LEARNING TO SELECT EVIDENCE FROM COALITION FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models using retrieved passages or stored memories must select useful evidence under a context budget. A common utility measure, leave-one-out deletion, misses useful passages when another passage supplies the same fact. On an exact-copy stress test, its supporting-passage AUROC is 0.50, compared with 0.84 for a Banzhaf coalition teacher that measures contributions across different subsets. We use this coalition feedback to train CAMS, a 462,849-parameter setwise selector whose credit objective requires no supporting-passage labels and whose online selection makes no generative language-model calls. Across five pool settings, credit supervision yields answer F1 close to passage-label supervision in most evaluated budgets. On a 100-passage hard-negative pool at budget eight, the pure credit-trained selector improves answer F1 from 0.608 to 0.647 over dense retrieval. Its forward pass takes 1.2 ms on an H20, compared with 3.6 s for LLM reranking and 8.4 s for prompted set selection; measured embedding, feature, and forward components total 65 ms. Coalition feedback thus turns an offline measure of contextual usefulness into inexpensive evidence selection.

1 INTRODUCTION

A language model answering a question from retrieved passages or stored memories has two budgets: how much evidence it can read, and how much computation it can spend choosing that evidence (Lewis et al., 2020; Du et al., 2026). Selecting every passage spends context on distractions and repeated facts; relevant information can also become harder to use as context grows (Shi et al., 2023; Liu et al., 2023; Yoran et al., 2023). The selection problem is therefore to retain useful evidence in a small context at a cost that leaves room for answering the question.

Consider two passages that both state the same access code. Remove either one, and the model can still read the code from the other (Figure 1a). A leave-one-out (LOO) score—the drop in answer quality after deletion—assigns zero to both, just as it would to an irrelevant passage. The issue is the context in which usefulness is measured: LOO always leaves every alternative source present. In our exact-copy test, this failure appears directly in supporting-passage AUROC: 0.500 for LOO and 0.837 for a coalition-based teacher.

Coalition feedback changes this conditioning. We evaluate a passage with different subsets of its companions, including subsets where its substitutes are absent, and average its marginal contribution. This is the Banzhaf value (Banzhaf III, 1964). For the two-copy example, each passage contributes in half of the contexts and receives credit $1/2$. The calculation supplies a training signal about what helps the reader, using question–answer pairs to define utility.

Computing these interventions for every new query would be expensive. We instead use them offline to train CAMS, a small setwise selector over frozen embeddings and retrieval features. At inference, one forward pass scores all candidates; the selector never calls the generative reader. Amortizing attribution has a substantial history (Jethani et al., 2021; Covert et al., 2024). Our question is whether coalition feedback can train a useful passage selector without supporting-passage labels in its loss, and where that selector fits relative to more expensive rerankers.

The experiments separate three parts of this question. Oracle attribution diagnoses what the feedback measures. Matched credit-only, label-only, and joint objectives test the supervision source. Deployable selection and timing then measure the resulting quality and cost. This separation mat-

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

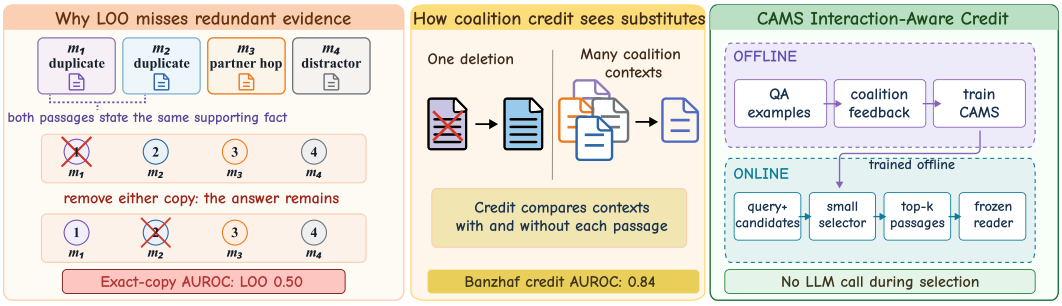

Figure 1: **LOO misses redundant evidence; coalition credit exposes substitutes and trains CAMS.** (a) Duplicate passages preserve the answer after either deletion (LOO AUROC 0.50). (b) Coalition credit compares contexts with and without each passage (AUROC 0.84). (c) Offline feedback trains CAMS for LLM-free online selection before a frozen reader.

ters because both copies can receive useful credit and still occupy two slots in a top-k ranking. We therefore test explicit deduplication with the same rule available to a Dense baseline.

On Hard-100 at $k = 8$, credit-only CAMS improves F1 over Dense by 0.0395, while its forward pass costs 1.2 ms. A cross-encoder offers higher quality at this budget, and generative selectors are stronger in several small-budget settings. The practical benefit is a low-overhead selection option trained from reader feedback; cross-encoder fusion and deduplication provide separately measured extensions. Our contributions are:

1. **A redundancy diagnosis.** A two-passage example explains LOO's blind spot, and an exact-copy test measures AUROC of 0.500 for LOO versus 0.837 for coalition credit.
2. **Coalition-supervised selection.** A 462,849-parameter setwise model learns from credit without supporting-passage labels in its objective. Five pool settings and three seeds compare this objective with label-only and joint supervision.
3. **A measured quality–cost trade-off.** Unified evaluation finds a 0.0395 F1 gain over Dense on Hard-100 at $k = 8$, with a 1.2 ms selector forward pass. Matched diversity controls and a second reader locate the regimes in which the approach helps.

2 Related Work

Context selection. BM25 and dense retrieval score relevance inexpensively (Robertson & Zaragoza, 2009; Xiao et al., 2024; Multi-Granularity, 2024). Cross-encoders and listwise LLM rankers spend additional computation on query–passage matching (Nogueira et al., 2020; Sun et al., 2023). Prompt compression reduces retained text (Jiang et al., 2023; Xu et al., 2024; Hwang et al., 2024), while self-reflective retrieval incorporates decisions into generation (Asai et al., 2024). MMR balances relevance and diversity (Carbonell & Goldstein, 1998); SetR learns set selection (Lee et al., 2025). We compare with retrieval, a BGE cross-encoder, LLM listwise ranking, and a SetR-inspired prompt. The latter uses the evaluation reader with no SetR fine-tuning. Our contribution concerns the supervision of a small selector; the final top-k rule and optional geometric deduplication remain explicit design choices.

Attribution as supervision. Removal-based explanations depend on their payoff and intervention distribution (Covert et al., 2021). Shapley and Banzhaf values average marginal contributions over coalitions with different weightings (Shapley, 1953; Lundberg & Lee, 2017; Banzhaf III, 1964), and related approaches value training data or graph nodes (Ghorbani & Zou, 2019; Wang & Jia, 2023; Wang et al., 2025). Coalition analyses also attribute generated text and tokens (Enouen et al., 2024; Xiao et al., 2025). ContextCite fits a response-specific surrogate (Cohen-Wang et al., 2024); SCARLet and CoRM-RAG derive retrieval signals from contextual utility and counterfactual robustness (Xu et al., 2025; Liu et al., 2026). Source attribution, response-distribution changes, efficient LOO approximation, and influence-based selection offer additional utility measures (Nematov et al., 2025; Li et al., 2025; Liu et al., 2025; Deng et al., 2026). Causal memory selection studies utility beyond

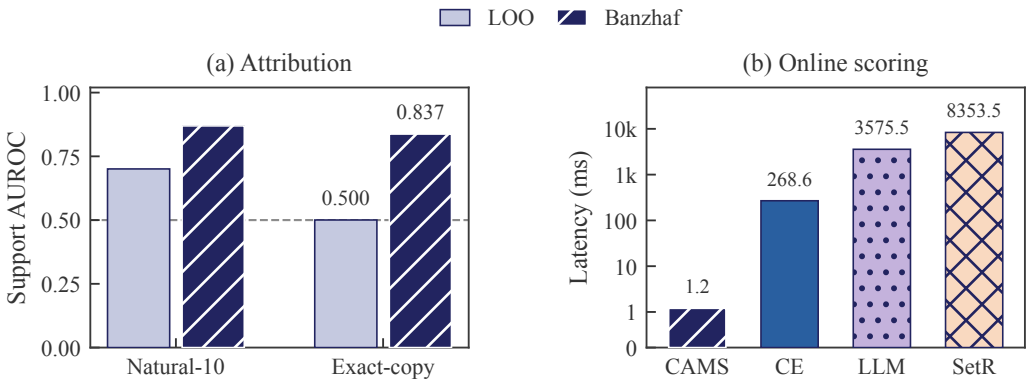

Figure 2: **Coalition feedback preserves signal under duplication and supports inexpensive online scoring.** (a) Pooled support AUROC for answer-aware LOO and Banzhaf teachers. (b) Mean component latency over 40 Hard-100 requests on one H20; CAMS is forward-only, LLM is listwise, and SetR is prompted. The latency axis is linear through 1 ms and logarithmic above it.

static QA (Srivastava, 2026). We focus on answer-quality feedback and evaluate its redundancy sensitivity and downstream use separately.

Amortized attribution. FastSHAP and subsequent methods learn predictors of costly attributions (Jethani et al., 2021; Yang et al., 2023; Covert et al., 2024). Stochastic Amortization includes Banzhaf targets and studies learning from noisy estimates. Building on this principle, we investigate small setwise passage selectors, compare supervision sources under matched architectures, and measure their deployment costs. The new evidence concerns redundant context and budgeted passage selection; the Banzhaf estimator and the general amortization principle are established tools.

3 Method

3.1 Coalition feedback

Let $\mathcal{M} = \{m_1, \ldots, m_n\}$ be a query's candidate passages and R a frozen reader. With gold answer $a^\star$, the utility of a subset is

$$v_\pi(S) = \mathrm{F1}\big(R(q, \pi(S)), a^\star\big), \qquad S \subseteq \mathcal{M}, \tag{1}$$

where $\pi(S)$ specifies passage order. Attribution uses the stored pool order restricted to S. Selection evaluation uses each method's ranking order, with dense-score tie-breaking, and Write-All retains stored order. Hence downstream F1 reflects the selected passages and their presentation order. Below we fix π and suppress it.

For passage i, full-context deletion and Banzhaf credit are

$$\ell_i = v(\mathcal{M}) - v(\mathcal{M} \setminus \{i\}), \tag{2}$$

$$c_i = 2^{-(n-1)} \sum_{S \subseteq \mathcal{M} \setminus \{i\}} [v(S \cup \{i\}) - v(S)]. \tag{3}$$

The two-copy example is the unit-payoff OR game, $v(S) = \mathbf{1}\{i \in S \text{ or } j \in S\}$. It gives LOO scores $(0, 0)$ and Banzhaf credits $(1/2, 1/2)$; an irrelevant passage receives zero. The distinction is specific to substitutes. For a complementary AND pair, $v(S) = \mathbf{1}\{i, j \in S\}$, LOO assigns $(1, 1)$ and Banzhaf assigns $(1/2, 1/2)$. LOO detects both necessary passages and attributes the full joint gain to each. Banzhaf credit averages contextual usefulness; in general it neither sums to total utility nor solves the best-subset problem.

At training width $n = 10$, we evaluate all 1,024 subsets and average credit over four reader runs. Exhaustive enumeration removes coalition-sampling error, while repeated runs account for variation in

batched reader execution. The held-out redundancy tests use sampled teachers; Paraphrase uses 256 antithetic coalitions in one reader run. Both LOO and these teachers require answers at evaluation time, so their scores serve as oracle diagnostics.

3.2 SETWISE LEARNING

We call the coalition signal RICA and its learned selector CAMS. Each passage is represented by four frozen 384-dimensional embedding blocks—passage, query, absolute difference, and element-wise product—and four normalized features: dense similarity, BM25, a surface heuristic, and token overlap. A two-layer, four-head set encoder with padding masks predicts a residual r_i. The score is

$$h_i = z(d_i) + \lambda r_i, \qquad \lambda = 0.5, \tag{4}$$

where d_i is the dense prior and z normalizes within the pool. The network has 462,849 parameters and uses no positional embeddings. Pool-dependent scores allow candidates to affect one another's ranking.

Our primary variant, **CAMS-C**, learns solely from credit in its task loss. Let $\bar{c} = z(c)$, $p_c = \mathrm{softmax}(c)$, and u_i denote the standard deviation of credit across reader runs. The objective is

$$\mathcal{L}_C = \mathrm{CE}(p_c, \mathrm{softmax}(h)) + 0.45 \sum_i \omega_i \, \mathrm{smooth}_{L_1}(h_i, \bar{c}_i) + 0.02 \, \frac{1}{n} \sum_i r_i^2, \tag{5}$$

with $\omega_i = \mathrm{clip}(0.05/(u_i + 0.05), 0.15, 1)$. The soft listwise term learns a ranking, regression uses the numerical credits, and the last term regularizes the residual. Two matched arms identify the role of the target: **CAMS-L** uses listwise, binary, and pairwise supporting-label losses; **CAMS-B** adds credit regression to those losses. All arms share architecture, training pools, optimization budget, and seeds (Appendix A).

Credit training needs question–answer pairs to compute Eq. (1). Its loss omits supporting-passage labels. Development annotations remain relevant: the original checkpoint rule uses support recall, and the optional deduplication threshold uses support coverage. We therefore restrict the annotation-saving claim to the training objective.

3.3 SELECTION AND EXTENSIONS

The pure selector ranks h in one forward pass. Optional fusion combines it with a named base ranker b:

$$s_i = (1 - \alpha) z(b_i) + \alpha z(h_i), \qquad S_k = \text{top-}k_i s_i. \tag{6}$$

For each base, seed, and budget, development F1 selects $\alpha \in \{0, .25, .5, .75, 1\}$; ties prefer fewer words, then the first grid value. The endpoints recover the base or pure selector. We report "+CE" separately because its cost includes cross-encoding. Fusion with an LLM base also incurs online generation.

To handle repeated evidence, **+dedup** scans a ranking and postpones candidates whose BGE cosine similarity to a retained passage exceeds a threshold. Deferred candidates are appended after the retained ones, allowing the ranking to fill larger budgets. Each ranker's threshold is chosen by development distinct-support coverage and shared across budgets. Dense+dedup provides a matched control. This rule addresses a remaining property of scalar ranking: useful substitutes can both score highly and consume scarce slots.

Pure CAMS-C, +CE fusion, and +dedup test different properties: supervision quality, score complementarity, and set diversity. Setwise representations still produce scalar scores; top-k does not reevaluate the chosen combination. Useful substitutes can therefore occupy multiple slots. The matched deduplication controls test this allocation problem, while Section 5 measures remaining subset-level headroom.

4 EXPERIMENTS

4.1 EXPERIMENTAL DESIGN

We use 1,500 HotpotQA training questions and disjoint development and test sets of 250 questions each (Yang et al., 2018). Figure 3 compares selection quality across passage budgets. Five controlled

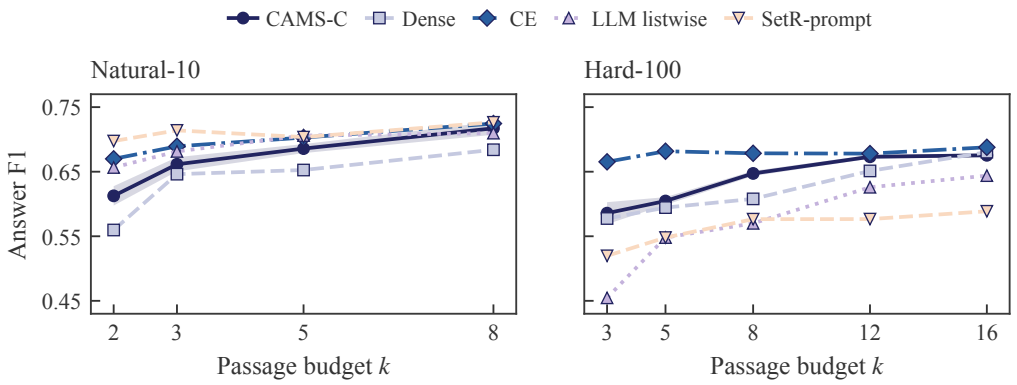

Figure 3: **The quality ordering changes with passage budget and pool construction.** Unified Qwen evaluation on 250 questions per pool; CAMS-C uses pure credit scores. Shading denotes three-seed SD for CAMS-C. All methods share each pool's evaluation session. The F1 axis is truncated.

pools vary the selection problem: Natural-10, Random-100, Hard-100, Exact-copy, and Paraphrase. The wide pools add donor distractors; Hard-100 chooses them by query similarity. Exact-copy duplicates supporting evidence, and Paraphrase replaces injected copies with generated rewrites. Training is repeated per held-out pool and seed using the same training aggregates. These are five pool constructions within one dataset. Pool filters and training details are in Appendix A.

The primary reader is Qwen2.5-14B-Instruct (Hui et al., 2024). Llama-3.1-8B-Instruct (**?**) evaluates three pools using the stored Qwen-trained scores. We use greedy decoding through vLLM (Kwon et al., 2023). Baselines include Random, Surface, BM25, Dense, MMR, BGE-reranker-v2-m3 (**?**), LLM listwise, SetR-prompt, and Write-All. SetR-prompt asks the reader to identify information needs and select up to k passages. Within each pool and reader, all methods share an evaluation process and identical ordered contexts are evaluated once.

Learned-method results give mean F1 $\pm$ population SD across seeds 101, 202, and 303. Paired 95% bootstrap intervals use 10,000 question resamples after averaging each question's results across seeds (Tibshirani & Efron, 1993); they capture question-level uncertainty conditional on these models. Tests are exploratory, with no multiplicity adjustment or equivalence margin. Full budgets appear in Appendices B–C.

The evaluation follows the method's causal chain through four questions. First, does redundancy erase the diagnostic signal of full-context deletion? Second, does credit train a selector that improves answer quality under a fixed budget? Third, can credit replace supporting-passage labels when the architecture and optimization budget are held fixed? Fourth, what computation is paid offline and online? Figure 2 answers the first and fourth at the component level; Tables 1 and 2 answer the two learning questions. Reader transfer and explicit diversity are evaluated separately because neither follows from attribution quality alone.

Testing the deletion diagnosis. Natural-10 gives pooled AUROC of 0.701 for LOO and 0.869 for the teacher. With exact copies, these become 0.500 and 0.837 (Figure 2a); per-query macro AUROC is 0.502 and 0.830. The same difference reaches downstream answers: at $k = 3$ on Exact-copy, oracle LOO scores 0.555 F1 and the oracle teacher 0.796, a paired gain of 0.241 (95% CI $[0.182, 0.298]$, $p < 0.0002$). Natural-10 LOO reaches 0.731 at that budget, locating the weakness in the redundant condition. Width-100 LOO is recomputed at the actual width and gives AUROC 0.644 on Random-100 and 0.635 on Hard-100. These answer-aware diagnostics are collected separately in Appendix E.

4.2 Selection Quality

Pure credit-trained selection improves on inexpensive retrieval in several regimes (Figure 3; Table 1). At $k = 8$ on Natural-10, CAMS-C scores 0.717, compared with Dense's 0.684 and MMR's 0.677.

Table 1: **CAMS improves low-cost retrieval, while stronger scorers remain competitive at tight budgets.** Qwen answer F1 on 250 test questions; learned rows report three-seed mean ± SD. Pure rows use no external reranker, and +CE rows include cross-encoder scoring.

	Natural-10		Random-100	Hard-100		
Method	$k=3$	$k=8$	$k=8$	$k=3$	$k=8$	$k=12$
BM25	0.515	0.680	0.604	0.455	0.549	0.630
Dense	0.646	0.684	0.648	0.578	0.608	0.651
MMR	0.634	0.677	0.647	0.578	0.612	0.638
CE	0.689	0.725	0.690	0.665	0.679	0.678
LLM listwise	0.681	0.710	0.603	0.455	0.570	0.626
SetR-prompt	0.714	0.727	0.570	0.520	0.577	0.577
CAMS-C (pure)	0.662 ± 0.011	0.717 ± 0.010	0.675 ± 0.003	0.586 ± 0.017	0.647 ± 0.004	0.673 ± 0.003
CAMS-L (pure)	0.691 ± 0.009	0.724 ± 0.005	0.688 ± 0.002	0.572 ± 0.005	0.641 ± 0.003	0.667 ± 0.009
CAMS-B (pure)	0.684 ± 0.010	0.724 ± 0.008	0.674 ± 0.018	0.577 ± 0.003	0.652 ± 0.003	0.665 ± 0.006
CAMS-C + CE	0.679 ± 0.009	0.727 ± 0.001	0.690 ± 0.000	0.655 ± 0.009	0.681 ± 0.002	0.717 ± 0.003
CAMS-B + CE	0.687 ± 0.002	0.734 ± 0.007	0.714 ± 0.006	0.665 ± 0.000	0.705 ± 0.002	0.707 ± 0.004

On Hard-100, its 0.647 exceeds Dense by 0.0395 (95% CI $[0.0125, 0.0675]$, $p = 0.0042$) and MMR by 0.0355 ($p = 0.0136$). These comparisons establish a downstream use for the credit target beyond the oracle attribution diagnosis.

Additional reranking computation can buy higher quality. On Hard-100 at $k = 8$, CE reaches 0.679 and CAMS-B+CE reaches 0.705. At $k = 12$, CE scores 0.678, CAMS-B+CE 0.707, and CAMS-C+CE 0.717. These fused gains are numerical comparisons; paired tests for the fused rows are unavailable. The fusion weights are selected separately for each arm, seed, and budget, so these rows describe configurations with both learned scores and a cross-encoder.

Budget also changes which method is preferable. On Natural-10 at $k = 3$, SetR-prompt reaches 0.714 against CAMS-C's 0.662. On Hard-100 at $k = 3$, CE reaches 0.665 against CAMS-C's 0.586 ($p = 0.0010$). LLM listwise and SetR-prompt perform less well on wide pools under the present prompt and decoding budgets; their implementation details in Appendix A delimit this comparison. Write-All scores 0.640 on Hard-100, showing that more context alone does not resolve the selection problem.

These results place the method on two distinct quality–cost paths. Pure CAMS-C targets settings that already compute dense embeddings and need to score many candidate pools without an online generator. Its Hard-100 gain appears at $k = 8$ and grows numerically through $k = 12$, where it reaches 0.673 versus Dense's 0.651. The +CE path targets higher quality after cross-encoding has been accepted: at $k = 12$, CAMS-C+CE reaches 0.717 versus 0.678 for CE. The comparison does not collapse into one best method across budgets. It identifies when the learned setwise residual is useful by itself and when it complements a stronger passage-level scorer.

4.3 Supervision source

Replacing supporting-label losses with credit yields similar observed F1 in most tested settings (Table 2). Across 22 Qwen pool–budget cells, C minus L ranges from -0.030 to $+0.014$; 21 comparisons have $p > 0.05$. The exception is Natural-10 at $k = 3$: C scores 0.662 and L scores 0.691, with a difference of -0.0298 (95% CI $[-0.0586, -0.0015]$, $p = 0.039$). The study supports credit as a usable alternative target, while statistical equivalence would require a pre-specified margin. Adding credit regression to label supervision yields no significant Qwen F1 gains in the reported tests.

The supporting-label metric also places the three objectives close together. On Hard-100, pooled AUROC is 0.961/0.962/0.964 for C/L/B, and macro AUROC is 0.960/0.959/0.961. On Natural-10, the joint model ranges from 0.921 to 0.946 across seeds. Thus high support discrimination is shared by the credit and label objectives; it cannot be attributed uniquely to coalition supervision. The supervision comparison holds the feature representation and selector capacity fixed. All three objectives retain the dense prior, so the ablation tests target source within a shared retrieval backbone.

Table 2: **Credit feedback supplies a competitive training target across five pool constructions.** Pure selectors use the Qwen reader at $k = 8$; C/L/B denote credit, label, and joint objectives. Learned scores are three-seed mean ± SD, with paired C−L intervals.

Pool	CAMS-C	CAMS-L	CAMS-B	Δ [95% CI]	p
Natural-10	0.717 ± 0.010	0.724 ± 0.005	0.724 ± 0.008	$-0.007\ [-0.031, +0.015]$	0.553
Random-100	0.675 ± 0.003	0.688 ± 0.002	0.674 ± 0.018	$-0.014\ [-0.035, +0.007]$	0.183
Hard-100	0.647 ± 0.004	0.641 ± 0.003	0.652 ± 0.003	$+0.006\ [-0.014, +0.026]$	0.558
Exact-copy	0.751 ± 0.002	0.762 ± 0.006	0.765 ± 0.003	$-0.010\ [-0.028, +0.007]$	0.236
Paraphrase	0.770 ± 0.003	0.775 ± 0.011	0.774 ± 0.005	$-0.005\ [-0.022, +0.013]$	0.600

The two metrics answer different supervision questions. Supporting AUROC asks whether a score separates annotated support from distractors, whereas answer F1 tests the ordered context consumed by the reader. Their agreement on Hard-100 indicates that credit can recover a strong support signal without using that annotation in the loss. Their local disagreement on Natural-10 at $k = 3$ shows why the downstream test remains necessary: two rankers can identify broadly similar supporting passages yet order the first few slots differently. The evidence therefore supports substituting the training target in most observed cells, while leaving the choice of a formal equivalence margin to a study designed for that hypothesis.

The substitution changes where supervision comes from. CAMS-L consumes passage-level support labels; CAMS-C derives its targets from question–answer pairs and frozen-reader evaluations. It therefore removes passage annotation from the training objective, while retaining an answer key and paying for reader calls offline. This setting fits collections where outcomes are logged more readily than evidence spans. The measured result is also narrower than a claim of equivalence: matching architecture and optimization isolates the target source, and the 22 paired cells show the observed range and its single unadjusted exception. Operationally, this separates data requirements from ranking capacity: CAMS-C uses the same frozen-reader evaluations for every pool, while CAMS-L additionally requires passage-level support spans. The close F1 values therefore support a low-label training path without changing the online selector or its inference cost.

4.4 Redundant evidence

Useful credit and diverse selection are separate properties. At $k = 2$ on Exact-copy, pure CAMS-C scores 0.448 F1, close to MMR, and 89.7% of query selections contain at least one repeated source. Deduplication raises its F1 to 0.622 and distinct-support coverage from 0.487 to 0.758. Joint CAMS-B+dedup reaches 0.626; its paired gain over the matched Dense+dedup control is 0.0617 (95% CI $[0.0236, 0.1008]$, $p = 0.0014$). This control separates the learned ranking's contribution from simply granting one method a diversity rule.

The pattern extends to rewritten copies. On Paraphrase at $k = 3$, CAMS-B+dedup obtains 0.714 F1, compared with Dense+dedup's 0.629 and MMR's 0.603; distinct-support coverage reaches 0.916. At $k = 8$, however, deduplication lowers CAMS-B from 0.774 to 0.744 ($p = 0.0106$). A threshold chosen for coverage can postpone evidence that still helps the reader. SetR-prompt remains stronger at $k = 2$ on both redundancy pools (0.719/0.723). Figure 4 and Appendix E therefore identify the tight-budget regime where inexpensive ranking plus explicit diversity is useful.

The budget dependence follows from what the filter changes. At $k = 2$, accepting a second copy can consume half of the available context without adding a new supporting source. Delaying that copy creates space for complementary evidence, which explains the simultaneous increase in distinct-support coverage and answer F1. At $k = 8$, the opportunity cost of a repeated passage is smaller, and lexical similarity may also remove passages that repeat one fact while adding another. Coalition credit resolves the attribution ambiguity in the first case; the geometric filter addresses slot allocation, and its large-budget reversal marks the boundary between those two operations. The aggregate metrics do not trace the reversal to individual deferred passages.

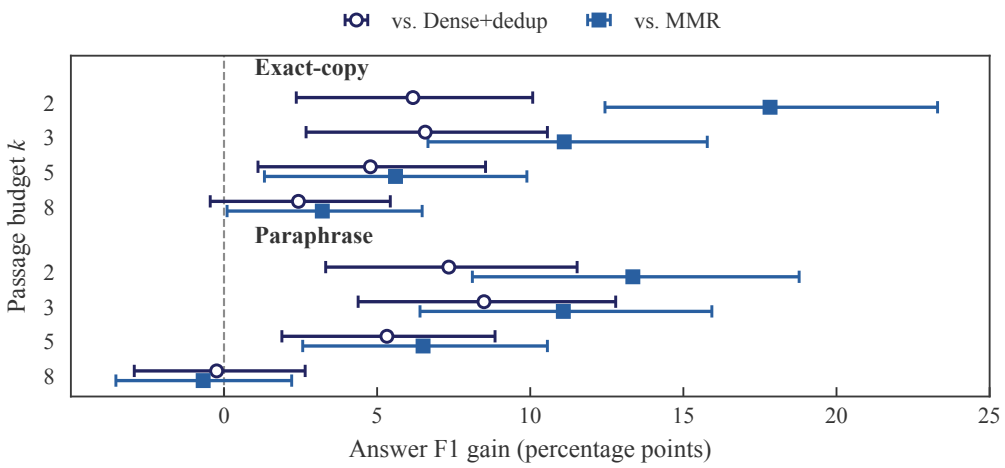

Figure 4: **Explicit diversity converts useful scores into stronger tight-budget sets.** Qwen F1 gains of CAMS-B+dedup over MMR and Dense+dedup use paired 95% bootstrap intervals over 250 questions after averaging three seeds. Rows group exact and paraphrased redundancy across four budgets; gains are in percentage points.

5 Analysis

5.1 Transfer and attribution fidelity

Keeping the Qwen-trained scores tests whether a new reader can use the learned ranking (Table 3). Llama's Exact-copy F1 at $k = 8$ is 0.684 for CAMS-C and 0.655 for Dense; Natural-10 gives 0.623 and 0.627. On Hard-100 at this budget, CAMS-C approximately matches Dense and trails CE. At $k = 16$, credit-only exceeds label-only by 0.0275 with unadjusted $p = 0.046$. These outcomes show that a ranking useful to one reader can have different effects with another. Earlier MuSiQue and 2WikiMultihopQA results (Trivedi et al., 2022; Ho et al., 2020) provide additional cross-dataset context for the unified evaluation.

The learned ranking also retains only part of the teacher's ordering. Original held-out diagnostics give teacher–student Spearman of 0.269 for credit-only and 0.287 for joint supervision, versus 0.251 for Dense; teacher repeat agreement is 0.983. Credit-only identifies the teacher's top two passages better than Dense (AUROC 0.824 versus 0.766), indicating a useful coarse signal alongside weak full-order fidelity. The teacher's stability leaves little support for explaining the low correlation solely by noisy targets.

Even perfect recovery of scalar credit would leave a subset-selection gap. On a separate 494-question enumeration cohort, the size-five subset chosen on two reader runs scores 0.868 F1 on two held-out runs; Banzhaf top-five scores 0.836 and Write-All 0.764. The 0.032 F1 gap measures subset-level headroom beyond ranking passages by mean credit. Subset size is fixed, and both choices are evaluated on held-out reader runs. This oracle comparison uses a different cohort from the main tables. Mean credit can also hide context dependence: among 2,517 near-zero-credit passages, 455 have substantial effects of both signs and 422 are nearly flat. The two categories cover only part of the near-zero group (Appendix E). Thus 18.1% of this group exhibits opposing effects; a small mean alone is insufficient to identify an uninformative passage. The remaining passages fall outside these two report-defined categories. Both signs must be retained to distinguish cancellation from uniformly weak effects.

Transfer and fidelity measure different properties. CAMS predicts Qwen-derived credit from features; Llama answers from the selected passages using a different response function. Exact-copy gains transfer to both readers, while Natural-10 does not improve for Llama. This evidence supports partial transfer of useful selection behavior, alongside weak recovery of the full teacher ranking (Spearman 0.269). Successful selection therefore cannot be read as evidence that all teacher scores have been recovered.

Table 3: **Qwen-trained credit scores retain their clearest gain under exact duplication with a second reader.** Answer F1 at $k = 8$ compares Dense and pure CAMS-C for Qwen and Llama. Learned entries are three-seed mean ± SD; Δ is CAMS-C minus Dense.

	Qwen reader			Llama reader		
Pool	Dense	CAMS-C	Δ	Dense	CAMS-C	Δ
Natural-10	0.684	0.717 ± 0.010	+0.033	0.627	0.623 ± 0.014	-0.005
Exact-copy	0.740	0.751 ± 0.002	+0.012	0.655	0.684 ± 0.005	+0.029
Hard-100	0.608	0.647 ± 0.004	+0.039	0.553	0.554 ± 0.009	+0.001

5.2 Computation and scope

On one H20 with 40 timed Hard-100 questions, the selector forward pass costs 1.20 ms, CE costs 268.61 ms, LLM listwise costs 3.58 s, and SetR-prompt costs 8.35 s at $k = 16$ (Figure 2b). The generative components are approximately 3,000 and 7,000 times the selector forward cost. Embedding the query and pool costs 57.92 ms and the measured feature subroutine costs 6.12 ms; these components together with forward computation total 65.24 ms. Embeddings therefore dominate the measured small-selector path when they are not cached.

The timing scope is component-level. Forward timing uses a fresh instance of the same architecture with synthetic scalar slots; some feature assembly, transfers, sorting, and optional deduplication are outside the measured sum. +CE additionally incurs cross-encoder time. Final-reader timing uses Dense-selected contexts, giving 1.80 s at $k = 8$ versus 4.85 s for the full pool. A complete trained CAMS-to-reader pipeline was not timed. Appendix F gives latency distributions and tokenizer-level input sizes, which determine KV-cache accounting (Ainslie et al., 2023).

Offline feedback trades substantial initial computation for inexpensive reuse. Exhaustive training used 6,144,000 reader calls and 68.8 GPU-hours. The separate width-100 timing study uses 101 calls per question for LOO and 256 for sampled Banzhaf; these are different estimators and budgets. Sampling can reduce coalition evaluation costs (Castro et al., 2009), but cheaper feedback must be assessed through the resulting selector. The current evidence is limited to controlled multi-hop QA pools, generated redundancy, and two reader families. Trained SetR and matched SCARLet/ContextCite selection experiments remain open comparisons; the prompted baseline does not resolve them.

Quality and cost select different operating points. Figure 3 shows stronger scorers leading at small budgets, while Table 1 separates pure CAMS from paid +CE fusion. Offline targets can be reused across future pools; a deployment-specific break-even volume additionally requires workload and hardware measurements. Deployment requires three separate checks: compare scorers at the actual passage budget (Figure 3), test distinct-support coverage under redundancy (Figure 4), and rerun selected contexts through the deployed reader (Table 3). Quality, slot allocation, and reader transfer can each change independently.

6 Conclusion

Deleting a passage from the full context can hide its contribution when another passage supplies the same fact. On Exact-copy, LOO support AUROC is 0.500, while coalition feedback reaches 0.837. Training a 462,849-parameter setwise selector on this feedback supplies an alternative to passage-label losses: paired tests across 22 pool–budget settings find one nominally significant difference, without establishing statistical equivalence. On Hard-100 at $k = 8$, pure credit-trained CAMS raises Dense F1 from 0.608 to 0.647; its forward component takes 1.20 ms with no generative-model call. Embedding, feature, and forward components together cost 65.24 ms. The budget curves, matched deduplication controls, and second-reader tests locate distinct limits: stronger scorers can earn their cost at small budgets, useful substitutes can waste slots, and gains need not transfer across readers. Coalition feedback thus provides reusable supervision for inexpensive selection, with quality, diversity, and deployment cost measured separately. The annotation saving concerns passage-label training losses; question–answer targets and development annotations remain part of the reported protocol.

AI USE STATEMENT

Generative AI tools were used to restructure and edit manuscript text, summarize and locate relevant literature, inspect experimental reports, interpret reported results, check numerical and LaTeX consistency, and create the conceptual artwork in Figure 1. No experimental measurements were generated by AI: Figures 2–4 and all tables are produced from logged experiment reports using deterministic scripts. All AI-assisted claims and artifacts were checked against the cited papers, source reports, and executable validation rules. The authors take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We follow the ICLR Code of Ethics. The study uses publicly available datasets and models under their respective licenses, including Natural Questions-derived multi-hop QA pools, BGE embeddings, Qwen2.5-14B-Instruct, and Llama-3.1-8B-Instruct. It contains no human-subject or animal experiments, personal-identification data, or user-facing deployment; all measurements are offline evaluations on released benchmarks.

REPRODUCIBILITY STATEMENT

The method definition and objectives appear in Section 3. Appendix A specifies training, candidate construction, ordering, baseline, deduplication, and uncertainty conventions. Appendices B–F report complete pool and budget results, oracle diagnostics, and latency distributions. The accompanying package regenerates every revised table and empirical figure from hashed report snapshots and includes the available evaluation and aggregation scripts; Appendix A identifies the training artifacts and per-question shards required for a complete experiment rerun.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, 2023.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pp. 9112–9141, 2024.

John F Banzhaf III. Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev.*, 19: 317, 1964.

Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 335–336, 1998.

Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Computers & operations research*, 36(5):1726–1730, 2009.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Mądry. Contextcite: Attributing model generation to context. *Advances in Neural Information Processing Systems*, 37:95764–95807, 2024.

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021.

Ian Connick Covert, Chanwoo Kim, Su-In Lee, James Zou, and Tatsunori Hashimoto. Stochastic amortization: A unified approach to accelerate feature and data attribution. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Jiale Deng, Yanyan Shen, Ziyuan Pei, Youmin Chen, and Linpeng Huang. Influence guided context selection for effective retrieval-augmented generation. *Advances in Neural Information Processing Systems*, 38:29225–29252, 2026.

Yiming Du, Bingbing Wang, Yang He, Bin Liang, Baojun Wang, Zhongyang Li, Lin Gui, Jeff Z Pan, Ruifeng Xu, and Kam-Fai Wong. Memguide: Intent-driven memory selection for goal-oriented multi-session llm agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(36): 30584–30592, 2026.

James Enouen, Hootan Nakhost, Sayna Ebrahimi, Sercan Arik, Yan Liu, and Tomas Pfister. Textgenshap: Scalable post-hoc explanations in text generation with long documents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13984–14011, 2024.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International conference on machine learning*, pp. 2242–2251. PMLR, 2019.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Taeho Hwang, Sukmin Cho, Soyeong Jeong, Hoyun Song, SeungYoon Han, and Jong C Park. Exit: Context-aware extractive compression for enhancing retrieval-augmented generation. *arXiv preprint arXiv:2412.12559*, 2024.

Neil Jethani, Mukund Sudarshan, Ian Connick Covert, Su-In Lee, and Rajesh Ranganath. Fastshap: Real-time shapley value estimation. In *International conference on learning representations*, 2021.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Llmlingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 13358–13376, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Dahyun Lee, Yongrae Jo, Haeju Park, and Moontae Lee. Shifting from ranking to set selection for retrieval augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17606–17619, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020.

Ruizhe Li, Chen Chen, Yuchen Hu, Yanjun Gao, Xi Wang, and Emine Yilmaz. Attributing response to context: A jensen-shannon divergence driven mechanistic study of context attribution in retrieval-augmented generation. *arXiv preprint arXiv:2505.16415*, 2025.

Fengyuan Liu, Nikhil Kandpal, and Colin Raffel. Attribot: A bag of tricks for efficiently approximating leave-one-out context attribution. In *International Conference on Learning Representations*, volume 2025, pp. 16687–16710, 2025.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

Peiyang Liu, Qiang Yan, Ziqiang Cui, Di Liang, Xi Wang, and Wei Ye. Beyond semantic relevance: Counterfactual risk minimization for robust retrieval-augmented generation. *arXiv preprint arXiv:2605.01302*, 2026.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Multi-Linguality Multi-Functionality Multi-Granularity. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 2024.

Ikhtiyor Nematov, Tarik Kalai, Elizaveta Kuzmenko, Gabriele Fugagnoli, Dimitris Sacharidis, Katja Hose, and Tomer Sagi. Source attribution in retrieval-augmented generation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 317–332. Springer, 2025.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document ranking with a pre-trained sequence-to-sequence model. In *Findings of the association for computational linguistics: EMNLP 2020*, pp. 708–718, 2020.

Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc, 2009.

Lloyd S. Shapley. A value for n-person games. In Harold W. Kuhn and Albert W. Tucker (eds.), *Contributions to the Theory of Games, Volume II*, pp. 307–317. Princeton University Press, Princeton, NJ, 1953.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pp. 31210–31227. PMLR, 2023.

Saksham Sahai Srivastava. Causal intervention-based memory selection for long-horizon llm agents. *arXiv preprint arXiv:2605.17641*, 2026.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is chatgpt good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 14918–14937, 2023.

Robert J Tibshirani and Bradley Efron. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57(1):1–436, 1993.

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

Fangxin Wang, Kay Liu, Sourav Medya, and Philip Yu. Bangs: Game-theoretic node selection for graph self-training. In *International Conference on Learning Representations*, volume 2025, pp. 81692–81719, 2025.

Jiachen T Wang and Ruoxi Jia. Data banzhaf: A robust data valuation framework for machine learning. In *International conference on artificial intelligence and statistics*, pp. 6388–6421. PMLR, 2023.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pp. 641–649, 2024.

Yingtai Xiao, Yuqing Zhu, Sirat Samyoun, Wanrong Zhang, Jiachen T Wang, and Jian Du. Token-shapley: Token level context attribution with shapley value. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3882–3894, 2025.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. Recomp: Improving retrieval-augmented lms with context compression and selective augmentation. In *International Conference on Learning Representations*, volume 2024, pp. 43478–43502, 2024.

Yilong Xu, Jinhua Gao, Xiaoming Yu, Yuanhai Xue, Baolong Bi, Huawei Shen, and Xueqi Cheng. Training a utility-based retriever through shared context attribution for retrieval-augmented language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 629–648, 2025.

Chenghao Yang, Fan Yin, He He, Kai-Wei Chang, Xiaofei Ma, and Bing Xiang. Efficient shapley values estimation by amortization for text classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8666–8680, 2023.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. Making retrieval-augmented language models robust to irrelevant context. *arXiv preprint arXiv:2310.01558*, 2023.

A Protocol and Reproducibility Details

Scope of the accompanying files. The accompanying package regenerates figures and tables from evaluation reports and includes the available evaluation, aggregation, deduplication, supervision-arm, timing, and enumeration scripts. Re-running training and per-question evaluation additionally requires the training implementation, checkpoints, and evaluation shards. The report summaries omit per-seed fusion weights, deduplication thresholds and seed spreads, and fused-method paired tests.

Training inputs and objectives. All three supervision arms use the same width-10 attribution aggregates (balanced, support-focused, and interaction-focused), totaling 1,500 training pools. Training is repeated for each held-out pool and seed, allowing development selection to vary across pools. Inputs have dimension $4 \cdot 384 + 4 = 1540$: four embedding blocks and four scalar features. The two-layer, four-head architecture has 462,849 parameters.

For labels $y_i \in \{0, 1\}$ and $\tilde{y} = y / \sum_j y_j$, the label-only arm uses

$$\begin{aligned}\mathcal{L}_L = & -\sum_i \tilde{y}_i \log \mathrm{softmax}(h)_i + 0.5 \sum_i \mathrm{BCE}_{\mathrm{logit}}(h_i, y_i) \\ & + 0.5\, \mathbb{E}_{y_i=1, y_j=0} \log(1 + e^{-(h_i - h_j)}) + 0.02\, n^{-1} \sum_i r_i^2. \end{aligned} \tag{7}$$

The joint arm adds $0.45\mathcal{L}_{\mathrm{credit}}$. Credit-only training sets the BCE and pairwise weights to zero and changes the listwise target to credit. The objective uses softmax credit targets and within-pool normalized regression targets. The available launchers import the original training implementation, whose full source is outside this package. The original checkpoint rule selects by development support recall; no verified credit-only override is recorded. Consequently, the loss-level annotation claim does not extend to label-free checkpoint selection.

Parameter	Value
Training seeds	101, 202, 303
Attribution reader runs	11, 23, 37, 53
Development/test split seed	20260714
Optimizer	AdamW
Learning rate / weight decay	5×10^{-4} / 3×10^{-4}
Batch size / gradient clipping	32 pools / 2.0
Maximum epochs / patience	300 / 50
Residual scale λ	0.5
Fusion grid α	$0, 0.25, 0.5, 0.75, 1$
Unified QA decoding	Greedy, 128 output tokens
LLM listwise / SetR-prompt output budgets	80 / 384 tokens
Unified maximum model length	32,768 tokens (script default)

The launchers inherit these optimization settings from the original training configuration. Maximum model length is a script setting; prompt truncation also depends on the rendered input.

Candidate pools. Random-100 and Hard-100 retain the original ten candidates and add donor passages from other questions in the evaluation bank. Hard negatives are chosen by BGE similarity to the query. A normalized-text filter rejects exact duplicates. A gold-answer substring guard is applied when the normalized answer has at least four characters and is not yes, no, true, or false. This controlled construction uses answers to reduce donor-label contamination. The substring guard neither detects all answer paraphrases nor ensures that every retained donor is useless.

Exact-copy adds redundant supporting evidence to the stored pools. Paraphrase rewrites injected copies using Qwen2.5-14B-Instruct, instructing it to preserve names, dates, numbers, and facts with different wording. These generated stress cases have no documented manual factual-equivalence audit. Banzhaf attribution for the paraphrase pool uses 128 Bernoulli(0.5) masks and their complements, hence 256 coalition evaluations in one reader run; it is not exhaustive. Sampling is a practical alternative to enumeration (Castro et al., 2009), but introduces coalition-sampling uncertainty.

Order and payoff conventions. The mathematical game uses a fixed pool order restricted to the coalition. The supplied paraphrase and width-100 LOO scripts explicitly preserve stored order. For the original width-10 enumeration, this convention follows the original stored coalition construction; its full generator is outside the supplied package. The unified evaluator ranks by score and then dense score via lexicographic sorting. Residual ties preserve input order. It sends selected text in ranking order, and caches by question identifier plus the ordered index tuple. It does not collapse permutations of the same subset. SetR-prompt's returned index order is preserved. Accordingly, evaluation measures the chosen subset together with its presentation order.

Baseline implementation boundaries. BGE-reranker-v2-m3 is a 560M cross-encoder. Its timing implementation tokenizes each query–passage pair with a 512-token cap. The listwise LLM parser accepts valid indices from the generated output and fills missing indices in Dense order. Its 80-token generation budget can constrain full rankings in width-100 pools, and we do not generalize its measured weakness to more generously budgeted listwise systems. SetR-prompt truncates each passage to 900 characters, requests at most k indices, removes repeated/invalid indices, and falls back to Dense top-k if no valid selection is parsed. It may choose fewer than k passages. MMR scores are read from prepared pools; the provided reporting artifacts do not record the original MMR tuning grid. SCARLet and ContextCite have no reported matched experiments, and the prompted SetR adaptation is not a substitute for evaluating the authors' trained model.

SetR-inspired prompt. The system prompt is "You select a set of passages that together answer a question." After the query and indexed candidates, the user prompt continues:

> First list, briefly, each piece of information needed to answer the question. Then choose at most k passages that together cover all of it, preferring passages that do not repeat each other. On the last line write only 'Selected: ' followed by the chosen indices, comma-separated.

The same evaluation reader executes this prompt and final QA; all methods share the final QA builder.

Deduplication. For each ranker, thresholds $\{0.80, 0.85, 0.90, 0.95, 1.01\}$ are evaluated by the mean distinct-support coverage over development questions and budgets $\{2, 3, 5, 8\}$. A strict improvement replaces the incumbent, so ties favor the first (lower) threshold. The same chosen threshold is then used at all test budgets for that ranker. The threshold 1.01 effectively disables filtering for normalized embeddings. Deferred candidates are appended after all retained candidates. This extension uses development supporting annotations even for a credit-only ranker, and therefore cannot support a completely passage-label-free claim. Selected numerical thresholds are absent from the text reports.

Uncertainty and significance. The report generator uses population standard deviation (NumPy's default, $\mathrm{ddof} = 0$) across the three seed-level means. Question-level bootstrap differences use the average of the seed predictions for each question. Percentile intervals use the 2.5th and 97.5th percentiles; the two-sided empirical p-value is twice the smaller bootstrap tail probability. A printed bootstrap value of 0.0000 reflects finite resampling; three-decimal tables render it as $p < 0.001$. No multiplicity correction or equivalence margin was pre-specified. G2 summaries omit per-seed standard deviations for deduplicated variants; those entries and plots report three-seed means with question-level paired intervals where available, without fabricated seed error bars.

Data provenance. All revised tables and experimental figures use the same report snapshots, with source hashes recorded in the numerical export. Original-session diagnostics are labeled separately. Shared-process evaluation and ordered-context reuse reduce execution mismatches within the unified comparisons; they do not establish invariance of paired differences across reader sessions.

B Complete Unified Qwen Selection Results

All entries use the Qwen reader and 250 test questions per pool. Values for CAMS-C/L/B are three-seed mean $\pm$ population SD. C, L, and B identify credit-only, label-only, and joint supervision. "Pure" has $\alpha = 1$; dense, CE, and LLM identify the fixed fusion base, with α selected on

development separately per budget and seed. Endpoints are included, so a fused row can equal the unmodified base. Printed means are rounded to three decimals; the source reports retain four decimals.

Pool	Dense	CAMS-C	CAMS-L	CAMS-B
Natural-10	0.684	0.717 ± 0.010	0.724 ± 0.005	0.724 ± 0.008
Random-100	0.648	0.675 ± 0.003	0.688 ± 0.002	0.674 ± 0.018
Hard-100	0.608	0.647 ± 0.004	0.641 ± 0.003	0.652 ± 0.003
Exact-copy	0.740	0.751 ± 0.002	0.762 ± 0.006	0.765 ± 0.003
Paraphrase	0.751	0.770 ± 0.003	0.775 ± 0.011	0.774 ± 0.005

Each row fixes the largest common budget (or the largest available budget) and reports the three supervision arms under the same reader session.

How to read the cross-pool summary. The five rows hold the reader, scoring features, and training seeds fixed while changing only the candidate-pool construction. Natural-10 contains ten retrieved passages; Random-100, Hard-100, Exact-copy, and Paraphrase contain one hundred candidates generated by the corresponding pool procedure. The Dense column is the fixed retrieval order used as the reference for every row. CAMS-C uses interaction credit targets, CAMS-L uses passage labels, and CAMS-B concatenates the two supervision signals. The row-wise comparison therefore isolates the supervision choice from the pool choice at the common operating point. The compact table is the cross-pool view of the budget sweeps used to select the development configuration; no score from a test question enters that selection.

Budget and seed convention. For each pool, the reported point is the largest budget shared by the unified Qwen runs (or the largest available budget when a pool has a shorter scan). Every CAMS entry averages three independently trained selectors. The displayed uncertainty is the population standard deviation of those three seed means; paired question-bootstrap intervals are used for the hypothesis tests in the main text. This distinction keeps operating-point comparisons and uncertainty estimates on their respective, reproducible scales.

C Complete Unified Llama Selection Results

This compact view reports the common $k = 8$ operating point for the three Llama evaluation pools. The trained selector scores are retained from Qwen supervision, while the reader and development selection are changed for this evaluation; the full budget scans remain in the accompanying package.

Pool	Dense	CAMS-C	CAMS-L	CAMS-B
Natural-10	0.627	0.623 ± 0.014	0.637 ± 0.007	0.638 ± 0.009
Exact-copy	0.655	0.684 ± 0.005	0.687 ± 0.006	0.702 ± 0.021
Hard-100	0.553	0.554 ± 0.009	0.570 ± 0.011	0.569 ± 0.005

The same trained selectors are evaluated with the second reader; values use the common budget shown in the table.

The second reader preserves the same broad ordering across these pools: all three trained selectors exceed Dense on Exact-copy, and the label-supervised and joint-supervised selectors exceed Dense on Natural-10 and Hard-100. The largest joint-supervision gain is on Exact-copy, where CAMS-B reaches 0.702 compared with 0.655 for Dense; on Hard-100 it reaches 0.569 compared with 0.553. Natural-10 rises from 0.627 to 0.638. These comparisons use the same trained selector weights, so they show that the learned passage ordering remains useful when the answer reader changes.

The table separates three training signals under one setwise architecture. CAMS-C is trained with coalition credit, CAMS-L with supporting-passage labels, and CAMS-B with both signals. Their shared candidate representation and inference rule make the columns directly comparable: each score summarizes selection quality for the same pool and reader, while the training objective varies. The seed variation is reported after each learned mean, so the table also shows how consistently each training signal transfers across independently trained selectors.

The shared $k = 8$ point is the largest common operating budget in these Llama sessions. Holding the budget fixed keeps the reader-transfer comparison at a common context size, while the three pool rows show behavior on retrieved, duplicated, and hard-negative candidates. The full scans use the same question identifiers, candidate order, and score normalization as the unified evaluation and are retained with the report snapshots for regeneration.

D SUPERVISION TESTS AND SUPPORTING-PASSAGE AUROC

Pool	k	Credit minus label F1	95% paired CI	p
Natural-10	2	-0.015	$[-0.042, +0.011]$	0.258
Natural-10	3	-0.030	$[-0.059, -0.002]$	0.039
Natural-10	5	+0.004	$[-0.017, +0.025]$	0.709
Natural-10	8	-0.007	$[-0.031, +0.015]$	0.553
Random-100	3	-0.006	$[-0.027, +0.014]$	0.536
Random-100	5	-0.007	$[-0.030, +0.016]$	0.578
Random-100	8	-0.014	$[-0.035, +0.007]$	0.183
Random-100	12	-0.009	$[-0.030, +0.013]$	0.436
Random-100	16	+0.009	$[-0.009, +0.026]$	0.319
Hard-100	3	+0.014	$[-0.009, +0.036]$	0.240
Hard-100	5	-0.001	$[-0.021, +0.018]$	0.958
Hard-100	8	+0.006	$[-0.014, +0.026]$	0.558
Hard-100	12	+0.007	$[-0.013, +0.026]$	0.512
Hard-100	16	-0.001	$[-0.022, +0.019]$	0.921
Exact-copy	2	+0.004	$[-0.021, +0.031]$	0.743
Exact-copy	3	-0.007	$[-0.031, +0.017]$	0.592
Exact-copy	5	-0.003	$[-0.023, +0.017]$	0.755
Exact-copy	8	-0.010	$[-0.028, +0.007]$	0.236
Paraphrase	2	-0.007	$[-0.041, +0.026]$	0.681
Paraphrase	3	-0.007	$[-0.032, +0.017]$	0.570
Paraphrase	5	-0.010	$[-0.028, +0.010]$	0.327
Paraphrase	8	-0.005	$[-0.022, +0.013]$	0.600

The paired rows quantify how closely credit supervision tracks passage-label supervision across pools and budgets. The source reports retain the joint comparisons and the corresponding Llama checks.

Pool	Signal	Pooled AUROC	Macro AUROC
Natural-10	CAMS-C	0.933 ± 0.003	0.926 ± 0.002
Natural-10	CAMS-L	0.944 ± 0.001	0.933 ± 0.003
Natural-10	CAMS-B	0.937 ± 0.011	0.927 ± 0.010
Natural-10	Dense	0.862	0.887
Random-100	CAMS-C	0.990 ± 0.000	0.988 ± 0.000
Random-100	CAMS-L	0.989 ± 0.000	0.988 ± 0.000
Random-100	CAMS-B	0.989 ± 0.000	0.988 ± 0.000
Random-100	Dense	0.986	0.987
Hard-100	CAMS-C	0.961 ± 0.001	0.960 ± 0.002
Hard-100	CAMS-L	0.962 ± 0.000	0.959 ± 0.001
Hard-100	CAMS-B	0.964 ± 0.001	0.961 ± 0.000
Hard-100	Dense	0.947	0.946
Exact-copy	CAMS-C	0.932 ± 0.006	0.920 ± 0.006
Exact-copy	CAMS-L	0.942 ± 0.002	0.928 ± 0.002
Exact-copy	CAMS-B	0.941 ± 0.002	0.928 ± 0.003
Exact-copy	Dense	0.868	0.891
Paraphrase	CAMS-C	0.931 ± 0.003	0.919 ± 0.003
Paraphrase	CAMS-L	0.941 ± 0.001	0.929 ± 0.002
Paraphrase	CAMS-B	0.940 ± 0.001	0.930 ± 0.001
Paraphrase	Dense	0.851	0.875

The pooled and macro columns use the same candidate-level and query-level definitions as the main text; seed variation is shown for learned selectors.

Pooled AUROC pools all candidate passages; macro AUROC averages per-query AUROC over valid queries. The table places the three supervision arms and Dense reference on one common scale across pools; exact arrays and width-specific diagnostics remain in the evidence files.

E Redundancy Diagnostics and Oracle References

These tables report F1, duplicate-query rate, and distinct-support coverage; together they show how selection quality and evidence coverage change under duplicated passages. Duplicate-query rate is $\mathbb{E}_q[\mathbf{1}\{|\operatorname{source}(S_q)| < |S_q|\}]$. Distinct-support coverage is the number of different supporting source passages selected divided by the number available, so copies do not increase coverage. CAMS entries are three-seed means; the report does not provide their seed SDs after deduplication. Teacher and LOO rows use test answers and are **oracle diagnostics only**. Their role is to diagnose the relation between attribution and selection. Scalar oracle rankings are not guaranteed upper bounds.

E.1 Exact-copy: F1 / duplicate-query rate / distinct-support coverage

Method	$k = 2$	$k = 3$	$k = 5$	$k = 8$
Dense	0.437 / 0.872 / 0.478	0.583 / 0.956 / 0.742	0.681 / 0.988 / 0.872	0.740 / 0.996 / 0.964
Dense+dedup	0.564 / 0.000 / 0.722	0.628 / 0.000 / 0.814	0.690 / 0.000 / 0.904	0.738 / 0.000 / 0.966
MMR	0.448 / 0.836 / 0.488	0.583 / 0.956 / 0.740	0.682 / 0.984 / 0.872	0.730 / 0.996 / 0.964
LLM	0.613 / 0.300 / 0.712	0.727 / 0.660 / 0.884	0.759 / 0.892 / 0.960	0.770 / 0.968 / 0.986
SetR-prompt	0.719 / 0.104 / 0.852	0.740 / 0.140 / 0.878	0.749 / 0.248 / 0.900	0.757 / 0.197 / 0.908
LOO	0.435 / 0.768 / 0.458	0.555 / 0.876 / 0.702	0.682 / 0.944 / 0.858	0.727 / 1.000 / 0.950
Teacher	0.665 / 0.784 / 0.498	0.796 / 0.896 / 0.752	0.819 / 0.936 / 0.854	0.814 / 0.976 / 0.930
Teacher+dedup	0.783 / 0.000 / 0.738	0.800 / 0.000 / 0.810	0.790 / 0.000 / 0.872	0.799 / 0.000 / 0.944
CAMS-B	0.440 / 0.892 / 0.485	0.633 / 0.969 / 0.775	0.734 / 0.996 / 0.935	0.765 / 0.999 / 0.987
CAMS-B+dedup	0.626 / 0.000 / 0.763	0.694 / 0.000 / 0.877	0.738 / 0.000 / 0.957	0.762 / 0.000 / 0.993
CAMS-C	0.448 / 0.897 / 0.487	0.627 / 0.975 / 0.766	0.731 / 0.996 / 0.923	0.751 / 0.999 / 0.980
CAMS-C+dedup	0.622 / 0.000 / 0.758	0.679 / 0.000 / 0.861	0.729 / 0.000 / 0.950	0.754 / 0.000 / 0.991
CAMS-L	0.443 / 0.873 / 0.483	0.634 / 0.967 / 0.787	0.734 / 0.996 / 0.934	0.762 / 0.999 / 0.985
CAMS-L+dedup	0.627 / 0.000 / 0.774	0.685 / 0.000 / 0.887	0.736 / 0.000 / 0.957	0.773 / 0.000 / 0.993

E.2 Paraphrase: F1 / duplicate-query rate / distinct-support coverage

Method	$k = 2$	$k = 3$	$k = 5$	$k = 8$
Dense	0.502 / 0.684 / 0.548	0.603 / 0.876 / 0.772	0.678 / 0.972 / 0.888	0.751 / 0.996 / 0.966
Dense+dedup	0.570 / 0.000 / 0.744	0.629 / 0.008 / 0.824	0.692 / 0.016 / 0.918	0.746 / 0.016 / 0.972
MMR	0.510 / 0.668 / 0.554	0.603 / 0.872 / 0.770	0.680 / 0.968 / 0.888	0.750 / 0.996 / 0.966
LLM	0.567 / 0.600 / 0.610	0.691 / 0.820 / 0.856	0.757 / 0.956 / 0.958	0.761 / 0.996 / 0.982
SetR-prompt	0.723 / 0.180 / 0.818	0.728 / 0.340 / 0.868	0.735 / 0.468 / 0.902	0.763 / 0.470 / 0.916
LOO	0.496 / 0.608 / 0.530	0.610 / 0.800 / 0.754	0.697 / 0.880 / 0.878	0.753 / 0.972 / 0.964
Teacher	0.726 / 0.684 / 0.556	0.803 / 0.864 / 0.748	0.813 / 0.936 / 0.852	0.812 / 0.976 / 0.936
Teacher+dedup	0.777 / 0.040 / 0.714	0.805 / 0.060 / 0.798	0.811 / 0.068 / 0.870	0.814 / 0.072 / 0.950
CAMS-B	0.507 / 0.711 / 0.565	0.675 / 0.908 / 0.837	0.747 / 0.987 / 0.959	0.774 / 0.999 / 0.995
CAMS-B+dedup	0.644 / 0.000 / 0.803	0.714 / 0.004 / 0.916	0.745 / 0.016 / 0.965	0.744 / 0.016 / 0.983
CAMS-C	0.508 / 0.716 / 0.567	0.663 / 0.916 / 0.815	0.744 / 0.992 / 0.951	0.770 / 0.997 / 0.989
CAMS-C+dedup	0.631 / 0.000 / 0.787	0.697 / 0.004 / 0.894	0.728 / 0.016 / 0.957	0.751 / 0.016 / 0.983
CAMS-L	0.516 / 0.693 / 0.573	0.670 / 0.912 / 0.843	0.754 / 0.985 / 0.957	0.775 / 0.995 / 0.995
CAMS-L+dedup	0.634 / 0.000 / 0.813	0.706 / 0.004 / 0.912	0.746 / 0.016 / 0.967	0.742 / 0.016 / 0.987

Oracle selection in the unified Qwen sessions.

Pool	Answer-aware score	$k = 3$	$k = 5$	$k = 8$
Natural-10	LOO	0.731	0.723	0.749
Natural-10	Teacher	0.791	0.789	0.770
Random-100	LOO	0.592	0.634	0.649
Hard-100	LOO	0.576	0.607	0.643
Exact-copy	LOO	0.555	0.682	0.727
Exact-copy	Teacher	0.796	0.819	0.814
Paraphrase	LOO	0.610	0.697	0.753
Paraphrase	Teacher	0.803	0.813	0.812

These values share the corresponding pool's unified evaluation session. Natural-10 teacher credit is based on exhaustive coalitions; the redundant-pool teacher uses sampled coalition attribution. Width-100 LOO is measured at the actual evaluation width. A high LOO F1 on Natural-10 is compatible with its low support AUROC: a support annotation and a reader's answer-quality intervention are different targets.

Exhaustive subset headroom (E9). A separate 494-question, width-10 cohort has all 1,024 coalitions available under reader runs 11, 23, 37, and 53. Subset choice and oracle scores use the mean over runs 11 and 23; reported held-out values use runs 37 and 53. The enumeration cohort is separate from the 250-query unified test split; its absolute F1 is therefore not directly comparable with Table 1.

	LOO	Banzhaf	Held-out best subset		In-sample maximum
k	top-k	top-k	$\lvert S\rvert = k$	$\lvert S\rvert \le k$	$\lvert S\rvert = k$
2	0.710	0.793	0.830	0.843	0.844
3	0.746	0.815	0.863	0.869	0.881
5	0.756	0.836	0.868	0.881	0.890
8	0.779	0.801	0.851	0.883	0.860

Write-All obtains 0.7639 F1 and the annotated supporting set 0.7851. The in-sample maximum is optimistic under stochastic payoff measurement; held-out evaluation reduces that selection optimism but does not turn the chosen subset into a strict bound on future runs. Even Banzhaf top-k remains below the held-out best-subset reference, illustrating the distinction between additive attribution and combinatorial selection.

Near-zero credit does not mean uniformly irrelevant (E10). For each of 4,940 passages in the enumeration cohort, compute all 512 marginal effects from the four-run mean coalition table. A passage is near-zero when $|c_i| < 0.02$. It is "flat" when also its marginal-effect SD is below 0.05; it is "cancelling" when more than 10% of marginal effects exceed 0.05 and more than 10% are below -0.05. There are 2,517 near-zero passages, including 422 flat and 455 cancelling. These two categories are not an exhaustive partition of near-zero passages. Supporting-label rates are 0.152 for flat and 0.040 for cancelling, versus 0.200 overall. Supporting passages have mean marginal-effect SD 0.304, compared with 0.218 for other passages. Average negative-marginal shares are 0.040 and 0.086 respectively. The analysis establishes heterogeneity of marginal behavior. A flat measured payoff alone cannot establish semantic irrelevance.

F Runtime, Offline Costs, and Token Accounting

The new timing study uses NVIDIA H20 hardware, one request at a time, synchronized GPU timers, two warm-up questions, 40 timed questions, and no prefix cache. Statistics below are milliseconds; p90 follows the sorted-sample index in the timing script.

Component	Mean (ms)	Median (ms)	p90 (ms)
Embedding (query + pool)	57.92	57.63	61.63
Scalar-feature subroutine	6.12	5.74	8.97
Selector GPU forward	1.20	1.16	1.40
Selector CPU forward	1.65	1.63	1.77
Cross-encoder	268.61	261.77	346.89
LLM listwise	3575.47	3411.50	4674.37
SetR-prompt ($k = 16$)	8353.53	7773.92	12264.54

The selector timing instantiates the actual architecture with fresh weights and zero scalar features. The measured scope is GPU/CPU forward computation. The feature timer covers dense similarity, BM25, and surface-feature computation; the zero scalar slots and array construction lie outside the forward timer. Host/device transfer, final sort, token-overlap assembly, and optional deduplication are not all instrumented in the reported 65.24 ms sum. The CPU forward pass averages 1.65 ms. Parameter count is measured as 462,849.

Final reader and context length. The timed reader consumes Dense-selected contexts at each budget. Measured mean input-token counts include the rendered prompt: 567.3, 809.6, 1,196.4, 1,701.4, and 2,184.2 for $k = 3, 5, 8, 12, 16$, versus 12,850.3 for the full pool. Mean reader times are respectively 1.60, 1.78, 1.80, 1.90, and 1.99 seconds, versus 4.85 seconds. The $k = 8$ prompt-token reduction is 90.7% in *this Dense timing workload*; it is not a measured CAMS reduction. Combining the CAMS and Dense-reader component times would only estimate end-to-end latency. The timing protocol uses one request at a time, two warm-up questions, and forty timed questions with synchronized GPU timers. The selector forward pass is measured after feature construction and before ranking; the feature timer covers dense similarity, BM25, and surface features. The reader measurements use the same tokenizer, prompt template, and greedy decoding budget across context sizes. Thus the component table separates the cost of constructing a score from the cost of generating an answer, while the token counts show why shorter selected contexts reduce reader work. The online selector is reused across questions after offline training, so its measured cost is paid per request after offline training; candidate-coalition calls occur during target construction.

KV accounting. For L decoder layers, H_{KV} key/value heads, head size d_h, and β bytes per element,

$$\mathrm{KV}(T) = 2LH_{\mathrm{KV}}d_h\beta T. \tag{8}$$

This idealized cache size scales with tokenizer-level context length T. It also excludes cache allocation overhead and other model memory. The main paper makes no exact CAMS cache-saving claim from the original word-count tables.

Offline cost and amortization. Four timed width-100 questions give LOO times of 303.68, 428.89, 334.93, and 658.73 seconds (101 reader calls each), and sampled Banzhaf times of 380.87, 506.68, 378.98, and 401.42 seconds (256 calls each). These batched offline figures are not equal-budget estimator comparisons and are not the cost of exhaustive width-10 training. Exhaustive training accounting is $1500 \times 1024 \times 4 = 6{,}144{,}000$ calls and 68.8 GPU-hours. Reader changes may motivate rebuilding targets, although the Llama experiment shows that some selection behavior can transfer without rebuilding them. Offline cost is amortized over future requests; no break-even request count is asserted because GPU-hours, query-time wall clock, hardware occupancy, and deployment workloads differ.

Original coalition-budget diagnostic. Subsampling 128 of the 1,024 enumerated coalitions gave 0.854 supporting AUROC versus 0.887 for full enumeration in the original training-pool analysis (500 questions, five resampling repeats). This motivates testing the downstream effect of cheaper targets. This diagnostic uses the original training-pool cohort, separate from the unified held-out supervision comparison. For a fresh reproduction, the deterministic order is: generate the frozen candidate pools, cache the reader outputs for the prescribed coalition masks, aggregate the payoff values into credits, fit each supervision arm with the shared architecture, select the development configuration, and then evaluate the held-out questions. The report snapshots record the pool identifiers, reader seeds, supervision labels, and budget grids used by the generated tables. The timing script uses the same candidate representation as the selector evaluation and records mean, median, and

p90 values for every component. Together these records make the numerical path from a pool-level report to a table cell explicit, while keeping offline target construction separate from online selection measurement.

The appendix tables use a consistent convention: means are rounded to three decimals, uncertainty is the population standard deviation across the three training seeds, and paired intervals are question-bootstrap intervals. Pool names identify the candidate construction, and k identifies the retained passage budget. This convention lets the compact cross-pool tables be read alongside the main tables without converting units or uncertainty definitions.