WHERE TO SPEND THE INTERVENTION BUDGET: UNDERSTANDING AND USING PREINTERVENTION INFORMATION

Anonymous authors*
Paper under double-blind review

ABSTRACT

An uncertain decision is worth revisiting only if an alternative action, a repair, can improve its outcome. With a limited budget, an agent must choose which decisions to evaluate before it knows what their alternatives will achieve. We study which available information helps make this choice, measured by Capture: the share of an oracle's repair value that a selection obtains under the same budget. Across four synthetic retrieval settings and seven seeds, features that describe the candidate alternatives yield 0.4266 Capture, compared with 0.2669 for retrieval features when both feed the same value regressor. Predicting whether a useful repair exists reaches 0.5078; nearest neighbor regression on similar, previously evaluated decisions reaches 0.6235. These findings motivate Cue Informed Allocation (CIA), which estimates repair value and whether a repair exists, chooses a scoring rule on previously evaluated decisions, and ranks new decisions for evaluation. At a 20% budget, it reaches 0.6316 Capture, mostly owing to the nearest neighbor estimate. We also compare eight policy learning methods adapted to the same allocation task and examine recorded outcomes from interactive tasks. Candidate features and outcomes of similar decisions account for most of the gain; combining predictions and choosing rules add smaller gains that depend on the setting.

1 INTRODUCTION

Learning from an agent's failure requires knowing what could have worked instead. Testing alternative actions, which we call an intervention, consumes model continuations, tool calls, or human judgments. A limited budget therefore forces a choice about which decisions to revisit. Hindsight and counterfactual credit assignment methods use observed outcomes to attribute success or failure to decisions (Pignatelli et al., 2023; Mesnard et al., 2021). We study the earlier step of choosing *which decisions to evaluate in search of a better action*.

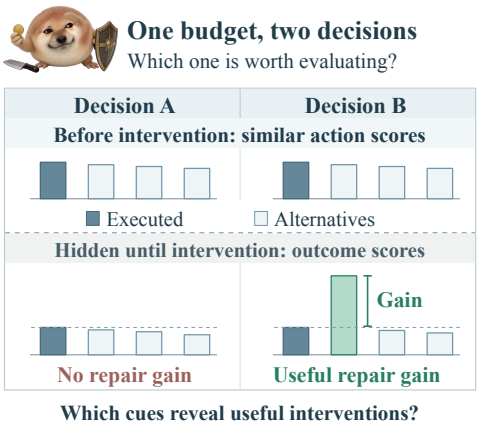

Figure 1: Equal confidence, unequal repair value (schematic).

Uncertainty alone does not settle this choice. Figure 1 contrasts two decisions with similar initial action scores. At A, no candidate improves on the executed action; at B, an alternative does, and we call such an alternative a repair. With enough budget to evaluate one decision, B is the better choice, but this distinction is hidden until the outcomes of its alternatives are observed. In search over two hops, the alternatives at A might offer only repeated evidence, while B includes a query that can recover the missing bridge fact, the intermediate fact that links the two hops. We therefore ask which cues, visible before intervention, reveal such opportunities.

*Code: `https://anonymous.4open.science/r/CIA-EE12/`

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.

An allocator chooses which decision points to evaluate; each has a set of candidate actions. Evaluating a decision means completing all its candidates and observing their outcomes. Previously evaluated decisions provide source data for training and validation; new decisions form the test set.

Repair value is the improvement that the best valid alternative gives over the executed action, divided by the effort of evaluating the candidates. Capture divides the repair value of the selected decisions by that of an oracle that knows all outcomes and has the same budget.

Across four synthetic retrieval settings and seven seeds, candidate features outperform retrieval confidence under the same value regressor. Predicting whether a useful repair exists, which we call recoverability, works better than predicting its value, while averaging the values of similar source decisions performs better still. Models fitted separately for each trajectory phase (evidence gathering, answer construction, verification) give smaller gains that vary across settings.

Cue Informed Allocation (CIA) estimates repair value from the nearest source decisions in feature space and predicts recoverability using all source examples or separate models for each phase. Validation on source data chooses which estimate or combination to use. The resulting rule ranks the decisions in Figure 1 before their alternative outcomes are known.

We compare scoring rules on the same features and source outcomes, then adapt policy learning methods that read text observations and keep their original credit mechanisms. These methods also differ from CIA in training feedback and optimization; we report these differences with the results.

Our contributions are:

- An analysis of which information predicts useful repairs, comparing retrieval features, candidate features, prediction targets, and similar source decisions.
- CIA, which uses this information to rank decisions for evaluation and chooses its scoring rule using only source data.
- Experiments separating the large benefit of similar source outcomes from the smaller gains of combining predictions and selecting rules, with comparisons to scoring baselines, adapted policy methods, and recorded interactive tasks.

2 Choosing which decisions to evaluate

Information before intervention. The two rows of Figure 1 correspond to the information available for allocation and the outcomes acquired afterward. At decision point x_i, the initial policy executes a_i^{exec} and a generator supplies alternatives $\mathcal{A}_i^{\text{alt}}$, giving the candidate set $\mathcal{A}_i = \{a_i^{\text{exec}}\} \cup \mathcal{A}_i^{\text{alt}}$. Before intervention, the allocator may use the observed prefix (the trajectory so far), candidate descriptions, and initial action scores s_{pre}. Completing the candidates reveals outcome scores s_{eval} and the best candidate $a_i^{\text{best}} = \arg\max_{a \in \mathcal{A}_i} s_{\text{eval}}(a)$. In synthetic search, the executed action is the candidate with the highest s_{pre}.

Define the positive score improvement Δ_i and the repair value v_i as

$$\Delta_i = [s_{\text{eval}}(a_i^{\text{best}}) - s_{\text{eval}}(a_i^{\text{exec}})]_+, \qquad v_i = g_i \frac{\Delta_i}{e_i}.$$

Here, $e_i > 0$ is the effort required to complete the candidate continuations, and $g_i \in \{0, 1\}$ indicates whether the candidates contain a valid repair. The simulator sets g_i when it constructs the alternatives, so positive score differences without a designated valid repair receive zero value. The allocator never observes g_i, which serves only to define outcomes. When every positive improvement is valid, $g_i = 1$ recovers the ordinary definition based on positive gain. Recoverability is the binary target $r_i = \mathbb{1}[v_i > \tau]$, which marks decisions whose repair value exceeds a small threshold ($\tau = 0.025$ in synthetic search).

Budget and Capture. For n decision points and budget fraction b, the allocator evaluates, or purchases, the candidate outcomes at $k = \lceil bn \rceil$ decision points. A selected set S_b has Capture

$$\text{Capture}(S_b) = \frac{\sum_{i \in S_b} v_i}{\sum_{j \in \text{TopK}(v,k)} v_j}.$$

The oracle uses the same candidates, and Capture is set to zero when the oracle's total value is zero. Capture measures acquired repair value rather than the return of a policy trained afterward. Because its denominator changes with b, it need not increase with the budget. Section 5.5 considers decisions with different evaluation costs.

Tasks and evaluation. We generate four synthetic retrieval settings with two search hops, inspired by HotpotQA, 2WikiMultiHopQA, MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). They differ in retrieval difficulty, ranking noise, and how highly distractor documents score. The names identify simulated retrieval conditions; the tasks are not drawn from the public question sets. Each setting is held out in turn for testing, while the other three supply 2,016 source decision points. The test set contains 1,512 points from separate tasks. Each task covers three trajectory phases and seven families of alternatives, such as query decomposition, repeated evidence, or missing context. Each decision has four candidate actions. The simulator evaluates every candidate to define the oracle, while the allocator sees only information available before intervention. Appendix A.1 gives the generation and sampling details.

The feature analysis and comparisons of scoring rules use seven seeds and four test settings, giving 28 paired evaluations on identical records. Confidence intervals over these evaluations resample the 28 pairs. Intervals across settings instead resample the four settings; they describe variation within this study, not uncertainty over new domains. The policy learning comparison uses one seed on the same task construction and is reported separately.

3 What information predicts repair opportunities?

Results use a 20% budget and average 28 evaluations unless stated otherwise. Four retrieval features describe the current ranking, eight candidate features describe the alternatives, and phase is recorded separately. Feature comparisons use the same ridge regressor and value target. The target and estimator comparisons use all 12 features, so their differences do not come from extra inputs.

3.1 Candidate features are more useful than retrieval confidence

Candidate features are more informative than retrieval confidence for the tested value regressor (Table 10). The eight candidate features achieve 0.4266 Capture, compared with 0.2669 for the four retrieval features; using both groups yields 0.4325. For this regressor, most of the useful information comes from the alternatives. As in Figure 1, candidate features distinguish alternatives that open new evidence paths from those that repeat a failure. Because features are compared in groups, we cannot attribute the effect to individual features.

Document overlap needs care because the simulator uses it both to describe candidates and to score outcomes. For Rule 1 of Section 4, which combines all three signals, overlap features alone achieve 0.5446 Capture, while excluding them gives 0.3565. Removing the part of the Rule 1 score that overlap predicts, with the outcome evaluator unchanged, gives 0.4426. Part of the overlap signal may thus reflect how outcomes are scored, and we have not tested outcomes that are independent of overlap.

3.2 Recoverability and value play different predictive roles

With all 12 features, a ridge model trained on r_i reaches 0.5078 Capture, compared with 0.4325 for scalar value regression. With this model class, separating useful repairs from low-value decisions works better than fitting the value itself. The value still matters, since Capture weights each repair by v_i and a binary target ignores it.

To test whether the recoverability model simply favors certain families or retrieval scores, we draw random selections that match its choices by family or by retrieval score range. They reach 0.2577 and 0.2322 Capture, respectively, compared with 0.5078 for the learned ranker. Shuffling labels within each family gives 0.2652. The learned ranking thus uses more than family or retrieval score.

3.3 SIMILAR SOURCE DECISIONS PROVIDE BETTER VALUE ESTIMATES

Nearest neighbor regression predicts repair value as a weighted average of the values of the 16 most similar source decisions, with similarity measured on all 12 features, and reaches 0.6235 Capture. This estimate keeps the value differences that binary labels discard and does not fit one linear function to all candidate sets. Only the estimator changes here, so the comparison says nothing about individual features.

4 CUE INFORMED ALLOCATION

CIA ranks decisions using three estimates computed from the same features: repair value from nearest neighbors, recoverability from all source examples, and recoverability within each trajectory phase. It selects one estimate or a combination by validation on source settings, then refits on all source data before scoring the test set (Figure 2).

Available features. The 12 features are four retrieval features (entropy, margin, score variation, and uncertainty before intervention) and eight candidate features: prefix diversity, document overlap, mean and variance of entity overlap, redundancy, fraction of unique documents, ambiguity pressure, and query edit distance. Phase is recorded separately. The inputs exclude test outcomes, repair availability, oracle choices, and final rewards.

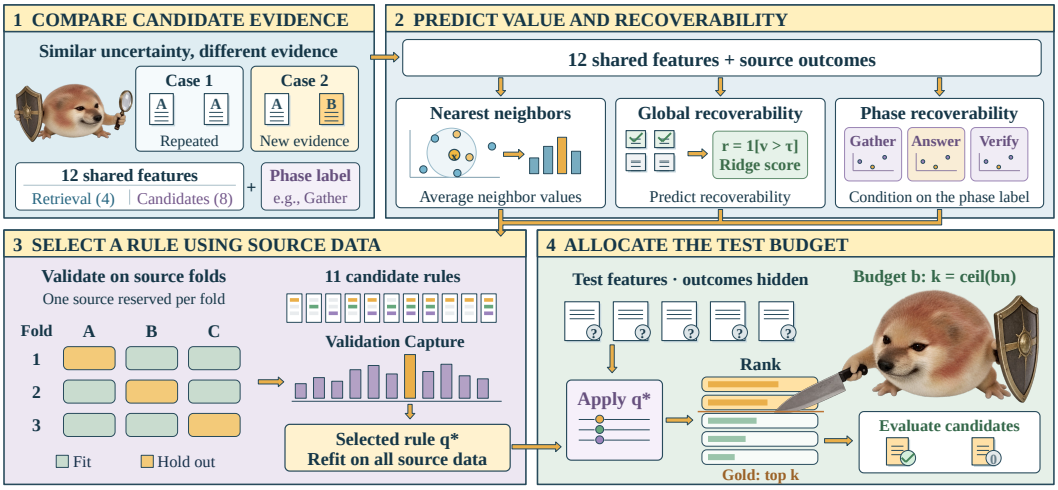

Figure 2: CIA fits three estimates from the same features and source outcomes, selects a scoring rule on source validation data, and evaluates the top k test decisions. Test outcomes are revealed only after selection. Illustrated candidates and scores are schematic.

Nearest neighbor regression. Let $\mathcal{N}_K(x)$ contain the K nearest source points after standardizing features using source statistics. Their weighted mean predicts repair value:

$$s_{\text{prox}}(x) = \frac{\sum_{j \in \mathcal{N}_K(x)} \alpha_j v_j}{\sum_{j \in \mathcal{N}_K(x)} \alpha_j}.$$

Exponentially weighted rules use $K \in \{8, 16, 32\}$ and $\alpha_j = \exp(-d_j/\sqrt{12})$, where d_j is the Euclidean distance to source point j.

Predicting recoverability. The global model s_{rec} fits weighted ridge regression to r_i, which indicates whether repair value exceeds τ. Its output is used for ranking and is not a calibrated probability. The phase model s_{phase} fits the same regression separately for evidence gathering, answer construction, and verification. It uses the global prediction when a phase has too few source examples. Both models use all 12 features.

Combining the estimates. We compare 11 scoring rules: individual estimates, weighted combinations, and variants of neighbor count and weighting. A combination has the form

$$s_w(x) = w_p z(s_{\text{prox}}(x)) + w_r z(s_{\text{rec}}(x)) + w_h z(s_{\text{phase}}(x)),$$

where z standardizes each prediction over the current validation or test batch without using outcome labels. This score normalization is separate from feature scaling, which uses only source statistics. Rule 1 combines all three signals, weighting the neighbor estimate most (Table 4). It is one of the 11 candidates, and CIA uses it only if source validation selects it.

Choosing and applying a rule. Each of the three source settings is held out once as a validation split while the estimators are fitted on the other two. Over these splits $\mathcal{C}_{\text{src}}$ and candidate rules $\mathcal{R}$, CIA chooses the rule with the highest mean validation Capture at a 20% budget:

$$q^\star = \arg\max_{q \in \mathcal{R}} \frac{1}{|\mathcal{C}_{\text{src}}|} \sum_{c \in \mathcal{C}_{\text{src}}} \text{Capture}_{0.20}(q; c).$$

After refitting on all source data, the selected rule scores the test set, and we evaluate the candidate outcomes at its top k decisions. The same rule serves all three test budgets, and the evaluator reveals test outcomes only after selection is complete. Validation uses Capture because the goal is to place useful repairs above the budget cutoff, and a model with low average prediction error can still order them poorly. Appendix B lists the candidate rules, weights, and fitting parameters.

Exact nearest neighbor scoring costs $O(nmD)$ for m source points, n test points, and D features; sorting costs $O(n \log n)$. The method requires evaluated source decisions that resemble the test decisions. It can only find repairs among the alternatives the candidate generator supplies.

5 Evaluating Budget Allocation

5.1 Comparing Scoring Rules on the Same Information

Table 1: Mean Capture over 28 evaluations; bold denotes the best listed value.

Scoring rule	10%	20%	30%
Random order	0.2319	0.2542	0.3059
Retrieval necessity	0.3008	0.3312	0.3750
Scalar value ridge	0.4449	0.4325	0.4880
Global rec.	0.4890	0.5078	0.5572
Phase rec.	0.5090	0.5084	0.5519
Nearest neighbors	0.6484	0.6235	0.6598
Nearest neighbors + rec.	0.6490	0.6255	0.6656
Nearest neighbors + phase	0.6513	0.6292	0.6673
All three signals	0.6482	0.6269	0.6672
CIA	**0.6568**	**0.6316**	**0.6694**

Figure 3: Cumulative CIA gains over the best heuristic, 28 pairs per budget.

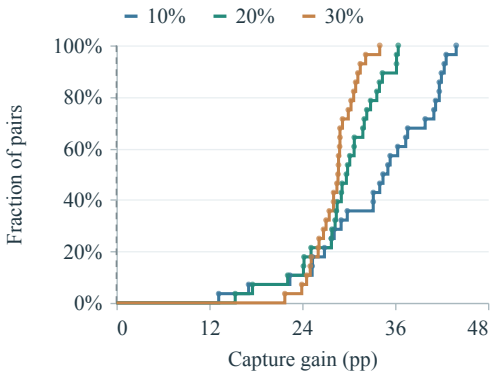

Table 1 compares rules on identical records, features, source outcomes, and budgets. Heuristics score visible properties directly. Regression models predict value or recoverability; nearest neighbors average over similar source decisions. The "All three signals" row is Rule 1; "rec." denotes recoverability. Other combinations use fixed weights, and CIA selects a rule by source validation.

At the primary 20% budget, CIA reaches 0.6316 Capture, compared with 0.2720 for the strongest generic acquisition rule (k-center coverage), 0.4325 for value regression, and 0.5078 for global recoverability. Nearest neighbors alone reach 0.6235, so most of the gain over these baselines comes from similar source outcomes. Rule selection adds 0.0081 over nearest neighbors, with positive paired gains in 22 of 28 evaluations and a 95% interval of [0.0012, 0.0150] across settings.

The six generic acquisition rules come from active learning and use density, uncertainty, coverage, or predicted model changes on the same features. We reproduce only their selection criteria, not the full active learning systems. Appendix C describes these and the additional scoring baselines.

Retrieval necessity, which combines retrieval entropy, uncertainty, and document diversity, is the strongest individual heuristic on average, reaching 0.3312 Capture. Taking the best of nine heuristics separately for each test set and budget gives 0.3393. We call this reference the best heuristic; choosing it requires test outcomes and is unavailable to an allocator. CIA exceeds it by 0.2923, with a 95% interval of [0.2727, 0.3101] over the 28 evaluations. All 28 paired gains are positive at each budget in Figure 3, with mean gains of 33.49, 29.23, and 28.25 percentage points. The improvement holds in all four settings and seven seeds.

5.2 Combining predictions and choosing a scoring rule

Adding phase to nearest neighbors raises Capture at 20% from 0.6235 to 0.6292, while adding global recoverability gives 0.6255. Rule 1 reaches 0.6269 (Table 1). More signals do not automatically improve the ranking; how they are combined matters as much as which ones are used. Source validation selects among these combinations and neighbor variants without using test outcomes.

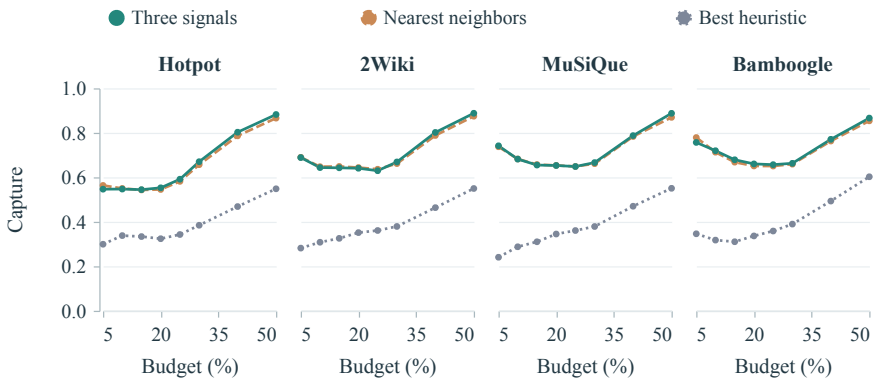

Figure 4: Rule 1 (all three signals), nearest neighbors, and the best heuristic across four retrieval settings and eight budgets. Each point averages seven seeds. The curves show fixed scoring rules; no validation step chooses among them.

Nearest neighbors and Rule 1 exceed the best heuristic at all 32 combinations of setting and budget in Figure 4. Combining signals gives smaller gains that vary with budget. Some Capture curves fall as the budget grows because the oracle value in the denominator also grows.

Replacing the global recoverability model in Rule 1 with a shallow random forest raises Capture to 0.6406. Nearest neighbors and phase stay in the score, so the number reflects a change of one component and says little about a forest used alone.

CIA improves on nearest neighbors with 16 neighbors in ten of the twelve combinations of setting and budget. The exceptions are MuSiQue at 10% and 20%, where it trails by 0.0011 and 0.0010. For the 30% test budget, validating at 30% gives 0.6763 instead of 0.6694. With only two source settings for training, the gain over nearest neighbors remains positive at 30%, but its 20% interval includes zero. These close comparisons depend on the available source data and validation budget.

5.3 Adapting policy learning methods to budget allocation

In a separate experiment with one seed, we adapt eight policy learning methods to allocation, using raw text observations. ProxMO (Fang et al., 2026) transfers credit between similar texts and scales it by success rate; GiGPO (Feng et al., 2025) groups episodes and matching states; GraphGPO (Cheng et al., 2026) computes credit on rollout transition graphs. ARPO (Dong et al., 2025) chooses where to branch rollouts by entropy, and CARL (Shen et al., 2025) concentrates updates on critical actions. HCAPO (Tan et al., 2026) uses hindsight verification and credit at multiple scales. For C3 (Chen et al., 2026), we use the value assisted counterfactual baseline from the authors' code. StepPO (Wang et al., 2026) trains an actor and critic at each step.

ProxMO, GiGPO, GraphGPO, ARPO, C3, and StepPO use the authors' implementations of their learning mechanisms; we implement CARL and HCAPO from their papers. All start from Qwen2.5-

Table 2: Policy learning adaptations, seed 47001. Raw observations and test budgets match; training feedback and test scoring differ (Section 5.3). †: our implementation from the paper; ‡: C3 variant with a learned value model. Uniform random shows expected Capture.

	Setting Capture at 20%				Mean over settings		
Method	Hotpot	2Wiki	MuSiQue	Bamboogle	10%	20%	30%
ProxMO (Fang et al., 2026)	0.2672	0.2862	0.2573	0.2311	0.2345	0.2604	0.3158
GiGPO (Feng et al., 2025)	0.2478	0.2836	0.2005	0.2214	0.2139	0.2383	0.2963
GraphGPO (Cheng et al., 2026)	0.2547	0.2939	0.2165	0.2143	0.2142	0.2449	0.2942
ARPO (Dong et al., 2025)	0.2684	0.2920	0.1773	0.2037	0.2041	0.2354	0.2726
CARL† (Shen et al., 2025)	0.2698	0.2986	0.2445	0.2169	0.2372	0.2574	0.3175
HCAPO† (Tan et al., 2026)	0.2411	0.2879	0.2207	0.2036	0.2271	0.2383	0.2986
C3‡ (Chen et al., 2026)	0.2840	0.3492	0.2339	0.2743	0.2490	0.2853	0.3347
StepPO (Wang et al., 2026)	0.2471	0.2964	0.2167	0.2099	0.2223	0.2425	0.3005
Uniform random	0.2295	0.2685	0.2714	0.2785	0.2388	0.2620	0.3032
CIA	**0.6331**	**0.6579**	**0.6759**	**0.6649**	**0.6879**	**0.6580**	**0.6635**

7B-Instruct, and training duration is selected on source data. Methods share source records and test budgets but differ in feedback and training computation. Appendix D gives the implementation, training cost, and evaluation details.

Information and training. All methods receive the same visible documents, candidates, retrieval scores, operation name, and known costs. CIA computes its 12 features from these inputs; policy methods read them as text and do not receive our features or estimates. Source and test tasks are disjoint, and no method sees test outcomes or the simulator's hidden document roles.

During training, each episode presents 20 decisions in sequence, and the policy must buy exactly four. An action is forced whenever the remaining quota permits only one choice. The terminal reward is binary, and its expectation equals episode Capture. In contrast, CIA fits the value and recoverability label of each source decision. At test time, each policy scores every decision independently by $\log P(\text{buy}) - \log P(\text{skip})$, as if it were the first of 20 decisions with a quota of four and no history. We select its top k scores before revealing any test outcomes.

Results and interpretation. Table 2 covers all eight adaptations and four settings. At 20%, CIA reaches 0.6580 on seed 47001; C3's value assisted variant is the best adaptation at 0.2853, versus 0.2620 expected Capture for uniform random selection. We keep these single-seed results apart from the seven-seed study.

About 15 of the 20 actions per episode are forced. In 77 to 95% of groups of eight rollouts, all rollouts receive the same binary reward, which gives little feedback about which choices were better. At test time, the policy also loses the history and changing quota used during training. These differences, along with representation and optimization, contribute to the gap. The comparison concerns the allocation task only and does not tell us how the methods rank on their original benchmarks.

5.4 Which source examples help?

Number of neighbors. We vary the number of neighbors in Rule 1 while keeping the features, source records, other estimates, and weights fixed; rule selection is not repeated. At 20%, Capture falls from 0.6269 with 16 neighbors to 0.6012, 0.5526, 0.5289, and 0.5137 with 32, 64, 96, and 128 neighbors. At 30%, the corresponding values are 0.6672, 0.6538, 0.6275, 0.6051, and 0.5848. Averaging over more distant examples weakens the ranking here, but another domain may favor a different number of neighbors.

Types of source examples. We remove the source decisions of each family in turn, refit the estimators, and keep the chosen rule and test set. Table 3 reports the repair rate (fraction with $v_i > \tau$) of each test family, its mean value over all decisions, the acquired fraction (the share of its total value that is selected), and the change in overall Capture after removal.

Table 3: Source families at a 20% budget over 28 evaluations. Acquired: fraction of a family's total value selected. Removal Δ: change in overall Capture without that family's source data.

Family	Repair rate	Mean value	Acquired	Removal Δ
Query decomposition	34.2%	0.204	38.8%	−0.0935
Entity ambiguity	25.7%	0.168	67.4%	−0.0581
Repeated evidence	27.5%	0.163	22.2%	−0.1075
Citation errors	25.8%	0.155	22.8%	−0.0458
Hop order	26.5%	0.135	73.0%	−0.0161
Distractors	20.8%	0.159	69.6%	−0.0525
Missing context	33.5%	0.206	61.5%	−0.0982

Repair rates and acquired fractions vary across families. Query decomposition has 34.2% recoverable points and 38.8% of its value selected, compared with 26.5% and 73.0% for hop order. Removing any source family lowers overall Capture. Repeated evidence and missing context cause the largest losses (0.1075 and 0.0982); hop order causes the smallest (0.0161). Yet only 22.2% of the value from repeated evidence is selected, versus 73.0% for hop order. A family can thus provide useful training examples even when its own test repairs are hard to identify. Removing a family also shrinks the source set, so these losses mix family identity with sample size.

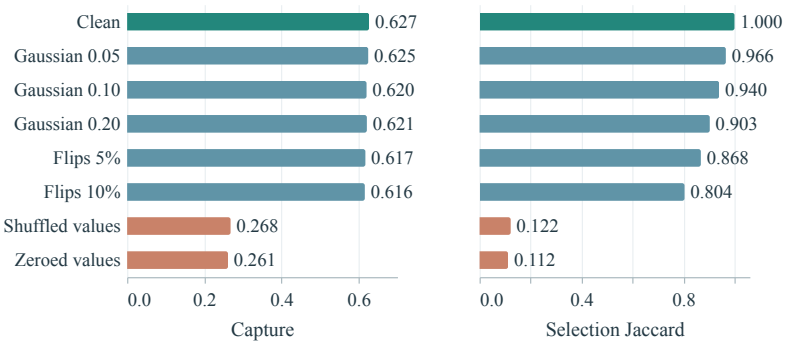

Figure 5: Rule 1 (all three signals) with noisy source labels at a 20% budget. Left: Capture. Right: overlap with the decisions selected using the original labels. Test outcomes are unchanged.

Errors in source labels. Figure 5 runs Rule 1 on eight versions of the source labels, using the same records throughout; rule selection is not repeated. Selection overlap is the Jaccard index (intersection over union of the two selected sets). Gaussian noise with $\sigma = 0.20$ gives overlap 0.903 and changes Capture by less than 0.006. Flipping 10% of labels changes more of the selected set (overlap 0.804) but retains 0.6157 Capture. Shuffling values reduces Capture to 0.2679 and overlap to 0.122; setting all source values to zero gives 0.2613 and 0.112. Moderate random errors are less damaging than removing the relation between features and outcomes.

5.5 Evaluation costs and available alternatives

Unequal evaluation costs. When decisions differ in evaluation cost d_i, the budget caps their total cost. The cost d_i determines which decisions fit under this cap, while continuation effort e_i defines their repair value. Dividing the Rule 1 score by cost raises Capture from 0.6276 to 0.6406 when runtime depends on family, and from 0.5820 to 0.6414 when runtime varies widely across alternatives. All comparisons use the same cap. Cost adjustment gives no improvement when costs come from work shared across alternatives or from estimated human review time. The oracle in the denominator of Capture picks decisions greedily by value per unit cost and so only approximates the knapsack optimum.

Incomplete candidate sets. With only two or three recorded candidate actions evaluated, the recoverability ranker reaches 0.4396 and 0.4765 Capture, compared with 0.3000 and 0.2706 for the

best heuristic. Features and effort keep their original values, so the test removes candidate outcomes without rerunning a smaller search. The ranker still finds useful purchases among the retained alternatives, though it cannot recover repairs missing from that pool.

Recorded interactive tasks. We also rank decisions using recorded outcomes from search, WebShop (Yao et al., 2022), and ALFWorld (Shridhar et al., 2021). Each domain uses its own features and a source validation split. Against the strongest baseline for each test set and budget, gains are positive on three of six sets at 20%, with one tie, and on five sets at 30%. The smallest test set contains only 12 decision points. These results are exploratory because the signs are mixed, the samples are small, and the records do not confirm that every feature was available before intervention. Appendix F.4 reports the splits and individual results.

6 Related work

Choosing interventions and assigning credit. Credit assignment attributes outcomes to decisions (Pignatelli et al., 2023; Mesnard et al., 2021). ProxMO, GiGPO, and GraphGPO share credit across similar texts, groups, and transitions (Fang et al., 2026; Feng et al., 2025; Cheng et al., 2026). HCAPO uses hindsight, C3 estimates counterfactual baselines, and StepPO assigns credit at each step (Tan et al., 2026; Chen et al., 2026; Wang et al., 2026). These methods learn policies from completed rollouts, whereas we choose which alternative outcomes to evaluate. Section 5.3 adapts them to this task.

Allocating evaluation and learning effort. ARPO branches rollouts using entropy, while CARL concentrates updates on critical actions (Dong et al., 2025; Shen et al., 2025). Active learning acquires labels using uncertainty, information, density, coverage, or predicted model changes (Ren et al., 2022; Ash et al., 2021; Hacohen et al., 2022; Yehuda et al., 2022; Parvaneh et al., 2022); we compare these criteria on the same features. Budgeted acquisition (Zhou et al., 2021) and causal bandits (Nair et al., 2021) account for selection costs. Wen et al. (2025) choose labels that improve both factual and counterfactual coverage for treatment effect estimation. Dukkipati et al. (2025) collect additional trajectories to improve offline policies. We select decisions whose alternatives can reveal useful repairs; prioritized replay instead reorders recorded transitions (Pan et al., 2022; Sujit et al., 2023).

Interactive agents and retrieval. Retrieval systems and interactive agents weigh the cost of additional evidence (Trivedi et al., 2023; Jeong et al., 2024; Asai et al., 2024; Jiang et al., 2023; Yao et al., 2023; Shinn et al., 2023; Schick et al., 2023; Qin et al., 2024). Agent benchmarks evaluate complete behavior (Liu et al., 2024; Ma et al., 2024; Koh et al., 2024; Trivedi et al., 2024; Yoran et al., 2024; Xie et al., 2024). Our synthetic tasks supply all candidate outcomes, so every allocator is compared with the same budgeted oracle before any policy learning.

7 Limitations and conclusion

Candidate features help identify useful repairs, and outcomes of similar source decisions help estimate their value. CIA uses source validation to choose among nearest neighbor estimates, recoverability predictions, and their combinations. Most of its improvement over the scoring baselines comes from nearest neighbors; combining predictions and selecting a rule give smaller gains that depend on the setting. A practical starting point is therefore nearest neighbor regression over a pool of evaluated source decisions, with the other estimates added when source validation shows a gain.

The feature findings apply to the estimators we tested. Correlated features and the simulator's use of overlap in scoring outcomes prevent us from assigning a causal contribution to each feature. Results also depend on relevant source examples and on the generator, since CIA can only find repairs among the proposed alternatives. The policy comparisons use one seed, and the adapted methods differ from CIA in training feedback and test scoring. In the recorded-outcome studies, we have not verified that every feature would be available before intervention in a live task. Whether source decisions from one domain help rank decisions in another domain is also untested. Larger interactive experiments are needed to test whether selecting these outcomes improves subsequent learning.

AI USE STATEMENT

AI tools assisted with drafting and polishing the manuscript and with retrieving and discovering related literature. The authors reviewed and verified all resulting text and references and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The supplementary archive contains intervention records, configurations, analysis code, and expected outputs. Appendices A.2, B, and F.1 specify the simulator, CIA, and the evaluation of purchase costs. Appendix D details the matched baseline implementations and evaluation; Table 14 reports their results for each profile.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024. doi: 10.48550/arxiv.2310.11511.

Jordan T. Ash, Surbhi Goel, A. Krishnamurthy, and S. Kakade. Gone fishing: Neural active learning with fisher embeddings. In *Neural Information Processing Systems*, volume 34, pp. 8927–8939, 2021. doi: 10.48550/arxiv.2106.09675.

Yanjun Chen, Yirong Sun, Hanlin Wang, Jinghan Wang, Xinming Zhang, Xiaoyu Shen, Wenjie Li, and Wei Zhang. Exact is easier: Credit assignment for cooperative LLM agents. arXiv preprint arXiv:2603.06859, 2026.

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

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization. arXiv preprint arXiv:2507.19849, 2025.

Ambedkar Dukkipati, Ranga Shaarad Ayyagari, Bodhisattwa Dasgupta, Parag Dutta, and Prabhas Reddy Onteru. Active reinforcement learning strategies for offline policy improvement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16418–16425, 2025. doi: 10.1609/aaai.v39i16.33803. URL https://ojs.aaai.org/index.php/AAAI/article/view/33803.

Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, Haolin Shi, Chang Liu, and Peilin Zhao. Proximity-based multi-turn optimization: Practical credit assignment for LLM agent training. arXiv preprint arXiv:2602.19225, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. arXiv preprint arXiv:2505.10978, 2025.

Guy Hacohen, Avihu Dekel, and D. Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8175–8195. PMLR, 2022. doi: 10.48550/arxiv.2202.02794.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7036–7050. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.389.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7969–7992. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.495.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.50.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024. doi: 10.48550/arxiv.2308.03688.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Neural Information Processing Systems*, 2024. doi: 10.48550/arxiv.2401.13178.

Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, Éric Moulines, Marcus Hutter, Lars Buesing, and Rémi Munos. Counterfactual credit assignment in model-free reinforcement learning. In *International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7654–7664. PMLR, 2021. doi: 10.48550/arxiv.2011.09464.

V. Nair, Vishakha Patil, and Gaurav Sinha. Budgeted and non-budgeted causal bandits. In *International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 2017–2025. PMLR, 2021. doi: 10.48550/arxiv.2012.07058.

Yangchen Pan, Jincheng Mei, Amir-massoud Farahmand, Martha White, Hengshuai Yao, Mohsen Rohani, and Jun Luo. Understanding and mitigating the limitations of prioritized experience replay. In *Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pp. 1561–1571. PMLR, 2022. doi: 10.48550/arxiv.2007.09569.

Amin Parvaneh, Ehsan Abbasnejad, Damien Teney, Reza Haffari, Anton van den Hengel, and Javen Qinfeng Shi. Active learning by feature mixing. In *Computer Vision and Pattern Recognition*, pp. 12227–12236, 2022. doi: 10.1109/cvpr52688.2022.01192.

Eduardo Pignatelli, Johan Ferret, Matthieu Geist, Thomas Mesnard, Hado van Hasselt, Olivier Pietquin, and Laura Toni. A survey of temporal credit assignment in deep reinforcement learning. *arXiv [cs.LG]*, 2 December 2023. doi: 10.48550/arxiv.2312.01072. URL `https://openreview.net/forum?id=bNtr6SLgZf`.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.378.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024. doi: 10.48550/arxiv.2307.16789.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B. Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Computing Surveys*, 54(9):1–40, 2022. doi: 10.1145/3472291.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, R. Raileanu, M. Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Neural Information Processing Systems*, 2023. doi: 10.48550/arxiv.2302.04761.

Leyang Shen, Yang Zhang, Chun Kai Ling, Xiaoyan Zhao, and Tat-Seng Chua. CARL: Criticality-aware agentic reinforcement learning. arXiv preprint arXiv:2512.04949, 2025.

Noah Shinn, Federico Cassano, Beck Labash, A. Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0377.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. doi: 10.48550/arxiv.2010.03768.

Shivakanth Sujit, Somjit Nath, Pedro H. M. Braga, and Samira Ebrahimi Kahou. Prioritizing samples in reinforcement learning with reducible loss. In *Neural Information Processing Systems*, 2023. doi: 10.48550/arxiv.2208.10483.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. arXiv preprint arXiv:2603.08754, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.557.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16022–16076. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.850.

Daoyu Wang, Qingchuan Li, Mingyue Cheng, Jie Ouyang, Shuo Yu, Qi Liu, and Enhong Chen. StepPO: Step-aligned policy optimization for agentic reinforcement learning. arXiv preprint arXiv:2604.18401v1, 2026.

Hechuan Wen, Tong Chen, Mingming Gong, Li Kheng Chai, Shazia Sadiq, and Hongzhi Yin. Enhancing treatment effect estimation via active learning: A counterfactual covering perspective. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 66437–66466. PMLR, 2025. URL https://proceedings.mlr.press/v267/wen25b.html.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, T. Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Neural Information Processing Systems*, 2024. doi: 10.48550/arxiv.2404.07972.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Neural Information Processing Systems*, 2022. doi: 10.48550/arxiv.2207.01206.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. doi: 10.48550/arxiv.2210.03629.

Ofer Yehuda, Avihu Dekel, Guy Hacohen, and D. Weinshall. Active learning through a covering lens. In *Neural Information Processing Systems*, 2022. doi: 10.48550/arxiv.2205.11320.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. AssistantBench: Can web agents solve realistic and time-consuming tasks? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8938–8968. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.505.

Yilun Zhou, Adithya Renduchintala, Xian Li, Sida Wang, Yashar Mehdad, and Asish Ghoshal. Towards understanding the behaviors of optimal deep active learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1486–1494. PMLR, 2021. URL https://proceedings.mlr.press/v130/zhou21b.html.

APPENDIX

A DATA GENERATION AND EVALUATION

A.1 DATA SPLITS AND STATISTICAL ANALYSIS

Corpora and alternatives. Each evaluation contains 2,016 source decision points and 1,512 target (test) decision points. A default decision point has four candidate actions. The initial policy selects one using only the evidence available before intervention, and this executed action remains in the candidate set when the evaluator scores all four continuations. Repair value is positive when a valid repair is available and another candidate scores better than the executed action. The four synthetic settings, also called corpora, use retrieval conditions patterned after HotpotQA, 2WikiMultiHopQA (2Wiki), MuSiQue, and Bamboogle; the tasks are generated by the simulator and are not copied from those datasets.

The simulator draws 96 source and 72 test tasks that each require two search hops. Each task contributes one decision point for each combination of three trajectory phases and seven families of alternatives (also called branch families, since each alternative branches off the trajectory), producing $96 \times 3 \times 7 = 2{,}016$ source and $72 \times 3 \times 7 = 1{,}512$ test decision points. The families modify query decomposition, entity ambiguity, repeated evidence, citation errors, hop order, distractors, or missing context. They preserve the trajectory prefix and change the evidence set or its interpretation; the action formed by the top two retrieved documents is included whenever it is the policy choice or a natural alternative.

For each test setting, feature statistics, model parameters, and neighbor pools are fitted on the remaining corpora. Source validation chooses the scoring rule. That rule ranks the n target decision points, and budget fraction b selects the top $k = \lceil bn \rceil$ decision points, where $\lceil \cdot \rceil$ rounds up to the next integer.

Models and uncertainty estimates. The recoverability model is ridge regression on the binary label $\mathbb{1}[v_i > 0.025]$. Features are standardized using source data. Source examples whose target is at or above its 80th percentile receive three times the loss weight; with the binary target, these are the positive examples. The regularization penalty is 0.01, and the regression output is used directly for ranking. The phase signal fits this model separately within evidence gathering, answer construction, and verification, falling back to the global source model if a phase has fewer than 14 source examples. The nearest neighbor rule uses Euclidean distance in the same standardized space of 12 features, averages 16 source values, and weights neighbor j by $\exp(-\|z - z_j\|/\sqrt{12})$. The scoring rules below also vary neighborhood size and weighting. Confidence intervals resample the 28 (corpus, seed) pairs and preserve pairing between selectors with the same candidate pool, values, and oracle denominator.

Statistical comparisons. Uncertainty estimates account for variation across evaluations defined by a corpus and seed. Every difference between CIA and the best heuristic is positive. A two-sided sign test gives $p = 7.45 \times 10^{-9}$ at each budget. Against nearest neighbors, CIA has positive gains in 18, 22, and 24 of 28 evaluations at 10%, 20%, and 30%; intervals across settings resample the four corpora. The standardized effect against the best heuristic, defined as mean gain divided by the standard deviation of gains, is 4.23, 5.80, and 10.62.

A.2 SIMULATOR DETAILS

Each seed draws disjoint pools of 96 source and 72 test tasks. Every task contains two supporting documents, matched distractors, a contradiction, and a document with low overlap. Three trajectory phases and seven branch families cover query decomposition, entity ambiguity, repeated evidence, citation errors, hop order, distractors, and missing context. At each decision point, the retrieval policy executes the action with the highest score among four choices; the remaining three become alternatives that modify one query, evidence choice, or interpretation while preserving the trajectory prefix.

The policy score available before intervention is 0.58 mean retrieval quality $+0.32$ mean entity overlap $+0.10$ document diversity, with a 0.05 redundancy penalty. The score after completing a continuation combines answer quality, support coverage, and citation support with weights $0.45/0.35/0.20$, then accounts for retrieval quality, entity overlap, distractors, and contradictions. Tasks, retrieval rankings, and candidate sets vary across seeds. Once a candidate is generated, its outcome score is a deterministic function of its evidence, so the candidate with the highest score is the best one generated. The candidate generator also records whether it included the designated valid repair. Values are zero when this condition is false, even if two evaluator scores differ; otherwise value is the positive gain over the executed action divided by effort. Evaluating all four actions and applying this gate gives the oracle value vector. The allocator uses the 12 features available before intervention and the recorded phase, while the policy adaptations use the underlying public observations.

B Allocation method and fitting details

B.1 Prediction models

CIA ranks decision points using information available before intervention. It constructs three parallel source signals, selects a rule by holding out each source setting in turn, and refits on all sources before ranking targets. Table 5 summarizes the signals.

Nearest neighbor regression averages the values of similar source decisions, while recoverability predicts whether repair value exceeds the stated threshold. Phase distinguishes evidence gathering, answer construction, and verification. Rule 1 combines all three signals ("All three signals" in the tables) and gives the neighbor estimate the largest weight; CIA uses it only when source validation selects it. Scalar value regression is a comparison control, not one of the 11 candidate rules selected by CIA.

B.2 Defining repair value

A decision point falls into one of three cases. The initial policy may already choose the best candidate; it may choose poorly when every alternative is also poor; or it may choose poorly while another alternative preserves better evidence. The candidate set always contains the action executed in the original trajectory, a_i^{exec}, together with the proposed alternatives. Only the third case can return corrective evidence from the intervention being purchased. We write the positive score improvement over the executed action as Δ_i and divide it by the work e_i required to complete the candidate continuations. The repair value is $v_i = g_i \Delta_i / e_i$, where g_i records whether the simulator provides a valid repair. A positive evaluator gap does not by itself make a valid repair.

The repair gate g_i excludes invalid improvements, and the positive part of Δ_i excludes decisions already as good as the alternatives. Dividing by e_i measures the improvement obtained per unit of continuation work. Summing v_i over the purchased set rewards rankings that spend their limited slots on valuable repairs. Two selectors can therefore have similar classification accuracy but different Capture.

In one released source record, the four candidates use support from both hops, support for the bridge only, support for the answer only, and the retriever's top two documents. Their initial scores are $(0.750, 0.801, 0.730, 0.872)$, so the trajectory executes the fourth action. Completing the candidates gives evaluator scores $(0.974, 0.295, 0.417, 0.341)$, making the first action best. Repair availability is one; the unrounded scores and effort give $v_i = 0.503$ ($e_i \approx 1.26$). The value measures improvement over the executed fourth action.

B.3 Features and scoring rules

The 12 input features are retrieval entropy, retrieval margin, variation in retrieval scores, uncertainty before intervention, prefix diversity, document overlap, mean and variance of entity overlap, alternative redundancy, the fraction of unique documents, ambiguity pressure, and query edit distance. Phase enters through a separate model for each phase. Outcome scores, final rewards, oracle choices, branch family, and task difficulty are excluded. The features describe the initial action's confidence and how the alternatives differ in their evidence.

Table 4: The 11 rules compared by source validation. The weights w_p, w_r, and w_h apply to the nearest neighbor value estimate, global recoverability, and phase recoverability (Section 4).

Rule	Estimates used	K	Weighting	w_p	w_r	w_h
1	All three signals	16	E	0.80	0.10	0.10
2	Nearest neighbors	16	E	1	0	0
3	Neighbors and phase	16	E	0.85	0	0.15
4	Neighbors and recoverability	16	E	0.85	0.15	0
5	Equal weights	16	E	$1/3$	$1/3$	$1/3$
6	Recoverability	n/a	n/a	0	1	0
7	Phase	n/a	n/a	0	0	1
8	Recoverability and phase	n/a	n/a	0	0.50	0.50
9	All three signals	16	R	0.80	0.10	0.10
10	Nearest neighbors	8	E	1	0	0
11	Nearest neighbors	32	E	1	0	0

Weighting E uses $\exp(-d/\sqrt{12})$ and R uses $1/(d+10^{-6})$, where d is Euclidean distance after source feature scaling. Entries marked n/a do not use neighbors. If rules tie in mean validation Capture, the lower-numbered rule is chosen.

Table 5: Prediction models and their roles. CIA selects among nearest neighbors, recoverability, phase, and their combinations; value regression is a comparison control.

Score	Role
All three signals	Rule 1: nearest neighbors, recoverability, and phase, weighted toward nearest neighbors.
Nearest neighbors	Averages the values of nearby source decision points.
Recoverability	Estimates whether an available alternative can improve the current decision.
Phase	Conditions the score on the stage of the trajectory.
Value regression	Predicts the scalar value gap from the same features available before intervention.

Table 4 lists all 11 scoring rules. A rule with one estimate ranks decisions by that estimate; a mixed rule standardizes each estimate before taking the weighted sum. Source validation selects the rule with the highest mean Capture at a 20% budget. Across the 28 evaluations, it selects a rule that uses only nearest neighbors seven times and a combination 21 times.

For n test decisions, m source decisions, and D features, exact neighbor scoring requires $O(nmD)$ operations. Approximate search reduces this term without changing the selection objective.

Source training rows provide feature statistics. Mixed rules standardize each predicted component on the current source validation or target set as $z(s_i) = (s_i - \overline{s})/(\mathrm{sd}(\mathbf{s}) + 10^{-9})$, using the population standard deviation and no outcome labels.

B.4 Scores used in each analysis

Some analyses test the complete allocator, and others test only one of its component scores. Table 6 lists the score used in each analysis. In the diagnostic tables, the CIA rows shown for reference still use the full rule selection.

C Baselines for selecting decisions

All direct comparisons rank the same recorded decision points for outcome acquisition. The following tables identify which scores use the visible features, source labels, or additional simulator annotations. The custom graph, group, and nearest neighbor scores are acquisition controls; the policy training adaptations are described separately.

Table 6: Scores evaluated in each analysis. Fitting a predictor and choosing a scoring rule are separate steps.

Analysis	Score and rule selection
Main CIA comparisons with direct scores and acquisition rules	CIA selects among the eleven rules, then fits on all source data. Every other row uses its stated score.
Policy learning comparison	Full CIA on the same decision pool as each policy; one seed.
Validation budget and CIA selection stability	Repeat rule selection under the stated source split and validation budget.
Source family removal	Keep the rule chosen with all source families, and refit its predictors after removing a family.
Feature groups, prediction targets, recoverability controls, class balance, and candidate truncation	Individual value or recoverability predictors; no rule selection.
Budget curves	Figure 4 uses Rule 1 (all three signals) and nearest neighbors. Table 18 uses recoverability. Neither selects rules.
Overlap, normalization, neighbor search, and source subset controls	Rule 1 (all three signals) with the stated feature, kernel, normalization, or search change; no rule selection.
Mixture sensitivity, model capacity, distance weighting, source label errors, purchase prices, and episode support	The stated mixture or component score; no rule selection.
Recorded Search, WebShop, and ALFWorld tasks	Select among six rules on a source holdout within each domain, then fit on that domain's source data.

C.1 Heuristics and the best heuristic

Metrics and heuristic comparisons. Capture is the selected value divided by the value of the oracle's k most valuable decision points at the same budget. Because this denominator changes with the budget, curves across budgets need not be monotone; comparisons are made within each budget. Episode support aggregates selected repairs along a trajectory (Appendix F.3.4). Cosine similarity measures directional agreement with the full vector of repair values, and recall at the top 10% is the fraction of the most valuable 10% of decision points that the selection recovers.

The nine heuristics rank candidates by retrieval confidence, retrieval necessity, information gain, uncertainty, ambiguity and overlap, projected retrieval or structure features, the spread in initial scores, or random order. Retrieval confidence combines the retrieval margin and entropy. Retrieval necessity combines entropy, uncertainty, and document diversity, while information gain combines diversity among alternatives, variation in retrieval scores, and overlap. Projected retrieval combines the retrieval margin, entity overlap, and entropy; projected structure combines diversity, overlap, ambiguity, and uncertainty. The remaining heuristic names state the quantity they use directly.

Throughout these comparisons, "best heuristic" denotes the hindsight maximum over the nine heuristics for each target and budget, rather than one deployable rule. Table 7 reports every heuristic separately. At 20%, retrieval necessity gives the best result in 20 of 28 evaluations and the spread in initial scores in eight; at 30%, retrieval necessity is best in 16, the spread in 11, and uncertainty in one. In tables, bold marks the best result and underlining marks the second best within each comparison block.

C.2 Regression and structured scoring baselines

The nine controls in Table 8 compare scoring assumptions on the recorded decision points. Value regression predicts scalar repair value from the 12 standardized features. Nearest neighbor regression averages 16 source values with exponential distance weights. The phase ranker predicts binary recoverability separately within each trajectory phase. These are acquisition scores, not official reproductions of agent training methods.

Group value pools source values by phase, simulator branch family, and retrieval margin bucket. Centered group value groups by phase and family, then subtracts the phase mean and adds the global mean. These two controls use a simulator family annotation in addition to the features available to

Table 7: Mean Capture for each heuristic over 28 evaluations. Best at 20% counts evaluations in which the row has the highest Capture among the nine heuristics.

Heuristic	10%	20%	30%	Best at 20%
Retrieval necessity	0.3008	0.3312	0.3750	20/28
Spread in initial scores	0.2595	0.2906	0.3568	8/28
Uncertainty	0.2459	0.2671	0.3185	0/28
Random order	0.2319	0.2542	0.3059	0/28
Information gain	0.1469	0.2539	0.2735	0/28
Projected retrieval	0.2228	0.2366	0.2823	0/28
Retrieval confidence	0.1999	0.2311	0.2803	0/28
Projected structure	0.2293	0.2203	0.2218	0/28
Ambiguity and overlap	0.0524	0.0401	0.1619	0/28

Table 8: Capture against nine internal scoring controls. Capture columns are means over 28 evaluations from four corpora and seven random seeds. The final column lists CIA's gain at 10%, 20%, and 30%, in that order; Self denotes comparison with itself.

Method	10%	20%	30%	CIA gain 10/20/30%
CIA	**0.6568**	**0.6316**	**0.6694**	Self
Nearest neighbors	0.6484	0.6235	0.6598	0.0084/0.0081/0.0096
Phase ranker	0.5090	0.5084	0.5519	0.1479/0.1232/0.1176
Value regression	0.4449	0.4325	0.4880	0.2119/0.1991/0.1814
Graph value	0.2740	0.3026	0.3701	0.3828/0.3290/0.2993
Centered group value	0.2696	0.2905	0.3329	0.3872/0.3411/0.3365
Action sensitivity	0.2107	0.2767	0.3522	0.4461/0.3549/0.3172
Tool uncertainty	0.2587	0.2684	0.3154	0.3982/0.3632/0.3540
Group value	0.2506	0.2680	0.3199	0.4062/0.3636/0.3496
Criticality	0.0528	0.0462	0.1636	0.6040/0.5854/0.5059

the proposed allocator. Graph value propagates source values over pseudo state nodes formed from phase and binned uncertainty, ambiguity, entropy, and retrieval margin; it uses task membership to link source nodes across phases.

Criticality is a heuristic using uncertainty and entropy. Tool uncertainty combines uncertainty, score gap, and a phase weight. Action sensitivity uses the spread and standard deviation of initial candidate scores together with overlap and query edit sensitivity. All controls purchase outcomes only after ranking. Their shared evaluation uses the same source/test records, candidate pool, budget, and Capture denominator.

C.3 Generic acquisition rules

Table 9 compares CIA with six generic acquisition rules at the primary budget. The k-center rule repeatedly selects the candidate farthest from those already chosen. Density averages similarities that decrease with squared distance to 32 source neighbors. Ensemble disagreement is the standard deviation of eight rankers trained on resampled source data. The uncertainty and diversity rule combines standardized uncertainty and k-center scores with weights 0.55 and 0.45. Gradient norm multiplies uncertainty by feature magnitude; expected model change uses the same two factors to estimate the size of an update.

C.4 Feature groups, prediction targets, and individual scoring rules

Section 3 uses the feature group and target comparisons in Table 10. Table 11 compares the method chosen on source data with representative individual candidate rules on all three budgets. The default nearest neighbor rule uses 16 source points; the smaller variant uses eight.

Table 9: CIA and six generic acquisition rules at a 20% budget. Higher Capture is better; k-center is the strongest generic rule.

Method	Capture
CIA	**0.6316**
Best heuristic	0.3393
k-center	0.2720
Density	0.2613
Ensemble disagreement	0.2463
Uncertainty + diversity	0.2443
Gradient norm	0.2028
Expected model change	0.1994

Table 10: Prediction targets and feature groups under ridge ranking. The feature group rows use scalar value regression; all prediction target rows use all 12 features. Results use a 20% budget; gain is measured from the best heuristic, and the final column counts evaluations with gain above zero.

Signal or target	Capture	Gain	Positive/28
Prediction target			
Recoverability	**0.5078**	**+0.1684**	28/28
Value regression	0.4325	+0.0932	24/28
Uniform example weights	0.4751	+0.1357	28/28
Value gap, no cost adjustment	0.4295	+0.0902	25/28
Value gap including negative values	0.3957	+0.0563	23/28
Feature set (scalar value target)			
Alternative structure (8 features)	0.4266	+0.0873	24/28
Retrieval only (4 features)	0.2669	−0.0725	0/28
Priority based on action scores			
Action score priority	0.2900	−0.0493	3/28

C.5 Recoverability target controls

Recoverability remains useful at different class balances. When recoverable cases make up 8%, 27.7%, or 50% of the source data, it reaches 0.5288, 0.5074, and 0.4850 Capture, respectively, compared with 0.3393 for the best heuristic. In the corresponding 5% budget analysis, cosine alignment with the complete value vector rises from 0.2838 to 0.3274, and recall of the 10% most valuable decision points rises from 0.0839 to 0.1445. The ranker still finds valuable branches when positive source examples become rarer or more common.

Family (branch type) and retrieval score do not reproduce the recoverability ranking (Table 12).

Random selections matched to the ranker's choices by family or by retrieval score range lower Capture from 0.5078 to 0.2577 and 0.2322, respectively. Shuffling recoverability labels within each family yields 0.2652. Keeping these categories while removing their within-category relation to outcomes does not reproduce the learned ranking.

The action score priority control reaches 0.2900. It raises the difference between the maximum and mean initial action score to the power 0.6 and divides by continuation effort. Although motivated by priority-based sampling, this custom score is not an implementation of Prioritized Experience Replay (Pan et al., 2022) and does not evaluate a policy trained with replay.

D Policy learning baselines: implementation and evaluation

D.1 Data identity and information access

The allocation adaptation in Table 2 uses the four synthetic retrieval settings with seed 47001. Each setting has 2,016 source and 1,512 target decision points. The public observation contains ranked document identifiers, their retrieval scores for each occurrence and entity match attributes, candidate document lists, the visible retrieval objective, operation name, and known costs. Document

Table 11: Selected allocation versus representative candidate rules. Means over 28 evaluations; bold denotes the best listed value at each budget. All rows use the same test records.

Scoring rule	10%	20%	30%
CIA (source selected)	**0.6568**	**0.6316**	**0.6694**
All three signals, exponential	0.6482	0.6269	0.6672
Nearest neighbors + phase	0.6513	0.6292	0.6673
Nearest neighbors + recoverability	0.6490	0.6255	0.6656
Nearest neighbors	0.6484	0.6235	0.6598
Nearest neighbors (8)	0.6483	0.6242	0.6617
Phase recoverability	0.5090	0.5084	0.5519
Global recoverability	0.4890	0.5078	0.5572

Table 12: Controls for branch type and retrieval score at a 20% budget. Higher Capture is better; gain is relative to the recoverability ranker.

Selection signal	Capture	Gain
Binary recoverability	**0.5078**	0.0000
Branch type	0.2901	−0.2176
Labels shuffled within type	0.2652	−0.2426
Random within branch type	0.2577	−0.2500
Random within retrieval score ranges	0.2322	−0.2756

identifiers are anonymized independently within each task. Hidden evidence roles, contradictions, simulator repair availability, and oracle alternative indices are excluded. The 12 features used by CIA are reconstructed from these observations and checked against the recorded feature values, so the LLM baselines receive the information from which our representation is derived.

Source and test tasks are disjoint. Methods share source records, test sets, intervention values, and oracle denominators. The reference CIA calculation uses its original source folds and rule selection. Policy baselines receive the text observations from which CIA derives its features, including visible operation context. They do not receive our feature vector, predicted signals, or model conditioned on phase.

D.2 Training environment and test scoring

A source episode contains 20 decision points presented sequentially. The actions are buy (B) and skip (S), with exactly four purchases. Once the quota is exhausted, the remaining actions are forced to S; when the remaining quota equals the remaining points, actions are forced to B. Invalid outputs first default to S and are then subject to quota enforcement and the method's penalty for invalid actions. The observation records the current public decision point and remaining quota. Previous choices are visible, but intermediate source rewards are not disclosed. Similarity anchors use the current visible record without history or budget counters.

For a selected source set S, terminal feedback is

$$R = \mathbb{1}\left[U < \frac{\sum_{i \in S} v_i}{\sum_{j \in \mathrm{TopK}(v,4)} v_j}\right], \qquad U \sim \mathrm{Uniform}(0,1).$$

Repeated rollouts in a group share U, so identical selections receive identical feedback. Expected reward equals episode Capture, with zero reward when the oracle value is zero. Binary feedback retains the success/failure semantics of the baseline credit mechanisms, applied here to allocation. Each trajectory records all 20 source label lookups, four purchases, and any forced or invalid actions. Branching and repeated rollouts reuse source labels; every lookup counts toward training feedback.

At evaluation, the trained LLM scores each of the 1,512 decision points independently with 20 decisions remaining, quota four, and no history. Its score is $\log P(B) - \log P(S)$. This readout converts the trained sequential policy into a ranking of the test set without executing target interventions. A separate evaluator receives the complete ranking and selections before opening outcomes. Selection

takes the first 152, 303, or 454 items at budgets 10%, 20%, or 30%, breaking ties in input order. Candidate identities, score coverage, purchase counts, selected value, and oracle denominators are checked for both the baseline and CIA.

D.3 Learning mechanisms and adaptations

ProxMO uses the author implementation at revision `f7b80a7f`, retaining the trajectory collector, TF-IDF soft grouping, modulation by success rate, advantage estimator, and PPO update. GiGPO and GraphGPO use the author code at `20bd331b`; GiGPO retains episode and state grouping, while GraphGPO constructs transitions from actual source rollouts and computes credit from the graph. This is the selected code version for the allocation experiment, not a claim of exact reproduction of earlier benchmark configurations. ARPO uses author code at `9470fa1f`, retaining branching that adapts to entropy and policy updates through its path for tool interaction.

We implement CARL from the paper's Monte Carlo estimation of action entropy, progressive forking, tree advantages, and selective updates (Shen et al., 2025). The common task permits eight leaves: one initial trajectory and seven expansions. Following the paper, actions on the initial path are excluded from updates. Our HCAPO implementation uses the paper's hindsight verification and multiscale credit (Tan et al., 2026). Both are reimplementations, whereas the other baselines use author code.

C3 uses author revision `628185be` with the value assisted credit variant and coefficient $\alpha = 1$. This branch trains a Q critic and subtracts the resulting counterfactual baseline. The cited paper emphasizes a different formulation that uses only rewards and omits one rollout at a time, without learned parameters; the result is identified as value assisted throughout. StepPO uses author revision `b124aa46` and retains training of the actor and critic at each step with the common purchase environment and readout. StepPO uses LoRA of rank 32 for both actor and critic. C3 and StepPO each select one complete epoch on source validation. Both include critic training in their computational cost.

All policies start from Qwen2.5-7B-Instruct revision `a09a3545`. ProxMO, GiGPO, GraphGPO, ARPO, CARL, and HCAPO use LoRA rank 32 and scale 64. C3 trains a full policy and a full Q critic, with learning rates 10^{-6} and 5×10^{-5} respectively. The LoRA adaptation and purchase interface over a given pool are resource and task adaptations. They are not the original methods' published benchmark settings. ProxMO, GiGPO, and GraphGPO use the shared PPO configuration with learning rate 10^{-5}; ARPO and HCAPO use 10^{-6}, and CARL uses 5×10^{-6}. The retained configuration for each method specifies its native rollout path, grouping or branching, and advantage calculation.

D.4 Training duration and costs

Three tasks from each source setting form a validation set of 189 points, leaving 1,827 source points for selection of training duration. Validation ranks these reserved source points with the same budget readout used at test time. Complete epochs correspond to 46 updates during source validation. If the latest epoch improves Capture at 20% by more than 0.01, another complete epoch is evaluated; otherwise the best observed duration is retained, with ties favoring the shorter duration. No target outcome participates in this decision. The selected duration is then applied while refitting on all 2,016 source records, which require 51 updates per complete epoch.

ProxMO, GiGPO, GraphGPO, ARPO, C3, and StepPO select one epoch (51 formal updates), while CARL and HCAPO select two (102 updates). Formal training uses 102 source sequences with eight trajectories or leaves per sequence in each epoch. Thus a fit over one epoch makes 16,320 lookups of source labels, and a fit over two epochs makes 32,640; runs for source validation incur additional feedback reuse. These totals are training costs, not additional target purchases. CIA directly fits recorded values and recoverability targets and validates its rule library. The comparison matches unique source information and target budgets, rather than equalizing repeated feedback or GPU compute.

Table 13: Training behavior averaged over settings; StepPO covers two settings. Identical feedback is the fraction of groups in which all eight rollouts receive the same binary terminal reward.

Method	Forced actions / 20	Identical feedback (%)
ProxMO	15.508	80.39
GiGPO	15.343	76.96
GraphGPO	15.370	79.17
ARPO	15.229	85.05
CARL	15.407	95.47
HCAPO	15.660	89.83
C3 (value assisted)	14.922	87.01
StepPO (two settings)	15.344	81.37

D.5 Evaluation checks and limits of the comparison

All 32 policy evaluations pass checks of candidate identity, score coverage, purchase count, source/target disjointness, selected value, and oracle denominator. Capture recomputed from stored selections and outcome vectors agrees within 10^{-12}. These checks establish that evaluation uses the intended records and selections. They leave differences in optimization and effective supervision between methods unresolved.

Expected random Capture in each test set is $(k/n)\sum_i v_i / \sum_{j\in \mathrm{TopK}(v,k)} v_j$. Averaging this quantity across settings gives 0.2620 at 20%. This analytic reference shares the pool and denominator with the methods and requires no random training run. Seven of the eight adaptations fall below it; C3's value assisted variant exceeds it. These comparisons describe the observed means without a significance test.

Feedback contrast and forced choices. Training diagnostics are available for 30 runs, including two StepPO settings (Table 13); outcome checks cover all 32. Each episode has 20 decisions and each group has eight trajectories or leaves. Enforcing the four purchase quota forces many actions, and the shared random threshold gives many groups identical terminal feedback. Both reduce the choices or reward contrasts available during training. Their effects can differ across state, hindsight, and critic mechanisms, so the counts neither imply zero gradients nor quantify how much of the performance gap the interface causes.

Test scoring and tied scores. Training uses history and a changing quota, whereas test scoring treats every item as the first decision in a 20/4 episode. The same scores serve the 10% and 30% budgets. Most policies favor buying in this state. Except for ARPO's scoring of the complete string, only 45 to 83 distinct scores occur among the 1,512 target items. BF16 logits and input order tie breaking may affect the cutoff. In the 30 runs with diagnostics, analytic random tie breaking changes each method's mean 20% Capture by less than 0.003. Recorded ties explain little of the gap under this test; the effect of higher precision scoring remains unmeasured.

What this comparison supports. Raw observations let the policies retain their text representations without borrowing CIA's 12 engineered features or predictions. Learning from a terminal Bernoulli reward, however, provides less granular feedback than fitting individual source values. Converting a sequential policy into an independent ranking also changes its decision context. The observed gap includes these adaptation effects. Native execution checks verify that the training mechanisms run, but do not reproduce the papers' benchmark performance or establish that this task uses each method's strengths fully.

D.6 Results for each setting

Table 14 reports CIA and the policy learning adaptations on each of the four settings at three budgets. All eight policy methods have results on all four settings. The mean columns in Table 2 average the four settings equally. All reported baseline values use the single seed 47001; no variability across training seeds is implied. The seven seed internal analyses elsewhere in the paper remain separate.

Table 14: Policy adaptation Capture by setting, seed 47001. C3 is value assisted; CARL and HCAPO are paper implementations.

Setting	Method	10%	20%	30%
Hotpot	ProxMO	0.2453	0.2672	0.3509
	GiGPO	0.2320	0.2478	0.3414
	GraphGPO	0.2533	0.2547	0.3234
	ARPO	0.2634	0.2684	0.3254
	CARL	0.2352	0.2698	0.3684
	HCAPO	0.2568	0.2411	0.3185
	C3 (value assisted)	0.2424	0.2840	0.3775
	StepPO	0.2554	0.2471	0.3437
	Uniform random (expectation)	0.1989	0.2295	0.3003
	CIA	**0.5955**	**0.6331**	**0.7002**
2Wiki	ProxMO	0.2508	0.2862	0.3226
	GiGPO	0.2625	0.2836	0.3162
	GraphGPO	0.2571	0.2939	0.3214
	ARPO	0.2616	0.2920	0.3119
	CARL	0.2961	0.2986	0.3372
	HCAPO	0.2780	0.2879	0.3314
	C3 (value assisted)	0.3134	0.3492	0.3852
	StepPO	0.2574	0.2964	0.3107
	Uniform random (expectation)	0.2362	0.2685	0.3003
	CIA	**0.6868**	**0.6579**	**0.6474**

Table 15: Policy adaptation Capture by setting (continued), seed 47001. C3 is value assisted; CARL and HCAPO are paper implementations.

Setting	Method	10%	20%	30%
MuSiQue	ProxMO	0.2151	0.2573	0.3059
	GiGPO	0.1843	0.2005	0.2713
	GraphGPO	0.1767	0.2165	0.2692
	ARPO	0.1200	0.1773	0.2238
	CARL	0.2132	0.2445	0.2983
	HCAPO	0.1708	0.2207	0.2657
	C3 (value assisted)	0.2014	0.2339	0.2575
	StepPO	0.1740	0.2167	0.2806
	Uniform random (expectation)	0.2477	0.2714	0.3036
	CIA	**0.7330**	**0.6759**	**0.6592**
Bamboogle	ProxMO	0.2267	0.2311	0.2839
	GiGPO	0.1767	0.2214	0.2564
	GraphGPO	0.1698	0.2143	0.2628
	ARPO	0.1715	0.2037	0.2291
	CARL	0.2044	0.2169	0.2661
	HCAPO	0.2030	0.2036	0.2789
	C3 (value assisted)	0.2388	0.2743	0.3187
	StepPO	0.2027	0.2099	0.2671
	Uniform random (expectation)	0.2723	0.2785	0.3088
	CIA	**0.7362**	**0.6649**	**0.6473**

E Analysis of features, rule selection, and robustness

E.1 What the predictors use

The controls in Appendix C.5 preserve family or retrieval score while breaking the remaining relation to outcomes. Their lower Capture means that these properties cannot account for the recoverability ranking on their own. Recoverability also outperforms scalar value regression under the tested ridge model. One possible reason is that candidate sets and effort can change value magnitude even when a useful repair remains available.

Table 16: Feature subsets and residualized scores in Rule 1 (all three signals); outcome values are unchanged. The projection prefilter retains 0.9956 of exact neighbors and nearly reproduces the complete selection. Results use a 20% budget; gain is measured from the best heuristic.

Control	Capture	Gain	Selection overlap
Overlap only	0.5446	+0.2053	0.386
Score residual after overlap fit	0.4426	+0.1033	0.424
Features without overlap	0.3565	+0.0171	0.242
Source normalization	0.6219	+0.2826	0.989
Projection prefilter	0.6233	+0.2840	0.991
Two source settings	0.5911	+0.2517	0.651

Table 17: Mixture sensitivity to signal composition, neighborhood size, and phase noise. Values are mean Capture; noise in phase labels is the fraction of labels replaced at random.

Rule	20% budget	30% budget
Signals		
CIA	**0.6316**	**0.6694**
All three signals	0.6269	0.6672
Nearest neighbors + recoverability	0.6247	0.6630
Nearest neighbors + phase	<u>0.6292</u>	<u>0.6673</u>
Nearest neighbors	0.6235	0.6598
Recoverability	0.5078	0.5572
Equal weights	0.5214	0.5917
Neighborhood size		
32 neighbors	0.6012	0.6538
64 neighbors	0.5526	0.6275
96 neighbors	0.5289	0.6051
128 neighbors	0.5137	0.5848
Noise in phase labels		
25%	0.6253	0.6664
50%	0.6247	0.6653
100%	0.6222	0.6655

Overlap controls. Overlap can mean that every alternative repeats the same evidence, or that several alternatives preserve a useful core while differing on the missing document. Table 16 varies the feature subset and removes from the Rule 1 score the part that a predictor using only overlap can explain. The evaluator and target values remain unchanged.

Overlap alone reaches 0.5446, whereas excluding overlap features gives 0.3565. Source normalization reaches 0.6219 with almost the same purchased set as the full calculation. A projection prefilter that keeps 256 candidates reaches 0.6233 and retains 0.9956 of the exact neighbors. Its selected set has Jaccard overlap 0.991 with the full calculation. These comparisons concern Rule 1. The feature subset variants also use a Gaussian distance kernel instead of the main exponential kernel, so their differences cannot be assigned solely to overlap or treated as feature contributions to the complete allocator.

E.2 Budgets and retrieval settings

Scoring signals. Table 17 compares CIA with signal combinations, neighborhood sizes, and noise in phase labels, defined as the fraction of labels replaced at random.

The table studies mixture sensitivities without rule selection. Its row combining nearest neighbors and recoverability uses weights 0.9/0.1, whereas the corresponding selection candidate uses 0.85/0.15. Its row using equal weights uses 96 neighbors, whereas the selection candidate uses 16. Rows varying neighborhood size and phase noise modify the 0.8/0.1/0.1 mixture. This diagnostic table evaluates the stated mixture variants rather than the complete candidate library.

Table 18: Recoverability and the best heuristic at budgets from 5% to 50%. Entries are Capture; gain is relative to the best heuristic.

Budget	Recoverability	Best heuristic	Gain
Target comparison			
5%	**0.4549**	0.3224	+0.1324
10%	0.4890	0.3219	+0.1671
20%	0.5078	0.3393	+0.1684
30%	**0.5572**	0.3869	+0.1702
50%	**0.7667**	0.5817	+0.1850

Table 19: Effect of training class balance at a 20% budget, and alignment with the complete value vector at 5%. Gain is recoverability minus the best heuristic.

Condition or metric	Recoverability	Best heuristic	Gain
Training prevalence			
Low (8%)	**0.5288**	0.3393	+0.1894
Observed (27.7%)	**0.5074**	0.3393	+0.1681
High (50%)	**0.4850**	0.3393	+0.1457
Credit alignment			
Cosine similarity	**0.3274**	0.2838	+0.0436
Recall in top 10%	**0.1445**	0.0839	+0.0606

Budget sweep. Table 18 compares recoverability with the best heuristic from 5% through 50%.

Recoverability exceeds the best heuristic at every tested budget, with gains from 0.1324 to 0.1850. Its advantage persists as the budget expands from purchasing a small fraction of outcomes to evaluating half of the pool.

Class balance and credit alignment. Table 19 changes the prevalence, or fraction, of positive training examples. It then reports cosine similarity with the complete value vector and recall of its most valuable 10%.

Recoverability exceeds the best heuristic for positive training prevalence from 8% to 50%. At a 5% budget, cosine similarity increases from 0.2838 to 0.3274 and recall of the most valuable 10% rises from 0.0839 to 0.1445. The purchased decisions better match the full value vector, both in direction and in covering its largest entries.

Validation budget. Table 20 reports the primary 20% result and validation at the 30% test budget.

CIA exceeds nearest neighbors by 0.0081 at 20%. Validating at the 30% test budget instead increases the gain over nearest neighbors to 0.0164 and improves Capture by 0.0068 over validation at 20%.

Selection stability. Table 21 evaluates a smaller source pool and an independent second validation split.

With two source settings, CIA improves over nearest neighbors by 0.0128 at 30%. A second validation split improves the 20% result by 0.0105 over the first split and by 0.0223 over nearest neighbors. The split changes which rule is selected. The comparison measures this sensitivity and offers no reason to prefer the second split on future data.

All three signals by setting. Table 22 compares the score using all three signals, nearest neighbors, and the best heuristic at a 30% budget.

At the 30% budget, Rule 1 leads the nearest neighbor rule with exponential weights and the best heuristic in all four synthetic settings. The observation applies only to that budget and mixture.

Table 20: Validation at the primary budget and at the 30% deployment budget. Gain is relative to nearest neighbors; intervals resample the four test corpora.

Budget	CIA	Nearest neighbors	Gain	95% interval
20%	**0.6316**	0.6235	+0.0081	[+0.0012,+0.0150]
30%	**0.6763**	0.6598	+0.0164	[+0.0125,+0.0224]

Table 21: Two source settings at 30% and a second validation split at 20%. Intervals resample matched evaluations; the last column counts positive gains.

Setting	Budget	Gain [95% interval]	Positive
Two source settings vs. Nearest neighbors	30%	+0.0128 [+0.0068,+0.0181]	67/84
Second validation split vs. first	20%	+0.0105 [+0.0054,+0.0156]	21/28
Second validation split vs. Nearest neighbors	20%	+0.0223 [+0.0138,+0.0292]	27/28

E.3 Predictor capacity and distance within signal mixtures

Table 23 varies one component of Rule 1 (all three signals). In its predictor block, each fitted model replaces the global prediction component in $0.8z(s_{\text{prox}})+0.1z(s_{\text{model}})+0.1z(s_{\text{phase}})$. The reported forest, tree, and linear results are therefore fused scores and do not compare standalone predictors. The forest and tree use maximum depths of three and two, respectively. The distance block varies the nearest neighbor calculation with 16 neighbors, except for the narrow radial basis variant with eight. Neither block reruns the full source procedure for selecting rules.

Using a forest for the global component gives 0.6406 Capture at 20%, compared with 0.6269 for the default mixture and 0.6316 for CIA. Predictor and distance choices can improve allocation even with the same input features. Some individual rules tested here outperform the selected allocator.

E.4 Source transfer and label noise

CIA versus nearest neighbors by setting. CIA improves on nearest neighbors in ten of the twelve combinations of setting and budget and leads on every corpus at the 30% budget. Its aggregate advantage is positive at all three budgets and is largest at 30%; the exceptions within individual settings and source subset results are detailed below.

Label noise. Table 24 perturbs source labels for Rule 1 while leaving target candidates and outcomes unchanged. It does not repeat the rule selection of CIA. The symbol σ is the standard deviation of added Gaussian noise; selection overlap is intersection size divided by union size relative to the selection with no added noise. In the control with zero values, every source value is set to zero before fitting Rule 1; this does not remove just one predictor. Figure 5 visualizes all eight conditions.

Gaussian noise up to $\sigma = 0.20$ retains at least 0.903 selection overlap and changes Capture by less than 0.007. Shuffling source values gives 0.2679 Capture and 0.122 overlap. Moderate random errors leave much of the ranking intact, while breaking the relation between source features and values removes most of its advantage.

E.5 Diagnostics of source transfer

Figure 6 shows how gains over nearest neighbors vary across budgets and amounts of source data.

CIA improves 10 of the 12 means over corpora and budgets in the upper panel. Its only negative corpus means occur on MuSiQue at 10% (-0.0011) and 20% (-0.0010); all corpora have positive gains at 30%.

Table 22: Rule 1 (all three signals), nearest neighbors, and the best heuristic at a 30% budget. Rows average seven seeds; gain is relative to nearest neighbors.

Corpus	All three signals	Nearest neighbors	Best heuristic	Gain
30% *budget*				
Bamboogle	**0.6630**	0.6593	0.3898	+0.0037
HotpotQA	**0.6703**	0.6569	0.3846	+0.0134
MuSiQue	**0.6663**	0.6614	0.3790	+0.0049
2Wiki	**0.6691**	0.6617	0.3792	+0.0074

Table 23: Selected predictor and distance variants in Rule 1 (all three signals) at a 20% budget. Predictor rows replace only the global component, retaining nearest neighbors and phase. Gain is relative to the exponential mixture, not CIA.

Setting	Capture	Gain
Global predictor within fusion		
Random forest	**0.6406**	+0.0137
Decision tree	0.6295	+0.0027
Penalized linear model (binary)	0.6266	−0.0003
Boosted decision stumps	0.6262	−0.0007
Penalized linear model (value)	0.6260	−0.0008
Robust linear model (value)	0.6249	−0.0020
Small multilayer perceptron (value)	0.6216	−0.0053
Logistic model (binary)	0.6207	−0.0061
Distance weighting within fusion		
Euclidean, radial basis, narrow	**0.6260**	−0.0008
Euclidean, radial basis	0.6234	−0.0035
Euclidean, uniform	0.6219	−0.0049
Cosine, radial basis	0.6069	−0.0199
Ranked Euclidean, radial basis	0.6023	−0.0245

Reducing the training pool from three source settings to two preserves a positive mean gain over nearest neighbors. At 30%, the gain is 0.0128 [0.0068, 0.0181] across the three possible source subsets and is positive in 67 of 84 evaluations. At 20%, the mean remains positive, but the interval includes zero. Using a second validation split raises the 20% gain to 0.0223 [0.0138, 0.0292] and produces positive gains in 27 of 28 evaluations. These results show sensitivity to source coverage and the validation split; they do not establish a monotonic benefit from adding source data.

E.6 Additional robustness comparisons

Recoverability remains stronger than the best heuristic when the recorded candidate pool is truncated for evaluation. With two retained actions, it reaches 0.4396 capture, compared with 0.3000 for the best heuristic; with three, the comparison is 0.4765 to 0.2706. The truncation uses the original representation and effort, as detailed in Appendix F.2.

Flipping 10% of source labels lowers three signal Capture from 0.6269 to 0.6157 and leaves a selection overlap of 0.804 with the unperturbed ranking. Shuffling values gives 0.2679 Capture and 0.122 overlap. The mixture tolerates some random label errors but still requires source interventions whose outcomes are informative about the target.

Price adjustment has a conditional benefit in the comparison of mixture scores. At the same spend, ranking by predicted value per unit price raises capture from 0.6276 to 0.6406 when prices depend on branch family, and from 0.5820 to 0.6414 under the most uneven prices. It changes little for nearly uniform prices or estimated human review time (Table 25).

Table 24: Source label perturbations for Rule 1 (all three signals). Gaussian noise has standard deviation σ; selection overlap is the intersection divided by the union with the unperturbed selected set.

Label change	Capture	Overlap with no noise
No added noise	**0.6269**	**1.000**
Gaussian noise ($\sigma = 0.05$)	0.6245	0.966
Gaussian noise ($\sigma = 0.10$)	0.6202	0.940
Gaussian noise ($\sigma = 0.20$)	0.6210	0.903
5% label flips	0.6172	0.868
10% label flips	0.6157	0.804
Shuffled values	0.2679	0.122
Zero source values	0.2613	0.112

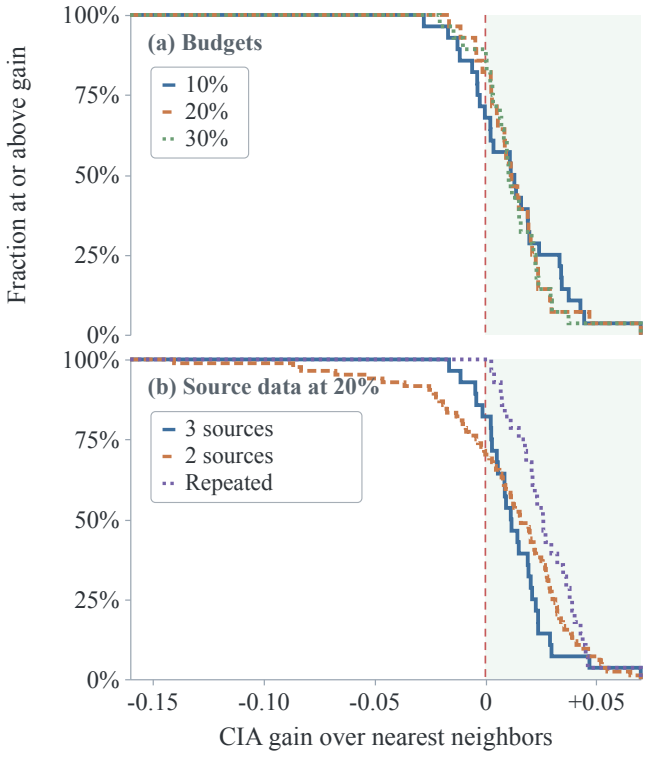

Figure 6: CIA gains over nearest neighbors. Curves show the fraction of evaluations at or above each gain; zero is dashed. The upper panel compares budgets. At 20%, the lower panel compares training on three source settings, training on two, and using a second validation split.

F Candidate coverage, costs, and observed outcomes

F.1 Unequal intervention costs

For nonuniform intervention costs, v_i remains the repair value and d_i denotes purchase price. Both the selected set and the greedy oracle set satisfy $\sum_{i \in S} d_i \leq B$. The greedy oracle fills the budget in descending v_i/d_i order, while the learned selector ranks by predicted value per unit price. Capture sums v_i for both sets. Thus e_i defines repair value normalized by effort, whereas d_i determines what can be purchased. The oracle is greedy and only approximates the exact binary knapsack solution.

Table 25: Capture with and without price adjustment under four purchase price models. Total price is $\sum_i d_i$; rows under each model share the same budget.

Price definition	Ranking	Capture	Total price
Shared work	Best heuristic	0.3355	849.4
	All three signals	**0.6208**	849.6
	Score per unit price	0.6206	849.5
Runtime by family	Best heuristic	0.3519	463.4
	All three signals	0.6276	463.3
	Score per unit price	**0.6406**	463.4
Uneven runtime	Best heuristic	0.3163	419.7
	All three signals	0.5820	419.7
	Score per unit price	**0.6414**	419.7
Human review	Best heuristic	0.3680	497.1
	All three signals	**0.6269**	497.2
	Score per unit price	0.6209	497.1

Table 26: Recoverability under candidate pool truncation and unequal purchase prices. Gain is measured from the best heuristic.

Condition	Recoverability	Best heuristic	Gain
Retained candidate actions			
2	**0.4396**	0.3000	+0.1396
3	**0.4765**	0.2706	+0.2060
4	**0.5078**	0.3393	+0.1684
Purchase budget			
10%	**0.5467**	0.3928	+0.1539
20%	**0.6930**	0.4839	+0.2090
30%	**0.7401**	0.5640	+0.1761

Cost adjustment. The main evaluation uses equal purchase slots. Table 25 introduces a purchase price d_i under four price models. Shared work reflects continuation work common to several alternatives, runtime by family assigns a multiplier to each branch family, uneven runtime adds larger variation among branches, and human review estimates review time from ambiguity, redundancy, and query edits. This purchase price is distinct from e_i, which normalizes the reported repair value of a completed intervention.

Within each price definition, both sets obey $\sum_i d_i \leq B$, and Capture uses the same v_i in its numerator and denominator. The greedy oracle considers decision points in descending v_i/d_i order and buys each item that fits. The price-adjusted selector follows this rule using predicted score divided by d_i. Capture rises from 0.6276 to 0.6406 when prices depend on branch family and from 0.5820 to 0.6414 under strongly uneven runtime. It changes little for nearly uniform prices or estimated human time.

F.2 Truncation of the candidate pool and purchase budgets

Recoverability remains above the best heuristic when evaluation is restricted to the first two or three recorded candidate actions (Table 26).

With two candidates, Capture is 0.4396 versus 0.3000; with three, it is 0.4765 versus 0.2706. We recompute the initial and best action within the retained set, while keeping the original features, effort, and repair gate. The comparison changes the outcome target without regenerating candidate descriptions. Under cost budgets, the recoverability gains range from 0.1539 to 0.2090.

Table 27: The gain over the best heuristic is positive for every corpus and seed. Entries are mean gains across seven seeds.

Corpus	10%	20%	30%
Bamboogle	+0.3998	+0.3311	+0.2724
HotpotQA	+0.2274	+0.2346	+0.2842
MuSiQue	+0.3807	+0.3074	+0.2849
2WikiMultiHopQA	+0.3318	+0.2958	+0.2884

F.3 Additional simulation results

F.3.1 Heuristic comparisons by setting

The advantage over the best heuristic holds in every setting (Table 27).

At 20%, mean gains range from 0.2346 on HotpotQA to 0.3311 on Bamboogle, and every comparison for an individual seed is positive. The same pattern holds at 10% and 30%.

F.3.2 Coverage of source families

Table 3 removes each training family in turn and refits the estimators on the remaining records while retaining the previously selected rule. The columns for repair rate and mean value summarize the target family's outcome distribution. The acquired column divides value purchased from that family by all its value, rather than by a top-k oracle for that family. The table in the main text reports all seven families with both descriptive statistics and results after source removal.

Removing any source family lowers overall Capture. Repeated evidence and missing context produce the largest drops, and hop order the smallest. Removal reduces the source count as well as changing its composition, so the differences reflect dependence on the available source pool and not the contributions of individual features.

F.3.3 Alignment with complete repair values

At a 5% budget, recoverability improves cosine alignment with the full vector of repair values from 0.2838 to 0.3274 and recall among the most valuable 10% from 0.0839 to 0.1445 over the best heuristic. The selected decisions thus cover more of the value in the complete set of repair values. Episode support then aggregates those repairs along a trajectory.

F.3.4 Support across complete episodes

The allocation unit is a decision point; here we aggregate candidate outcome scores within each trajectory. A selected decision point receives the highest recorded outcome score, and an unselected one keeps the executed action's score. Let $\mathcal{I}_t$ contain the decisions in trajectory t, with T trajectories in total. Write $u_i = s_{\text{eval}}(a_i^{\text{exec}})$ for the executed action's score and $u_i^\star = \max_{a \in \mathcal{A}_i} s_{\text{eval}}(a)$ for the best candidate's score. For $\mathcal{I} = \bigcup_t \mathcal{I}_t$, the mean score and normalized episode support are

$$U(S) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} \left[u_i + \mathbb{1}[i \in S](u_i^\star - u_i) \right],$$

$$\text{Support}(S) = \frac{U(S) - U(\varnothing)}{U(\mathcal{I}) - U(\varnothing)},$$

so Support is the gain over selecting nothing divided by the gain from selecting every decision point, and it is zero when this denominator is zero. Each trajectory thus has equal weight, and we normalize after averaging over trajectories. This aggregation uses neither the repair gate g_i nor the division by effort in v_i and involves no policy update. At a 20% budget, Rule 1 (all three signals) gives 0.3331 normalized episode support, compared with 0.3323 for nearest neighbors, 0.2670 for the best heuristic, and 0.2015 for uniform selection (Table 28). As in the decision point results, nearest neighbors nearly match Rule 1, so values of similar source examples supply most of the

Table 28: Normalized episode support from the recorded candidate outcomes. Intervals resample the seven test seeds; gain is relative to the best heuristic.

Ranking	Episode support	95% interval	Gain
All three signals	**0.3331**	[0.3262,0.3404]	**+0.0661**
Nearest neighbors	0.3323	[0.3258,0.3388]	+0.0653
Best heuristic	0.2670	[0.2597,0.2745]	0.0000
Uniform random	0.2015	[0.1987,0.2042]	−0.0655

Table 29: CIA gain over the best recorded scoring control for each test set and budget. Positive values favor CIA.

Test set	20% budget		30% budget	
	Best control	Gain	Best control	Gain
Search A	Tool failure	+0.0606	Criticality	+0.0531
Search B	Tool failure	−0.0073	Branch divergence	+0.0308
Search C	Criticality	+0.0773	Branch divergence	+0.0577
WebShop A	Action count	+0.4965	Search state	+0.3864
WebShop B	Action count	−0.0203	Reward of the executed action	−0.0489
ALFWorld	Criticality	+0.0000	Criticality	+0.6237

useful ranking information. Rule 1 keeps a 0.0661 gain over the best heuristic when outcomes are aggregated within trajectories.

F.4 Recorded interactive tasks

The evaluation on recorded tasks fits on each domain's source split and ranks its test split using the recorded representation. For recoverability, τ is the 25th percentile of the positive repair values in that domain's source split, or 10^{-9} if none are positive; synthetic search uses $\tau = 0.025$. The threshold uses no test outcomes. Source validation uses a deterministic approximately 70/30 holdout and chooses among six rules: two mixtures of all three signals using Gaussian weights or $\alpha_j = 1/(d_j + 10^{-6})$, nearest neighbor regression using 16 or 8 source points, equal signal weights, and global recoverability. Synthetic search instead compares 11 rules by holding out each source setting in turn.

Search uses 16 recorded features; WebShop and ALFWorld use domain-specific state and repair descriptors. Search uses three splits with 2,560 source and 2,560 test decision points each. WebShop uses two splits with 90/90 and 257/254 source/test decisions. ALFWorld uses 18 source and 12 test repair decisions. Search measures evidence recovered after the remaining search is run from an alternative; WebShop records assign observed contribution to shopping decisions; and ALFWorld records indicate whether a changed branch repairs the subsequent trajectory.

Every record associates a decision point with an observed nonnegative repair value, so Capture with an equal cost for every intervention uses the same top k denominator as the synthetic study. At 20%, the selected counts are 512 in each Search split, 18 and 51 in WebShop, and three in ALFWorld; at 30%, they are 768, 27 and 77, and four.

The Search baselines rank turns by predicted tool failure, branch divergence, or the recorded criticality score. The WebShop baselines rank by the number of available actions, whether the page is a search state, or the reward already attached to the executed action. The reward of the executed action is treated as already observed. This replay evaluation assumes that the recorded descriptors are available before purchasing the additional outcome; their names alone do not guarantee this timing in a live deployment. The recorded criticality score is the best control at both ALFWorld budgets.

F.5 When the method applies

When the method is most useful. CIA requires many proposed interventions, a limited purchase budget, and a source pool of evaluated decisions. Related source interventions supply estimates of repair value from similar examples; source validation chooses how to rank a new corpus. Dividing scores by price helps under the tested runtime prices that vary by family or vary widely across branches. It gives no improvement when prices reflect continuation work shared across alternatives or human review time.

From allocation to learning. Capture isolates acquisition quality under the same candidate set and spend. The selected outcomes could subsequently supervise a critic or policy update, but this study does not measure the learning gain from such an update.

F.6 Interpretation

Why does a binary target help? The scalar value v_i changes with the candidate set, continuation cost, and evaluator scale. Binary recoverability asks whether the available alternatives contain a useful correction, while nearest neighbor regression estimates the likely magnitude of that correction.

Why do nearest neighbors help? Decision points close in feature space have similar ambiguity, overlap, and alternative structure. Their source values yield 0.6235 Capture through nearest neighbor regression alone. Source validation lets CIA choose another neighborhood, distance weighting, or auxiliary signal. Its gain over default nearest neighbors is small and varies with the source data and purchase budget.

Practical interpretation. A review is most useful when the candidates include a plausible repair and related source interventions suggest that it is valuable. Uncertainty or an error in the executed action is not enough on its own. This matters most when continuations, tool calls, or human reviews account for much of the training cost.

G Reproducibility

The accompanying code and data archive contains the corpus, seed, features available before intervention, alternative values, recoverability label, and intervention cost for each decision point. The analysis fits source models, constructs neighbor pools, selects scoring rules using validation on source settings, scores the test candidates, and regenerates the reported statistics.

The synthetic feature and direct allocation studies use seeds 47001 through 47007; each seed resamples tasks, retrieval rankings, and repair availability while keeping source and test tasks disjoint. The threshold for positive value is 0.025 and the regularization penalty is 0.01. Validation compares the 11 rules listed above at a 20% budget, using only Capture on source settings reserved for validation.

Policy learning adaptations use seed 47001 and the interfaces in Appendix D. The studies of recorded tasks use the domain-specific representations and holdout procedure in Appendix F.4. Ties are resolved by input order, and each resampling script records its random seed. The released records retain the corpus split, candidate set, executed and best action indices, both score vectors, repair value, selected rule, and comparison result used to generate the reported tables and comparisons of neighbor search.