.17, +4.63]
x4 − x1	−10.08 [−13.29, −7.13]	−6.96 [−9.42, −4.54]	—
x4 − long at B	**−4.54** [−6.71, −2.46]	**−3.33** [−5.17, −1.54]	—
interaction at $k = 2$	**+4.29** [+0.96, +7.67]	**+4.00** [+1.29, +6.75]	**+5.63** [+2.33, +8.92]
interaction at $k = 4$	+2.92 [−0.29, +6.13]	+3.83 [+1.00, +6.75]	—

Table 7: Reserved-pool mean success rates per arm (descriptive; comparisons stay inside this pool, whose absolute rates run higher than the independent pool's). Success rates are proportions. The M1 $2B$ cells are empty by registration, not pending: that unit evaluates on the independent pool only.

instance	long	S3	k=2	k=4	short	x1 / x2 / x4	start
M1	.8646	.8467	.8479	.7608	.7975	—	.4225
M2	.6471	.6054	.5871	.5392	.5650	.6833 / .6633 / .6138	.5450

E COMPLETE MODEL AND DELIVERY LEDGERS

This section gives complete model-level and paired delivery results for the executed allocation policies, without adding training runs or evaluation requests. Tables 8 and 9 give the M1 and M2

model ledgers. The model ledger contains every fixed-splitting candidate (selected or discarded), the reused long runs, both S3 stems and the delivered continuations. Entries are observed successes out of 400 episodes in each pool. Validation scores are the sealed selection counts; the independent and reserved scores are read from their completed evaluation records. A continuation has no new validation pass: NA marks that case, rather than copying its stem's score. NE marks protocol-defined non-evaluation, including discarded candidates on the reserved pool, both S3 stems on the independent and reserved pools, and every M1 $2B$ model on the reserved pool. Delivered means selected for delivery (or the sole long run); Promoted identifies the stem resumed by S3. Candidate numbers are local to the instance, block and arm; full model identifiers and source hashes are retained in the machine-readable companion ledgers. These model rows are not independent training blocks: inference and six-block means retain the original block pairing.

Table 8: M1: complete allocation-policy model ledger. V, I and R are validation, independent and reserved successes out of 400. NA denotes no new validation; NE denotes no evaluation.

Budget	Arm	Block	Cand.	V	I	R	Role
B	long	1	1	375	359	383	Delivered
B	k2	1	1	347	336	351	Delivered
B	k2	1	2	308	313	NE	Not selected
B	k4	1	1	206	210	NE	Not selected
B	k4	1	2	258	268	NE	Not selected
B	k4	1	3	175	187	NE	Not selected
B	k4	1	4	270	277	271	Delivered
B	S3 stem	1	1	333	NE	NE	Promoted
B	S3 stem	1	2	326	NE	NE	Not promoted
B	S3 cont.	1	–	NA	338	358	Continuation
$2B$	x1	1	1	379	341	NE	Delivered
$2B$	x2	1	1	375	358	NE	Delivered
$2B$	x2	1	2	358	351	NE	Not selected
B	long	2	1	370	365	361	Delivered
B	k2	2	1	343	331	335	Delivered
B	k2	2	2	300	273	NE	Not selected
B	k4	2	1	287	257	NE	Not selected
B	k4	2	2	300	294	NE	Not selected
B	k4	2	3	328	328	NE	Not selected
B	k4	2	4	337	316	326	Delivered
B	S3 stem	2	1	196	NE	NE	Not promoted
B	S3 stem	2	2	344	NE	NE	Promoted
B	S3 cont.	2	–	NA	353	342	Continuation
$2B$	x1	2	1	352	324	NE	Delivered
$2B$	x2	2	1	328	340	NE	Not selected
$2B$	x2	2	2	372	335	NE	Delivered
B	long	3	1	335	330	334	Delivered
B	k2	3	1	317	319	NE	Not selected
B	k2	3	2	353	329	344	Delivered
B	k4	3	1	247	209	NE	Not selected
B	k4	3	2	284	285	NE	Not selected
B	k4	3	3	144	169	NE	Not selected
B	k4	3	4	291	295	302	Delivered
B	S3 stem	3	1	307	NE	NE	Not promoted
B	S3 stem	3	2	332	NE	NE	Promoted
B	S3 cont.	3	–	NA	336	335	Continuation
$2B$	x1	3	1	377	349	NE	Delivered
$2B$	x2	3	1	360	348	NE	Delivered
$2B$	x2	3	2	337	335	NE	Not selected
B	long	4	1	282	328	330	Delivered
B	k2	4	1	350	344	338	Delivered

Continued on next page

Table 8 (continued)

Budget	Arm	Block	Cand.	V	I	R	Role
B	k2	4	2	328	314	NE	Not selected
B	k4	4	1	173	189	NE	Not selected
B	k4	4	2	307	285	301	Delivered
B	k4	4	3	250	255	NE	Not selected
B	k4	4	4	257	264	NE	Not selected
B	S3 stem	4	1	365	NE	NE	Promoted
B	S3 stem	4	2	353	NE	NE	Not promoted
B	S3 cont.	4	–	NA	342	340	Continuation
$2B$	x1	4	1	364	338	NE	Delivered
$2B$	x2	4	1	360	359	NE	Not selected
$2B$	x2	4	2	375	364	NE	Delivered
B	long	5	1	325	329	307	Delivered
B	k2	5	1	344	323	333	Delivered
B	k2	5	2	339	290	NE	Not selected
B	k4	5	1	187	186	NE	Not selected
B	k4	5	2	316	284	NE	Not selected
B	k4	5	3	213	229	NE	Not selected
B	k4	5	4	333	311	324	Delivered
B	S3 stem	5	1	337	NE	NE	Not promoted
B	S3 stem	5	2	343	NE	NE	Promoted
B	S3 cont.	5	–	NA	347	321	Continuation
$2B$	x1	5	1	378	343	NE	Delivered
$2B$	x2	5	1	372	346	NE	Delivered
$2B$	x2	5	2	351	324	NE	Not selected
B	long	6	1	355	360	360	Delivered
B	k2	6	1	356	330	334	Delivered
B	k2	6	2	329	311	NE	Not selected
B	k4	6	1	314	319	NE	Not selected
B	k4	6	2	296	270	NE	Not selected
B	k4	6	3	323	316	302	Delivered
B	k4	6	4	277	272	NE	Not selected
B	S3 stem	6	1	285	NE	NE	Not promoted
B	S3 stem	6	2	312	NE	NE	Promoted
B	S3 cont.	6	–	NA	333	336	Continuation
$2B$	x1	6	1	320	351	NE	Delivered
$2B$	x2	6	1	365	352	NE	Delivered
$2B$	x2	6	2	350	330	NE	Not selected

Table 9: M2: complete allocation-policy model ledger. V, I and R are validation, independent and reserved successes out of 400. NA denotes no new validation; NE denotes no evaluation.

Budget	Arm	Block	Cand.	V	I	R	Role
B	long	1	1	235	247	264	Delivered
B	k2	1	1	214	197	238	Delivered
B	k2	1	2	168	180	NE	Not selected
B	k4	1	1	188	177	NE	Not selected
B	k4	1	2	216	201	216	Delivered
B	k4	1	3	184	172	NE	Not selected
B	k4	1	4	203	196	NE	Not selected
B	S3 stem	1	1	236	NE	NE	Promoted
B	S3 stem	1	2	216	NE	NE	Not promoted
B	S3 cont.	1	–	NA	237	256	Continuation
$2B$	x1	1	1	270	287	284	Delivered

Continued on next page

Table 9 (continued)

Budget	Arm	Block	Cand.	V	I	R	Role
$2B$	x2	1	1	277	259	271	Delivered
$2B$	x2	1	2	257	258	NE	Not selected
$2B$	x4	1	1	233	225	250	Delivered
$2B$	x4	1	2	216	207	NE	Not selected
$2B$	x4	1	3	231	221	NE	Not selected
$2B$	x4	1	4	225	216	NE	Not selected
B	long	2	1	244	235	239	Delivered
B	k2	2	1	218	204	235	Delivered
B	k2	2	2	203	209	NE	Not selected
B	k4	2	1	210	193	221	Delivered
B	k4	2	2	196	211	NE	Not selected
B	k4	2	3	202	193	NE	Not selected
B	k4	2	4	193	174	NE	Not selected
B	S3 stem	2	1	219	NE	NE	Promoted
B	S3 stem	2	2	216	NE	NE	Not promoted
B	S3 cont.	2	–	NA	229	239	Continuation
$2B$	x1	2	1	248	265	256	Delivered
$2B$	x2	2	1	261	248	278	Delivered
$2B$	x2	2	2	252	259	NE	Not selected
$2B$	x4	2	1	225	219	NE	Not selected
$2B$	x4	2	2	211	197	NE	Not selected
$2B$	x4	2	3	227	227	NE	Not selected
$2B$	x4	2	4	228	216	244	Delivered
B	long	3	1	246	250	269	Delivered
B	k2	3	1	198	188	NE	Not selected
B	k2	3	2	242	214	243	Delivered
B	k4	3	1	198	191	NE	Not selected
B	k4	3	2	196	194	NE	Not selected
B	k4	3	3	211	195	220	Delivered
B	k4	3	4	196	195	NE	Not selected
B	S3 stem	3	1	186	NE	NE	Not promoted
B	S3 stem	3	2	190	NE	NE	Promoted
B	S3 cont.	3	–	NA	226	247	Continuation
$2B$	x1	3	1	250	263	275	Delivered
$2B$	x2	3	1	251	242	261	Delivered
$2B$	x2	3	2	243	248	NE	Not selected
$2B$	x4	3	1	200	188	NE	Not selected
$2B$	x4	3	2	228	223	NE	Not selected
$2B$	x4	3	3	226	228	NE	Not selected
$2B$	x4	3	4	235	217	238	Delivered
B	long	4	1	248	244	264	Delivered
B	k2	4	1	207	206	NE	Not selected
B	k2	4	2	240	218	221	Delivered
B	k4	4	1	181	171	NE	Not selected
B	k4	4	2	207	189	213	Delivered
B	k4	4	3	183	181	NE	Not selected
B	k4	4	4	202	183	NE	Not selected
B	S3 stem	4	1	225	NE	NE	Promoted
B	S3 stem	4	2	185	NE	NE	Not promoted
B	S3 cont.	4	–	NA	229	234	Continuation
$2B$	x1	4	1	258	263	285	Delivered
$2B$	x2	4	1	246	254	NE	Not selected
$2B$	x2	4	2	252	245	248	Delivered
$2B$	x4	4	1	230	233	249	Delivered
$2B$	x4	4	2	216	209	NE	Not selected

Continued on next page

Table 9 (continued)

Budget	Arm	Block	Cand.	V	I	R	Role
$2B$	x4	4	3	223	216	NE	Not selected
$2B$	x4	4	4	225	221	NE	Not selected
B	long	5	1	253	259	272	Delivered
B	k2	5	1	224	213	232	Delivered
B	k2	5	2	211	198	NE	Not selected
B	k4	5	1	189	179	NE	Not selected
B	k4	5	2	203	204	NE	Not selected
B	k4	5	3	214	184	212	Delivered
B	k4	5	4	196	187	NE	Not selected
B	S3 stem	5	1	210	NE	NE	Not promoted
B	S3 stem	5	2	229	NE	NE	Promoted
B	S3 cont.	5	–	NA	228	236	Continuation
$2B$	x1	5	1	219	238	248	Delivered
$2B$	x2	5	1	230	234	NE	Not selected
$2B$	x2	5	2	251	245	263	Delivered
$2B$	x4	5	1	205	203	NE	Not selected
$2B$	x4	5	2	221	225	NE	Not selected
$2B$	x4	5	3	220	219	NE	Not selected
$2B$	x4	5	4	236	233	245	Delivered
B	long	6	1	236	232	245	Delivered
B	k2	6	1	194	199	NE	Not selected
B	k2	6	2	220	212	240	Delivered
B	k4	6	1	205	193	212	Delivered
B	k4	6	2	205	204	NE	Not selected
B	k4	6	3	198	180	NE	Not selected
B	k4	6	4	179	180	NE	Not selected
B	S3 stem	6	1	200	NE	NE	Not promoted
B	S3 stem	6	2	220	NE	NE	Promoted
B	S3 cont.	6	–	NA	230	241	Continuation
$2B$	x1	6	1	275	284	292	Delivered
$2B$	x2	6	1	244	256	NE	Not selected
$2B$	x2	6	2	256	255	271	Delivered
$2B$	x4	6	1	239	234	247	Delivered
$2B$	x4	6	2	210	209	NE	Not selected
$2B$	x4	6	3	214	209	NE	Not selected
$2B$	x4	6	4	230	223	NE	Not selected

E.1 Paired per-block delivery levels and contrasts

Tables 10 and 11 give percentages for absolute rates and percentage points for differences. L is the paired long run at the same budget, J is the delivered model, and $\Delta = J - L$. Max is the largest observed independent-pool rate among the fixed-splitting candidates; it equals the sole delivered model for long runs and S3 continuations. For S3 it does not imply that stems were independently evaluated. Each Six-block mean row averages six block-level quantities; Max is therefore the mean of the per-block maxima, not a maximum across blocks. The same-pool maximum is a noisy, optimistic post-hoc diagnostic, not a deployable selection rule. The preceding ledger identifies the delivered candidate.

Table 10: M1: full paired delivery ledger. Absolute rates are percentages; differences are percentage points. NE means the pool was not evaluated.

Budget	Policy	Block	L_I	J_I	Δ_I	Max_I	L_R	J_R	Δ_R
B	Long	1	89.75	89.75	0.00	89.75	95.75	95.75	0.00
B	Long	2	91.25	91.25	0.00	91.25	90.25	90.25	0.00

Continued on next page

Table 10 (continued)

Budget	Policy	Block	L_I	J_I	Δ_I	Max_I	L_R	J_R	Δ_R
B	Long	3	82.50	82.50	0.00	82.50	83.50	83.50	0.00
B	Long	4	82.00	82.00	0.00	82.00	82.50	82.50	0.00
B	Long	5	82.25	82.25	0.00	82.25	76.75	76.75	0.00
B	Long	6	90.00	90.00	0.00	90.00	90.00	90.00	0.00
B	Long	Six-block mean	86.29	86.29	0.00	86.29	86.46	86.46	0.00
B	Fixed $k = 2$	1	89.75	84.00	-5.75	84.00	95.75	87.75	-8.00
B	Fixed $k = 2$	2	91.25	82.75	-8.50	82.75	90.25	83.75	-6.50
B	Fixed $k = 2$	3	82.50	82.25	-0.25	82.25	83.50	86.00	+2.50
B	Fixed $k = 2$	4	82.00	86.00	+4.00	86.00	82.50	84.50	+2.00
B	Fixed $k = 2$	5	82.25	80.75	-1.50	80.75	76.75	83.25	+6.50
B	Fixed $k = 2$	6	90.00	82.50	-7.50	82.50	90.00	83.50	-6.50
B	Fixed $k = 2$	Six-block mean	86.29	83.04	-3.25	83.04	86.46	84.79	-1.67
B	Fixed $k = 4$	1	89.75	69.25	-20.50	69.25	95.75	67.75	-28.00
B	Fixed $k = 4$	2	91.25	79.00	-12.25	82.00	90.25	81.50	-8.75
B	Fixed $k = 4$	3	82.50	73.75	-8.75	73.75	83.50	75.50	-8.00
B	Fixed $k = 4$	4	82.00	71.25	-10.75	71.25	82.50	75.25	-7.25
B	Fixed $k = 4$	5	82.25	77.75	-4.50	77.75	76.75	81.00	+4.25
B	Fixed $k = 4$	6	90.00	79.00	-11.00	79.75	90.00	75.50	-14.50
B	Fixed $k = 4$	Six-block mean	86.29	75.00	-11.29	75.63	86.46	76.08	-10.38
B	S3	1	89.75	84.50	-5.25	84.50	95.75	89.50	-6.25
B	S3	2	91.25	88.25	-3.00	88.25	90.25	85.50	-4.75
B	S3	3	82.50	84.00	+1.50	84.00	83.50	83.75	+0.25
B	S3	4	82.00	85.50	+3.50	85.50	82.50	85.00	+2.50
B	S3	5	82.25	86.75	+4.50	86.75	76.75	80.25	+3.50
B	S3	6	90.00	83.25	-6.75	83.25	90.00	84.00	-6.00
B	S3	Six-block mean	86.29	85.38	-0.92	85.38	86.46	84.67	-1.79
$2B$	Long	1	85.25	85.25	0.00	85.25	NE	NE	NE
$2B$	Long	2	81.00	81.00	0.00	81.00	NE	NE	NE
$2B$	Long	3	87.25	87.25	0.00	87.25	NE	NE	NE
$2B$	Long	4	84.50	84.50	0.00	84.50	NE	NE	NE
$2B$	Long	5	85.75	85.75	0.00	85.75	NE	NE	NE
$2B$	Long	6	87.75	87.75	0.00	87.75	NE	NE	NE
$2B$	Long	Six-block mean	85.25	85.25	0.00	85.25	NE	NE	NE
$2B$	Fixed $k = 2$	1	85.25	89.50	+4.25	89.50	NE	NE	NE
$2B$	Fixed $k = 2$	2	81.00	83.75	+2.75	85.00	NE	NE	NE
$2B$	Fixed $k = 2$	3	87.25	87.00	-0.25	87.00	NE	NE	NE
$2B$	Fixed $k = 2$	4	84.50	91.00	+6.50	91.00	NE	NE	NE
$2B$	Fixed $k = 2$	5	85.75	86.50	+0.75	86.50	NE	NE	NE
$2B$	Fixed $k = 2$	6	87.75	88.00	+0.25	88.00	NE	NE	NE
$2B$	Fixed $k = 2$	Six-block mean	85.25	87.63	+2.38	87.83	NE	NE	NE

Table 11: M2: full paired delivery ledger. Absolute rates are percentages; differences are percentage points. NE means the pool was not evaluated.

Budget	Policy	Block	L_I	J_I	Δ_I	Max_I	L_R	J_R	Δ_R
B	Long	1	61.75	61.75	0.00	61.75	66.00	66.00	0.00
B	Long	2	58.75	58.75	0.00	58.75	59.75	59.75	0.00
B	Long	3	62.50	62.50	0.00	62.50	67.25	67.25	0.00
B	Long	4	61.00	61.00	0.00	61.00	66.00	66.00	0.00
B	Long	5	64.75	64.75	0.00	64.75	68.00	68.00	0.00

Continued on next page

Table 11 (continued)

Budget	Policy	Block	L_I	J_I	Δ_I	Max_I	L_R	J_R	Δ_R
B	Long	6	58.00	58.00	0.00	58.00	61.25	61.25	0.00
B	Long	Six-block mean	61.13	61.13	0.00	61.13	64.71	64.71	0.00
B	Fixed $k = 2$	1	61.75	49.25	-12.50	49.25	66.00	59.50	-6.50
B	Fixed $k = 2$	2	58.75	51.00	-7.75	52.25	59.75	58.75	-1.00
B	Fixed $k = 2$	3	62.50	53.50	-9.00	53.50	67.25	60.75	-6.50
B	Fixed $k = 2$	4	61.00	54.50	-6.50	54.50	66.00	55.25	-10.75
B	Fixed $k = 2$	5	64.75	53.25	-11.50	53.25	68.00	58.00	-10.00
B	Fixed $k = 2$	6	58.00	53.00	-5.00	53.00	61.25	60.00	-1.25
B	Fixed $k = 2$	Six-block mean	61.13	52.42	-8.71	52.63	64.71	58.71	-6.00
B	Fixed $k = 4$	1	61.75	50.25	-11.50	50.25	66.00	54.00	-12.00
B	Fixed $k = 4$	2	58.75	48.25	-10.50	52.75	59.75	55.25	-4.50
B	Fixed $k = 4$	3	62.50	48.75	-13.75	48.75	67.25	55.00	-12.25
B	Fixed $k = 4$	4	61.00	47.25	-13.75	47.25	66.00	53.25	-12.75
B	Fixed $k = 4$	5	64.75	46.00	-18.75	51.00	68.00	53.00	-15.00
B	Fixed $k = 4$	6	58.00	48.25	-9.75	51.00	61.25	53.00	-8.25
B	Fixed $k = 4$	Six-block mean	61.13	48.13	-13.00	50.17	64.71	53.92	-10.79
B	S3	1	61.75	59.25	-2.50	59.25	66.00	64.00	-2.00
B	S3	2	58.75	57.25	-1.50	57.25	59.75	59.75	0.00
B	S3	3	62.50	56.50	-6.00	56.50	67.25	61.75	-5.50
B	S3	4	61.00	57.25	-3.75	57.25	66.00	58.50	-7.50
B	S3	5	64.75	57.00	-7.75	57.00	68.00	59.00	-9.00
B	S3	6	58.00	57.50	-0.50	57.50	61.25	60.25	-1.00
B	S3	Six-block mean	61.13	57.46	-3.67	57.46	64.71	60.54	-4.17
$2B$	Long	1	71.75	71.75	0.00	71.75	71.00	71.00	0.00
$2B$	Long	2	66.25	66.25	0.00	66.25	64.00	64.00	0.00
$2B$	Long	3	65.75	65.75	0.00	65.75	68.75	68.75	0.00
$2B$	Long	4	65.75	65.75	0.00	65.75	71.25	71.25	0.00
$2B$	Long	5	59.50	59.50	0.00	59.50	62.00	62.00	0.00
$2B$	Long	6	71.00	71.00	0.00	71.00	73.00	73.00	0.00
$2B$	Long	Six-block mean	66.67	66.67	0.00	66.67	68.33	68.33	0.00
$2B$	Fixed $k = 2$	1	71.75	64.75	-7.00	64.75	71.00	67.75	-3.25
$2B$	Fixed $k = 2$	2	66.25	62.00	-4.25	64.75	64.00	69.50	+5.50
$2B$	Fixed $k = 2$	3	65.75	60.50	-5.25	62.00	68.75	65.25	-3.50
$2B$	Fixed $k = 2$	4	65.75	61.25	-4.50	63.50	71.25	62.00	-9.25
$2B$	Fixed $k = 2$	5	59.50	61.25	+1.75	61.25	62.00	65.75	+3.75
$2B$	Fixed $k = 2$	6	71.00	63.75	-7.25	64.00	73.00	67.75	-5.25
$2B$	Fixed $k = 2$	Six-block mean	66.67	62.25	-4.42	63.38	68.33	66.33	-2.00
$2B$	Fixed $k = 4$	1	71.75	56.25	-15.50	56.25	71.00	62.50	-8.50
$2B$	Fixed $k = 4$	2	66.25	54.00	-12.25	56.75	64.00	61.00	-3.00
$2B$	Fixed $k = 4$	3	65.75	54.25	-11.50	57.00	68.75	59.50	-9.25
$2B$	Fixed $k = 4$	4	65.75	58.25	-7.50	58.25	71.25	62.25	-9.00
$2B$	Fixed $k = 4$	5	59.50	58.25	-1.25	58.25	62.00	61.25	-0.75
$2B$	Fixed $k = 4$	6	71.00	58.50	-12.50	58.50	73.00	61.75	-11.25
$2B$	Fixed $k = 4$	Six-block mean	66.67	56.58	-10.08	57.50	68.33	61.38	-6.96

E.2 AUXILIARY CONTROLS

The parent short candidates use their original budgets and selection, not the new fixed-splitting recipe counts. They and the supervised starts are reported as auxiliary controls in Table 12, separate from the main allocation-policy comparison. For the starts, NA means no candidate-selection validation

is reported here; their zero PPO batches do not mean the supervised training was free. Only the parent selected short candidate of each block was reserved-pool evaluated.

Table 12: Auxiliary parent short candidates and supervised starts. V, I and R are successes out of 400; NE and NA retain the meanings above.

Instance	Control	Block	PPO batches	V	I	R	Role
M1	Short a	1	88	334	337	NE	Not selected
M1	Short b	1	88	338	328	316	Selected
M1	Short a	2	88	219	214	NE	Not selected
M1	Short b	2	88	319	324	317	Selected
M1	Short a	3	88	296	312	300	Selected
M1	Short b	3	88	291	283	NE	Not selected
M1	Short a	4	88	323	308	NE	Not selected
M1	Short b	4	88	327	335	337	Selected
M1	Short a	5	88	350	321	310	Selected
M1	Short b	5	88	324	288	NE	Not selected
M1	Short a	6	88	352	316	334	Selected
M1	Short b	6	88	338	330	NE	Not selected
M1	Supervised start	—	0	NA	168	169	Reference
M2	Short a	1	81	189	203	NE	Not selected
M2	Short b	1	81	207	201	223	Selected
M2	Short a	2	81	199	199	NE	Not selected
M2	Short b	2	81	210	210	218	Selected
M2	Short a	3	81	222	214	NE	Not selected
M2	Short b	3	81	225	234	237	Selected
M2	Short a	4	81	204	214	NE	Not selected
M2	Short b	4	81	225	219	232	Selected
M2	Short a	5	81	202	190	NE	Not selected
M2	Short b	5	81	221	215	225	Selected
M2	Short a	6	81	226	197	221	Selected
M2	Short b	6	81	217	194	NE	Not selected
M2	Supervised start	—	0	NA	183	218	Reference

F PER-BLOCK CONTRASTS

Tables 13 and 14 give the paired contrasts for M1 and M2. All values are in percentage points, with blocks 1–6 in order.

Table 13: M1, independent pool (top block of rows) and reserved pool (bottom block of rows), in pp.

contrast	b1	b2	b3	b4	b5	b6	mean
independent pool							
$k2-$ long	-5.75	-8.50	-0.25	$+4.00$	-1.50	-7.50	-3.25
$k4-$ long	-20.50	-12.25	-8.75	-10.75	-4.50	-11.00	-11.29
S3 $-$ long	-5.25	-3.00	$+1.50$	$+3.50$	$+4.50$	-6.75	-0.92
S3 $-$ $k2$	$+0.50$	$+5.50$	$+1.75$	-0.50	$+6.00$	$+0.75$	$+2.33$
S3 $-$ $k4$	$+15.25$	$+9.25$	$+10.25$	$+14.25$	$+9.00$	$+4.25$	$+10.38$
x1 $-$ long	-4.50	$\mathbf{-10.25}$	$+4.75$	$+2.50$	$+3.50$	-2.25	-1.04
x2 $-$ $k2$	$+5.50$	$+1.00$	$+4.75$	$+5.00$	$+5.75$	$+5.50$	$+4.58$
x2 $-$ x1	$+4.25$	$+2.75$	-0.25	$+6.50$	$+0.75$	$+0.25$	$+2.38$
interaction $k2$	$+10.00$	$+11.25$	0.00	$+2.50$	$+2.25$	$+7.75$	$+5.63$
reserved pool							
$k2-$ long	-8.00	-6.50	$+2.50$	$+2.00$	$+6.50$	-6.50	-1.67
$k4-$ long	-28.00	-8.75	-8.00	-7.25	$+4.25$	-14.50	-10.38
S3 $-$ long	-6.25	-4.75	$+0.25$	$+2.50$	$+3.50$	-6.00	-1.79
S3 $-$ $k2$	$+1.75$	$+1.75$	-2.25	$+0.50$	-3.00	$+0.50$	-0.13
S3 $-$ $k4$	$+21.75$	$+4.00$	$+8.25$	$+9.75$	-0.75	$+8.50$	$+8.58$

Table 14: M2, independent pool (top block of rows) and reserved pool (bottom block of rows), in pp. Block 5 is bold where it dominates a mean.

contrast	b1	b2	b3	b4	b5	b6	mean
independent pool							
$k2-$ long	-12.50	-7.75	-9.00	-6.50	-11.50	-5.00	-8.71
$k4-$ long	-11.50	-10.50	-13.75	-13.75	-18.75	-9.75	-13.00
S3 $-$ long	-2.50	-1.50	-6.00	-3.75	-7.75	-0.50	-3.67
x1 $-$ long	$+10.00$	$+7.50$	$+3.25$	$+4.75$	$\mathbf{-5.25}$	$+13.00$	$+5.54$
x2 $-$ $k2$	$+15.50$	$+11.00$	$+7.00$	$+6.75$	$+8.00$	$+10.75$	$+9.83$
x4 $-$ $k4$	$+6.00$	$+5.75$	$+5.50$	$+11.00$	$+12.25$	$+10.25$	$+8.46$
x2 $-$ x1	-7.00	-4.25	-5.25	-4.50	$+1.75$	-7.25	-4.42
x4 $-$ x1	-15.50	-12.25	-11.50	-7.50	-1.25	-12.50	-10.08
x4 $-$ long at B	-5.50	-4.75	-8.25	-2.75	-6.50	$+0.50$	-4.54
interaction $k2$	$+5.50$	$+3.50$	$+3.75$	$+2.00$	$\mathbf{+13.25}$	-2.25	$+4.29$
interaction $k4$	-4.00	-1.75	$+2.25$	$+6.25$	$\mathbf{+17.50}$	-2.75	$+2.92$
reserved pool							
$k2-$ long	-6.50	-1.00	-6.50	-10.75	-10.00	-1.25	-6.00
$k4-$ long	-12.00	-4.50	-12.25	-12.75	-15.00	-8.25	-10.79
S3 $-$ long	-2.00	0.00	-5.50	-7.50	-9.00	-1.00	-4.17
x1 $-$ long	$+5.00$	$+4.25$	$+1.50$	$+5.25$	$\mathbf{-6.00}$	$+11.75$	$+3.63$
x2 $-$ $k2$	$+8.25$	$+10.75$	$+4.50$	$+6.75$	$+7.75$	$+7.75$	$+7.63$
x4 $-$ $k4$	$+8.50$	$+5.75$	$+4.50$	$+9.00$	$+8.25$	$+8.75$	$+7.46$
x2 $-$ x1	-3.25	$+5.50$	-3.50	-9.25	$+3.75$	-5.25	-2.00
x4 $-$ x1	-8.50	-3.00	-9.25	-9.00	-0.75	-11.25	-6.96
x4 $-$ long at B	-3.50	$+1.25$	-7.75	-3.75	-6.75	$+0.50$	-3.33
interaction $k2$	$+3.25$	$+6.50$	$+3.00$	$+1.50$	$\mathbf{+13.75}$	-4.00	$+4.00$
interaction $k4$	$+3.50$	$+1.50$	$+3.00$	$+3.75$	$\mathbf{+14.25}$	-3.00	$+3.83$

G LEAVE-ONE-BLOCK-OUT ROBUSTNESS

Each contrast was recomputed six times per instance and pool, deleting one block each time (`analyze_loo.py`); Table 15 classifies every headline contrast. **stable** = all six variants keep the sign and exclude zero; **fragile** = the six-block interval excludes zero but at least one variant's covers it, with those blocks in parentheses; **covers zero** = the six-block interval already covers zero. A dagger marks a cell in which some deletion reverses the sign of the point estimate, a double dagger one in which a deletion makes it exactly zero; the blocks are named in the notes. Interval endpoints are snapped to the attainable grid of $0.25/b$ pp before zero is tested.

Table 15: Leave-one-block-out classification of every headline contrast.

contrast	M1 indep.	M1 reserved	M2 indep.	M2 reserved
$k4-$ long	stable	stable	stable	stable
S3 $-$ $k4$	stable	stable	stable	stable
$k2-$ long	fragile (b1,b2,b6)	covers zero	stable	stable
S3 $-$ long	covers zero†	covers zero	stable	stable
S3 $-$ $k2$	fragile (b1,b2,b3,b5,b6)	covers zero†	stable	fragile (b1,b4)
x1 $-$ long	covers zero†	not run	stable	stable
x2 $-$ $k2$	stable	not run	stable	stable
x4 $-$ $k4$	not run	not run	stable	stable
x4 $-$ long at B	not run	not run	stable	stable
x4 $-$ x1	not run	not run	stable	stable
x2 $-$ x1	fragile (b1,b2,b4)	not run	stable	covers zero
interaction $k2$	**stable**	not run	fragile (b5)	fragile (b5)
interaction $k4$	not run	not run	covers zero‡	fragile (b5)

"not run" marks a cell whose arm was never executed, not one awaiting results. † sign reverses under a single deletion: M1 indep. S3 $-$ long without b6 ($-0.92 \to +0.25$ pp); M1 reserved S3 $-$ $k2$ without b3 ($\to +0.30$) or b5 ($\to +0.45$); M1 indep. x1 $-$ long without b2 ($-1.04 \to +0.80$). ‡ mean becomes exactly zero without b5; its interval also covers zero when b2, b3, b4 or b5 is dropped.

Block-exclusion sensitivity for the incident blocks. Excluding the two blocks that contain a manually re-run candidate, M1 $k4-$ long is -13.06 pp $[-18.13, -8.19]$ on the independent pool and -13.00 pp $[-17.69, -8.56]$ on the reserved pool, so the confirmatory conclusion is unchanged. Excluding the block containing the automatically re-run stem, M1 S3 $-$ $k2$ is $+2.45$ pp $[-0.30, +5.20]$ and $+0.30$ pp $[-2.50, +3.10]$: the S3-over-$k2$ evidence for M1 is again weaker than its headline. These checks cannot restore the discarded first attempts or prove the protocol filter is unbiased.

Block-level dispersion. Table 16 summarizes the six paired block values behind each headline contrast: mean, sample standard deviation, range and the number of blocks whose sign agrees with the six-block mean, beside the game-cluster interval from the same snapshot. The interval describes evaluation-game sampling for the trained models; the block spread describes variation across the six training blocks. Six blocks do not support an interval for a new training campaign, so the block columns are descriptive, and the sign counts together with Table 15 are the appropriate reading of them: a contrast with six agreeing blocks and a stable deletion class rests on every block, whereas one with a block on the other side of zero and a fragile class rests on the six-block average.

Table 16: Block-level dispersion of the headline contrasts (pp). Each row summarizes the six paired block values behind one headline contrast of Section 4: their mean, sample standard deviation across blocks, range, and the number of blocks whose sign agrees with the six-block mean (a block at exactly zero does not agree). The interval is the game-cluster 95 % bootstrap interval conditional on the trained models, copied from the same snapshot. The block columns describe variation across the six training blocks; with six blocks they are descriptive and are not an interval for a new training campaign. Table 15 gives the deletion sensitivity of the same contrasts.

contrast	instance, pool	mean	SD	min	max	same sign	95 % interval
$k4-$ long at B	M1 indep.	-11.29	5.27	-20.50	-4.50	6/6	$[-15.79, -7.04]$
	M1 reserved	-10.38	10.58	-28.00	+4.25	5/6	$[-14.08, -6.83]$
	M2 indep.	-13.00	3.26	-18.75	-9.75	6/6	$[-17.00, -9.13]$
	M2 reserved	-10.79	3.77	-15.00	-4.50	6/6	$[-14.25, -7.46]$
S3 $-$ $k4$ at B	M1 indep.	+10.38	3.99	+4.25	+15.25	6/6	$[+7.08, +14.00]$
	M1 reserved	+8.58	7.52	-0.75	+21.75	5/6	$[+5.75, +11.63]$
	M2 indep.	+9.33	1.09	+7.75	+11.00	6/6	$[+6.08, +12.79]$
	M2 reserved	+6.63	1.93	+4.50	+10.00	6/6	$[+4.17, +9.21]$
$k2-$ long at B	M1 indep.	-3.25	4.83	-8.50	+4.00	5/6	$[-6.00, -0.71]$
	M1 reserved	-1.67	6.07	-8.00	+6.50	3/6	$[-4.38, +1.04]$
	M2 indep.	-8.71	2.89	-12.50	-5.00	6/6	$[-11.71, -5.96]$
	M2 reserved	-6.00	4.16	-10.75	-1.00	6/6	$[-8.17, -4.04]$
x2 $-$ x1 at $2B$	M1 indep.	+2.38	2.64	-0.25	+6.50	5/6	$[+0.17, +4.63]$
	M2 indep.	-4.42	3.27	-7.25	+1.75	5/6	$[-6.83, -2.13]$
	M2 reserved	-2.00	5.59	-9.25	+5.50	4/6	$[-4.08, +0.04]$
interaction $k2$ $(2B - B)$	M1 indep.	+5.63	4.65	0.00	+11.25	5/6	$[+2.33, +8.92]$
	M2 indep.	+4.29	5.11	-2.25	+13.25	5/6	$[+0.96, +7.67]$
	M2 reserved	+4.00	5.88	-4.00	+13.75	5/6	$[+1.29, +6.75]$
x4 $-$ x1 at $2B$	M2 indep.	-10.08	5.03	-15.50	-1.25	6/6	$[-13.29, -7.13]$
	M2 reserved	-6.96	4.11	-11.25	-0.75	6/6	$[-9.42, -4.54]$

H Extended experimental and statistical protocol

H.1 Setup

Task family. TextWorld games (Côté et al., 2018) whose verified reference solution has length $H_{\text{ref}} = 8$, with a hard cap of 24 actions per episode—a safety limit, not the number of steps a task needs. Training uses 64 fixed games in batches of 8 distinct games drawn by a seeded permutation without replacement, so eight batches cover the training pool exactly once.

Three disjoint evaluation pools, each 100 games $\times$ 4 repeats $=$ 400 episodes, each with its own seed base: validation (seed base 3101, ids `h8-validation-`$\star$) for candidate selection only, executed inside the training framework with vLLM; independent (4101, `h8-exploration_audit-`$\star$) for all descriptive results, executed in a fresh Transformers process from the verified final files; and reserved (5101, `h8-reserved_test-`$\star$), opened once, for the single confirmatory test of Section 4.1 and its descriptive companion table.

Selection uses validation successes only, ties to the lowest candidate index, and the group is sealed—the winner recorded, the file hashed—before any independent-pool request is issued. For the fixed-k arms every candidate, winner and loser, is then evaluated on the independent pool. The successive-halving arm is the one exception, by registration: its two stems are validated but never evaluated on the independent pool, and only the continuation is evaluated (Section H.2).

Two model instances, one recipe. M1 starts from a supervised fine-tune of Qwen2.5-1.5B-Instruct (Qwen Team, 2024); M2 from the same supervised recipe applied to SmolLM2-1.7B-Instruct (Ben Allal et al., 2025). That supervised stage uses an isolated demonstration pool of $H_{\text{ref}} = 4$ games, disjoint from the H8 games used for PPO training and from all three evaluation pools (64 games $\times$ 4 reference actions $=$ 256 supervised samples, seed 820001, 3 epochs, 96 optimizer steps, AdamW at 10^{-5}); the two instances see the same demonstration messages, which

their different tokenizers turn into different token sequences. The intended differences between the two instances are the base model, its tokenizer and chat template, and the resulting supervised start; everything else is shared. Two instances are not two model families in any general sense, and we never pool them.

All PPO runs (Schulman et al., 2017) use GAE (Schulman et al., 2016) with $\gamma = 1$, $\lambda = 1$, a sparse terminal 0/1 reward with a KL penalty coefficient of 0.01 reported separately, one PPO epoch per batch, mini-batch 8, micro-batch 1, actor learning rate 10^{-6} and critic learning rate 10^{-5} on cosine schedules with no warm-up, and critic-only updates for the first two batches—so a run of N batches performs $N - 2$ actor updates and N critic updates.

Only a run's fixed final checkpoint is eligible for selection. For the fixed-k arms each run reloads the same supervised start with a fresh optimizer, schedule and random state, inherits nothing from any other run, and its cosine schedule spans its own batch count. The successive-halving arm differs on every one of these points and is defined separately in Section H.2: its stems are scheduled over the full potential horizon T from the first batch, and the continuation resumes the winning stem's parameters, optimizer state, RNG state and absolute batch order rather than restarting—it does not repeat the two critic-only batches.

Generation uses a 4096-token context, at most 128 tokens per action, temperature 0.7, top-p 1, top-k disabled and one sample per request; evaluation seeds are derived from split, game, repeat and action indices, training seeds from the run's training seed, phase and absolute global step. Training runs on verl (Sheng et al., 2025) with the recipe lineage of Wang & Ammanabrolu (2025).

Blocks and arms. Six blocks per instance, each with pre-registered seeds. Each block contributes a long run ($k = 1$) and the candidate sets of the split arms; the long run and a "short" reference run come from the parent batches and are reused unchanged, never retrained and never re-tuned. The arms are: long ($k = 1$ at budget B), k2/k4 (validation winner among k candidates at B), s3 (two stems, one continuation, at B; Section H.2), and x1/x2/x4 (the same policies at $2B$).

Operational verification. Run-input verification, checkpoint/resume acceptance, and related production-log details are recorded in Appendix C.

H.2 Successive-halving arm

The arm. Two stems of n_0 batches are trained from the common start and validated; the stem with more validation successes is reloaded and continued to T batches; only the continuation is evaluated on the independent pool and only it is recovered and delivered. This is the smallest adaptive allocation in the successive-halving family of Jamieson & Talwalkar (2016), on the breadth-versus-depth axis that Li et al. (2018) make explicit—promotion depends on an intermediate validation result, so the allocation is adaptive; what is fixed in advance is the stage lengths and the promotion rule—and it is the closest arm in this paper to the "start many, stop early, continue some" strategy of Dodge et al. (2020), with the validation cost charged to the budget.

H.3 Cost model

H.3.1 Components

A calibration run of 16 batches fixes, per instance: u, seconds per training batch, taken as $1.25\times$ the mean wall-clock of batches 9–16 (a margin, frozen before use); c_1, seconds for one 400-episode validation pass, $1.25\times$ the measured pass; and the overhead components I (process start-up), S (final save), R (reload verification), F (packaging, transfer and local verification of one model's files, i.e. "recovery") and d (that run's share of the one-time deployment window). Measured values are in Appendix A.

H.3.2 Two accountings, and why the permissive one is the one we run

Accounting A (per delivery). Every candidate pays the whole per-run overhead $o_1 = I + S + R + F + d$, so $N_A(k) = \lfloor (B/k - c_1 - o_1)/u \rfloor$.

Accounting B (single delivery). Every candidate pays $q = I + S + R + d$ and its validation c_1; the recovery and delivery cost F is paid once, by the model actually delivered: $N_B(k) = \lfloor ((B - F)/k - c_1 - q)/u \rfloor$.

The two agree at $k = 1$. The per-candidate overhead under A is $o_1 + c_1 = 928.87$ s for M1 and 887.67 s for M2, i.e. 25.8 % and 24.7 % of a $B = 3600$ s budget. Under B it is $q + c_1 = 362.09$ s and 400.58 s (10.1 % and 11.1 %), with F charged once. The full derived grid, including unexecuted cells, is in Appendix A. Affordable candidate counts follow from these overheads before training.

Three consequences, all arithmetic. First, under accounting A the registered grid $\{1, 2, 4, 8\}$ caps the affordable candidate count at 2: at $k = 4$ fewer than the required 16 training batches remain. This is a property of the pipeline's overheads, not a discovery about learning; an intermediate $k = 3$, which the registered grid does not contain, would still train 27 batches per candidate in both instances. Second, we run everything under accounting B, which is strictly more generous to splitting and makes $k = 4$ feasible; any deficit we measure at $k = 4$ is measured under the accounting that favours the arm we find losing. Third, at $B = 3600$ s, $k = 8$ is infeasible even under B, which is why the executed grid stops at $k = 4$; at $2B$ it would be feasible (47 and 39 batches) but we did not run it.

H.3.3 Frozen batch counts and the successive-halving budget

Batch counts are frozen before the corresponding production stage starts and never adjusted for wall-clock, spend or performance. The fixed-k counts were frozen before the first training run; the successive-halving counts were frozen after the k arms had begun but before any successive-halving run, because they depend on a resume cost that had to be measured on the GPU first. No count was changed after the stage it governs had started.

The successive-halving arm pays a different overhead, registered as $O = 2I + 3S + R + F + 2d + 2c_1 + \text{resume}$ with $\text{resume} = I+$ measured checkpoint load (201.64 s for M1, 163.71 s for M2): two stems are started, saved and validated, the better one is reloaded and continued to T, and only the continuation is recovered and delivered. Total overhead is 1495.08 s (M1) and 1458.85 s (M2) of the same $B = 3600$ s.

Reference delivery budget, not equal wall-clock. B fixes how many batches an arm may train under the rule above. It is not a wall-clock stopwatch and not the cash cost; the physical cost of the campaign is reported separately in Appendix B, and it is larger.

H.4 Measurements and inference

For a block b let p_L be the independent-pool success rate of that block's long run, $p_1 \dots p_k$ the rates of the k candidates of a split arm, and J the index the validation pool selected. All quantities are per block, then averaged over the six blocks of an instance; differences are in percentage points (pp) of episode success rate.

Primary quantity. The delivery difference $D_b = p_J - p_L$: what the practitioner actually gets, minus what the long run would have given.

Bookkeeping, not mechanism. With $C_b = \text{mean}_i\, p_i - p_L$ (candidate-mean gap) and $G_b = p_J - \text{mean}_i\, p_i$ (selection gain), $D = C + G$ is an identity. It is useful for reporting, but "C is more negative than D" only restates $G > 0$; it cannot establish that the deficit is caused by shorter training rather than by selection error, because the arms differ in more than training length. Likewise $G \le \max_i p_i - \text{mean}_i\, p_i$ holds by definition, and at $k = 2$ the point estimate of G is exactly half the candidate difference—an identity, not a law about seeds.

The informative pair. To assess the room for better selection within each observed candidate set, we report the empirical best-candidate gap $\max_i p_i - p_{\text{ref}}$—could the arm have beaten the $k = 1$ delivery at its own budget, if an oracle had picked its best candidate on the independent pool itself?—and the empirical selection shortfall $\max_i p_i - p_J$—how much did the validation pick cost relative to that oracle? Both are post-hoc and optimistic by construction ($\max_i$ on the same pool is itself noisy, and is not a deployable rule).

Paired contrasts. Every comparison below is a paired per-block difference of the two arms being compared, then averaged—never a comparison of two separately reported intervals, which does not test their difference.

Uncertainty. 10000 game-cluster bootstrap resamples drawing the same 100 games for every model in an instance, with nearest-rank intervals; resampling seeds were registered in advance. These intervals describe game-sampling uncertainty conditional on the policies we actually trained. With one long run and k candidates per block, training-seed uncertainty is represented only by the six-block spread. Following Agarwal et al. (2021) and Patterson et al. (2024) we report direction, spread, per-block sign counts and intervals rather than a single number, and we avoid normal-approximation intervals on 400-episode pools (Bowyer et al., 2025).

Registration. Everything outside Section 4.1 is descriptive. Section 4.1 is the single confirmatory test, whose rule—pool identity, seed base, model-list definition, statistic, one-sided direction and Holm correction over the two instances—was fixed before any group was sealed and before any independent-pool evaluation.

I Extended comparisons, registration and incident records

This section records the detailed comparisons; Appendix E gives every candidate's sealed validation count, available independent and reserved outcomes, delivery identity, and the full paired per-block results, including six-block means and auxiliary controls. Unmeasured cells are explicitly marked, not imputed.

I.1 Fixed-k delivery at the reference budget

Delivery difference D on the independent pool, mean over six blocks: M1 -3.25 pp $[-6.00, -0.71]$ at $k = 2$ and -11.29 pp $[-15.79, -7.04]$ at $k = 4$; M2 -8.71 pp $[-11.71, -5.96]$ and -13.00 pp $[-17.00, -9.13]$.

For fixed splitting on the independent pool, 23 of the 24 per-block contrasts are negative. The single exception is M1 block 4 at $k = 2$ ($+4.00$ pp, interval covering zero). Both $k = 4$ contrasts exclude zero in both instances and on both pools, and survive every leave-one-block-out deletion (Section I.6). The $k = 2$ result is directionally the same but weaker, and for M1 it is fragile.

I.2 Same-pool best-candidate diagnostic

The diagnostic compares each fixed-splitting arm with the $k = 1$ delivery at its own budget—the parent long run for the B arms, that block's x1 for the $2B$ arms.

At $k = 4$, in every block of both instances, even the empirical maximum on the independent pool itself remained below the long run, by 10.7 and 11.0 pp on average, while the validation rule gave away a further 0.63 pp (M1) and 2.04 pp (M2) on average—up to 3.0 and 5.0 pp in the worst single block. The selection losses are real but small next to the gap they would have to close.

Within these candidate sets and on this independent pool, selecting the best candidate after the fact would not have reached the long run at $k = 4$. Selection loss exists and is measured above, a better selection rule could help in other settings, and the arms differ in candidate set, training stochasticity and learning-rate timeline together, so the deficit is not attributed to training length alone. For completeness, the selection gains the identity of Section H.4 accounts for are $G = +3.6$ pp (M1 $k = 2$), $+9.6$ pp (M1 $k = 4$), $+1.6$ and $+1.0$ pp (M2). M1's $k = 4$ selection gain is not small; it is simply not large enough, because the candidates it selects among are all far below the long run.

I.3 Successive halving

The paired comparisons support three observations. (i) S3 beats fixed splitting at $k = 4$ everywhere—both instances, both pools, every leave-one-block-out variant.

(ii) Against $k = 2$ the answer depends on the instance and the pool. For M2 it is positive on both pools. For M1 it is $+2.33$ pp on the independent pool but -0.13 pp on the reserved pool, where

the descriptive means also put $k = 2$ marginally ahead (.8479 vs .8467); the independent-pool advantage stays positive but its interval covers zero in 5 of 6 leave-one-block-out variants. There is therefore no reproduced ordering "long $\geq$ S3 $>$ k2 $>$ k4" across both instances and both pools.

(iii) Relative to the long run, S3 is lower for M2 on both pools; for M1 the interval covers zero on both pools, which means the data allow a small advantage or a small disadvantage; it is not evidence of equivalence, because no equivalence margin and no equivalence test were pre-specified.

Per block, the continued stem was the second stem in 3 of 6 blocks in each instance, so the validation pick is not degenerate.

I.4 Doubling the budget

Both instances were run at $2B$, but not identically: M2 covers $k \in \{1, 2, 4\}$ on both pools, while the M1 unit was registered later for $k \in \{1, 2\}$ on the independent pool only, and adds no model to the reserved list. Two questions have to be kept apart: how much each policy gains from the doubled budget, and where the policies end up relative to each other.

Gains—doubling the budget helps splitting more than it helps the long run, in both instances. The same-strategy increment from B to $2B$, paired per block, is for M2 $+5.54$ $[+3.38, +7.79]$ for the single long run (x1 $-$ long), $+9.83$ $[+6.54, +13.29]$ for two candidates (x2 $-$ k2) and $+8.46$ $[+5.42, +11.71]$ for four (x4 $-$ k4) on the independent pool, and $+3.63$, $+7.63$, $+7.46$ on the reserved pool. For M1 the two-candidate gain is $+4.58$ $[+2.25, +7.00]$, while the long-run increment x1 $-$ long is -1.04 $[-3.67, +1.50]$, an interval covering zero whose sign reverses if block 2 is deleted. Every one of these gains except that last one survives all six leave-one-block-out deletions.

The consequence is the budget interaction $[x_k - x1] - [k - \text{long}]$, which at $k = 2$ is positive and excludes zero in both instances: $+4.29$ $[+0.96, +7.67]$ for M2 on the independent pool, $+4.00$ $[+1.29, +6.75]$ on its reserved pool, and $+5.63$ $[+2.33, +8.92]$ for M1. The two are not equally robust to deletion: M1's excludes zero under every single-block deletion, M2's does not on either pool once block 5 is removed (below, and Section I.6). A doubled budget therefore measurably narrows the gap that Section I.1 opens at $k = 2$ in both instances, on the six blocks we ran.

Final rankings and the cheaper long-run reference. We distinguish the same-budget ranking from comparisons against the long run at B.

Robust, for M2: more total compute does not automatically beat a cheaper long run. Four candidates trained at twice the budget deliver 3.3–4.5 pp below a single long run at half of it, on both pools, surviving every leave-one-block-out deletion, and at equal budget the ordering x1 $>$ x2 $>$ x4 is unchanged from B. We did not run $k = 4$ for M1 at $2B$ (its $k = 4$ arm at B is in Section I.1), nor successive halving at $2B$ in either instance, so this is a statement about the fixed-k policies we tested on M2, not a ranking over all candidate strategies at that budget.

Where the instances part company. M2's x2 $-$ x1 is clearly negative on the independent pool, while M1's six-block estimate is $+2.38$ $[+0.17, +4.63]$: the observed ordering at $2B$ differs between the two instances we tested. M1's $+2.38$ is fragile—its interval covers zero when block 1, 2 or 4 is deleted, though the point estimate stays positive in all six variants ($+1.55$ to $+2.90$)—and the companion estimate x1 $-$ long, -1.04, covers zero and reverses sign without block 2. The long-run interval permits both an increase and a decrease, and the design does not isolate an effect of model identity or architecture. The observed ordering differs between the two instances, and M1's advantage is sensitive to which blocks are included.

Also fragile: how large the interaction is for M2. At $k = 2$ M2's interaction excludes zero on both pools, and at $k = 4$ only on the reserved pool—but it rests materially on one block. Block 5 is the only M2 block where the $2B$ long run underperforms the B long run (x1 $-$ long $= -5.25$ pp independent, -6.00 pp reserved), which inflates the interaction there ($+13.25$ and $+13.75$ at $k = 2$). Delete block 5 and M2's $k = 2$ interaction falls to $+2.50$ $[-1.30, +6.45]$ and $+2.05$ $[-1.10, +5.30]$. M1's interaction has no such dependence: it stays within $+4.50$ to $+6.75$ and excludes zero under every deletion.

We report the full six-block estimates as the results and these deletions as sensitivities: dropping the block that contributes the largest positive term to this mean necessarily lowers this mean, so the

check shows where an estimate's weight sits, not what an unbiased estimate would be. A doubled budget narrows the $k = 2$ deficit in both instances, the size of that narrowing is less sensitive to single-block deletion for M1 than for M2, and splitting's deficit at $2B$ does not disappear for M2, while for M1 it reverses in the six-block estimate but not robustly to single-block deletion.

I.5 THE REGISTERED CONFIRMATORY TEST

Statistic and rule, registered in advance. D = mean over the six blocks of $p_J(k{=}4 \text{ winner}) - p_L$ (parent long) on the reserved pool; one-sided H_4: $D < 0$; game-cluster bootstrap with 10000 resamples at registered seeds; Holm correction over the two instances at level 0.05. Result: M1 $D = -10.38$ pp $[-14.08, -6.83]$ and M2 $D = -10.79$ pp $[-14.25, -7.46]$, both with a registered tail proportion of $1/10001$, Holm-adjusted ≈ 0.0002, both meeting the registered rule.

Three qualifications apply. First, the registered quantity $(1 + \#\{D^* \geq 0\})/10001$ is an uncentered bootstrap tail proportion with a smoothing term, not a p-value with guaranteed finite-sample calibration; registering a formula does not change its statistical nature. A centred (null-shifted) resampling sensitivity gives 0 of 10000 tail events for both instances, so the direction is unchanged, and the registered procedure remains the primary record. We therefore state the result as "no resample of the 100 reserved games reached zero" rather than as a precise small probability.

Second, the timestamps behind "registered in advance": the rule was written at a point when no group had been sealed, no independent-pool evaluation had run and the reserved pool had not been touched, but one validation-pool count from the pipeline's smoke run (206/400 for a single $k = 4$ candidate) already existed. Third, the 80 model identities were instantiated in two waves as the corresponding groups sealed, by that rule, each model sealed before its own reserved evaluation; the evidence supports "rule fixed first, list generated by the rule", not the stronger statement that the complete 80-identity file existed before the pool was opened.

Descriptive reserved-pool levels include the auxiliary short references and supervised starts (Appendix D).

I.6 ROBUSTNESS

Every headline contrast was recomputed six times per instance and pool, each time deleting one block, and sorted into three states: covers zero already, when the six-block interval itself covers zero; stable, when all six variants keep the sign and exclude zero; and fragile, when the six-block interval excludes zero but at least one variant's interval covers it.

Zero has to be detected exactly. Success rates are multiples of $1/4$ per game and $1/400$ per pool, so a mean over b blocks of paired differences lies on a grid of $0.25/b$ pp and so do the bootstrap interval endpoints; nothing between grid points is attainable. We snap endpoints to that grid before testing whether an interval contains zero. The snapping matters: the $k = 4$ budget interaction with block 2 deleted has a lower endpoint that is mathematically zero but is stored as 2.8×10^{-16}, and a naive comparison reads it as excluding zero.

Table 15 gives the complete classification, and Sections I.3 and I.4 discuss the comparisons sensitive to block deletion.

Deletion can flip a sign, and does. Three cells reverse direction under a single deletion, all of them already interval-covering zero at six blocks: M1's S3 − long on the independent pool (-0.92 pp over six blocks, $+0.25$ pp without b6), M1's S3 − $k = 2$ on the reserved pool (-0.13 pp, becoming $+0.30$ without b3 and $+0.45$ without b5), and M1's x1 − long on the independent pool (-1.04 pp, becoming $+0.80$ without b2). Separately, M2's $k = 4$ interaction on the independent pool falls to exactly zero without b5, while its interval already covers zero in the full six-block estimate. We report these rather than describe leave-one-out as affecting intervals only. Leave-one-out is a post-hoc sensitivity analysis; the full six-block estimate stays the reported result in every case. Appendix G gives the complete table.

I.7 INCIDENTS AND WHAT THEY CONDITION ON

Three runs were stopped and re-run. All three are model-output protocol exceptions—an action ended with the base model's end-of-text token instead of the protocol end token, which the rollout refuses fail-closed—not infrastructure faults; two were stopped and re-run manually with written reasons recorded before the re-run, and one was re-run automatically under a single-retry rule registered after the second incident.

Four properties of these re-runs qualify the results. Each first attempt trained to completion and produced a partial validation record before the stop—256 episodes for each manual case and 32 for the automatic one, which stopped in its second 32-request validation group—so none of the three has a complete validation summary, though each has some validation data. The re-runs reproduced the binding (identical configuration and inputs) but not identical weights or trajectories: comparing the two attempts of the manual cases, critic scores differ at 21/40 and 22/40 steps and the actor loss differs at all 38 update steps. The re-run candidate became the block winner in one case, was not selected in the second, and the auto-retried stem was not continued.

Finally, the procedure conditions on a candidate completing the protocol, which is a selection filter whose effect the data cannot bound; Appendix G reports the block-exclusion sensitivity in place of an assumption that the effect is negligible. The weights of the three first attempts were not retained, so the discarded candidates cannot be re-evaluated under a common failure-scoring rule; for these attempts, the block-exclusion sensitivity of Appendix G is the check completed so far. Every extra attempt and wait is in the physical cost.

A fourth stop, during the $2B$ unit of the first instance, was of a different kind and is reported separately because that retry discarded no trained candidate and used no training or validation outcome to select one: a run died 17.5 s in, at the pre-training GPU check, with a CUDA driver initialisation error, having produced no run directory, no batches and no validation episodes. The GPU was healthy again when inspected minutes later, and the run was started again under a byte-identical input binding.

These properties are taken from the retained incident record; its copy of the failed attempt was moved to a Pod-side archive that was not retrieved before shutdown, while the retry's binding, plan and marker are local. Nothing was trained and nothing was discarded, so unlike the three protocol stops this one introduces no selection filter—a statement about this stop, not a reassessment of those three. We record it so that the run counts below are reproducible from the markers.

Run accounting: 168 successful training executions plus 3 failed protocol attempts and 1 aborted start; of these, 132 ordinary runs and 24 stems carry a final validation, and the 12 continuations are not validated again, so "168 runs with a 400-episode validation" would be wrong.

J REPRODUCTION

Inputs. Two groups. *For every statistical table:* the per-episode evaluation summaries of the independent and reserved pools (one JSON per model, each listing 400 episodes with game id, repeat and outcome), the group selection seals that fix each arm's delivered model, and the parent batches' evaluation audits for the long runs at B. *Additionally for the cost table:* each Pod's per-run completion markers and per-stage command timings, and the Pod registry giving each Pod's hourly rate and billing start; the 404.80 USD rental-window figure is summed from those billing starts and is not something the analysis scripts derive. Environment: Python 3.13 with `numpy`; no GPU and no network access are required.

```
OUT=review-full; mkdir "$OUT";  S=scripts/sprint
M=artifacts/sprint-results/merged
for F in m1 m2; do
  python $S/analyze_k.py --model $F \
      --evaluations-dir $M/evaluations --seals-dir $M/seals \
      --output $OUT/analysis-$F.json
  python $S/analyze_s3.py --model $F --output $OUT/analysis-s3-$F.json
done
python $S/analyze_contrasts.py --output $OUT/contrasts.json
python $S/analyze_reserved.py --output $OUT/reserved.json
```

```
python $S/analyze_loo.py       --output $OUT/loo.json
python $S/cost_table.py --cutoff-utc 2026-09-12T19:20:00Z \
                        --output $OUT/cost.json
```

`analyze_k` produces the per-arm D, C and G of Section 3.3; `analyze_contrasts` produces every paired contrast and the best-candidate diagnostic in Appendix D, recording for each arm which $k = 1$ delivery is its same-budget reference; `analyze_reserved` performs the registered test of Section 4.1 with the Holm correction; `analyze_loo` produces the classification of Section 4.5 and Appendix G together with the incident-block exclusions of this appendix; `cost_table` builds Appendix B. The cutoff shown is the one behind the reported table and is required: without it the scan also charges every later unit present in the same directory. In the commands as shown, `$M` is read only by `analyze_k`; the other five take their default input tree, so changing `$M` alone does not redirect the whole analysis.

Overwrite behaviour. The five analysis scripts open their `--output` with exclusive creation and fail rather than overwrite, so `$OUT` must not already contain them; `cost_table.py` currently writes its output unconditionally. Use a fresh output directory.

Expanded table ledgers. The complete appendix ledgers and machine-readable delivery summary are generated by `reports/iclr2027-sprint/comprehensive-tables-20260914/build_tables.py`. It reads the same sealed model identities, completed evaluation summaries and parent audits, verifies every observed count, and checks the paired point estimates and empirical maxima against the frozen S1 snapshot. Its intervals and LOO states are copied from the bound S1/S2 snapshots, not re-estimated. The generated JSON/CSV ledgers retain full model IDs, and the source-verification report records hashes of all inputs. It adds no training or evaluation requests. Table 16 is produced by `reports/iclr2027-sprint/block-dispersion-20260916/build_block_table.py` from the frozen contrast snapshot; every cell is arithmetic on the six per-block differences already in that snapshot.

What was checked. Running the commands above into a fresh directory reproduced the published contrast and cost snapshots byte-identically on the platform that produced them; the scripts write JSON with the platform's line endings, so on another platform the archive's `compare_snapshots.py` compares the parsed values instead of the digests. The supplementary archive contains the scripts, the sealed inputs, the four snapshots and an index (`reproduction-index.md`) listing each snapshot's SHA-256 digest and byte size and the digest of every script; running the commands above from the archive root reproduced all four snapshots. The two test modules in the archive check the cells of Tables 4, 6, 5, 15 and 16, the run counts of Appendix I.7 and the cost increment of Appendix B against those snapshots, so a reader can compare digests and rerun the tests.