KV THICKETS: TASK EXPERTS FROM RANDOM PREFIX-CACHE PERTURBATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural Thickets reveal task-improving experts through random weight perturbations, but each candidate defines a distinct parameter state. In-context learning provides a natural starting point: a prefix steers a frozen model, and its key–value (KV) states permit continuous changes without editing discrete tokens. Can modest populations of small random prefix-KV perturbations also contain useful task experts? We introduce KV Thickets, which independently sample perturbations from prespecified distributions, select candidates on labeled examples, and reuse the selected perturbations across inputs while keeping both model weights and prefix text unchanged. Our vLLM implementation batches their evaluation under shared weights. Across seven models and seven benchmarks, 32 candidates yield a selection-split improver in 45 of 49 settings; at 256 candidates, coverage reaches all 49, with a mean improving fraction of 28.7%. On examples disjoint from selection, the top selection-ranked candidate and an eight-candidate plurality vote improve mean task score by 2.70 and 4.83 points, respectively, over the unperturbed greedy baseline. These comparisons are descriptive because the evaluation includes previously examined benchmark pools. Weight-space experts provide a complementary reference: their induced cache changes have mixed direct effects, but their blockwise magnitudes define another useful random-search profile. Together, these findings identify fixed-prefix KV states as a search space for reusable task adaptation under a single frozen model.

1 INTRODUCTION

Random perturbations around pretrained weights can expose task-improving candidates, as demonstrated by Neural Thickets (Gan & Isola, 2026). This suggests that useful adaptations need not always be reached through iterative optimization. Each weight-space candidate nevertheless defines a distinct parameter state, which must be applied when evaluating that candidate. This raises a question beyond how to search parameter space more efficiently: *can a population of useful experts be accessed without changing the model weights at all?*

In-context learning provides a natural starting point: instructions and demonstrations alter a frozen model's predictions without updating its parameters (Brown et al., 2020). Its interpretation as implicit adaptation motivates examining the internal states through which a prefix influences subsequent computation (Dai et al., 2023). Directly perturbing prompt tokens makes small continuous changes difficult to define (Shin et al., 2020; Prasad et al., 2023), whereas their layer-wise key–value (KV) states provide a continuous representation of the prefix's influence on subsequent generation (Vaswani et al., 2017). Moving from token edits to prefix-KV fields therefore makes small, continuously scaled perturbations possible while leaving the textual prefix fixed.

Continuous context adaptation is already effective. Prefix tuning optimizes task-specific continuous states, Black-Box Tuning searches continuous prompts without gradients, and cache steering constructs directed KV interventions (Li & Liang, 2021; Sun et al., 2022; Belitsky et al., 2025). We consider a complementary route: sampling candidate interventions independently from a fixed distribution. *Can a modest population of small prefix-KV perturbations contain useful task experts?*

We introduce **KV Thickets** to investigate this question. We retain the sampling, task-based selection, and voting procedure of Neural Thickets, but apply it to prefix-local fields within a single frozen model. Each candidate perturbs the prefix's keys and values at magnitudes normalized to the corresponding

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.

clean-cache root-mean-square (RMS), using prespecified scales. Task scores determine which candidates are retained but do not update the sampling distribution. Each selected field is reused unchanged across examples. This construction requires neither gradients nor a source weight expert.

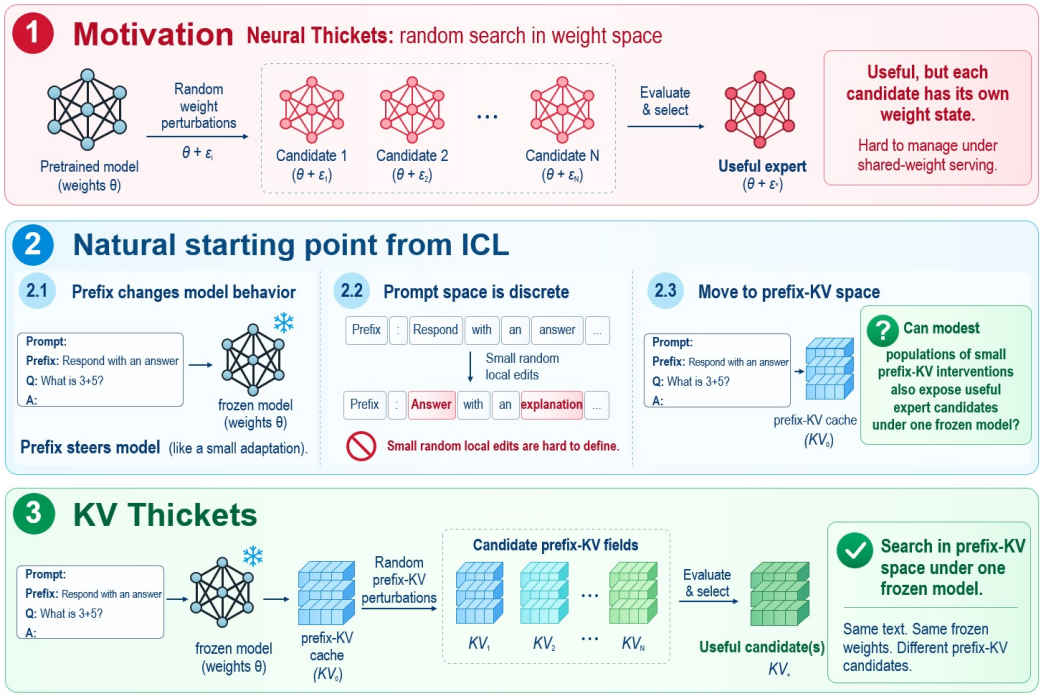


Figure 1: **From weight-space experts to prefix-KV candidate search.** Weight perturbations give each candidate its own parameter state. With a fixed textual prefix and frozen model, KV Thickets samples and selects continuous prefix-KV perturbation fields instead of discrete token edits.

Weight-space experts offer a concrete reference for examining this population view. Changing a model's weights also changes the cache produced by the fixed prefix, providing both an induced field and a profile of how strongly different key and value blocks change. We first evaluate the measured field under the base weights, then retain its blockwise magnitudes and sample fresh directions for task-based selection. The direct effects are mixed, while search at these magnitudes yields useful selected candidates in many settings. A weight-induced cache change can thus provide a reference for exploration without prescribing the directions of the discovered KV candidates.

The prefix-local representation also addresses the parameter-state cost motivating the search. Our vLLM extension batches candidates as requests under one resident model, with each request carrying its own perturbation field and runtime KV state. Switching candidates therefore changes the field rather than the shared weights, providing a common offline execution path to evaluate populations and reuse selected members.

Experiments across seven models and seven benchmarks support this search-space perspective. With 32 candidates, KV Thickets finds an observed selection-split improver in 45 of 49 settings; with 256, coverage reaches all 49 and the mean improving fraction is 28.7%. An eight-candidate vote selected from each 256-candidate bank improves 41 settings and gains 4.83 task-score points on average over the unperturbed greedy baseline on selection-disjoint evaluation sets that include previously examined benchmark pools. In the separate weight-derived study, a single candidate selected from 128 KV fields improves 33 settings and gains 3.48 points on average over that study's unperturbed baseline on earlier registered evaluation items. Together, these results provide empirical evidence for useful prefix-KV populations under both clean-cache-normalized and weight-derived magnitude profiles.

Our main contributions are:

- **KV Thickets.** We introduce random candidate search over the continuous KV states of a fixed textual prefix. Independently sampled fields are selected by task score and reused across examples under unchanged model weights.
- **Empirical characterization of the KV search space.** Across 49 model–task settings, we distinguish selection-split candidate accessibility from selected-predictor performance. The weight-derived study further distinguishes the effects of a particular induced field from the usefulness of searching fresh directions at its magnitudes.
- **Shared-weight population evaluation.** We implement offline batching of candidate-specific prefix-KV fields in vLLM under one resident model and quantify their dense perturbation storage separately from shared weights and runtime caches.

2 Related Work

Population search in parameter space. Neural Thickets samples Gaussian weight perturbations and finds recurring task-improving candidates through selection or voting (Gan & Isola, 2026). Parameter Ensembling by Perturbation instead averages predictions from perturbed weight states and reports improvements in ensemble likelihood and calibration (Mehrtash et al., 2020). We test whether task-improving candidates also recur among independent perturbations of a fixed textual prefix's realized KV cache. A separate analysis asks whether a selected weight candidate's cache-difference magnitudes can set the scale of another KV population.

Prompt and activation interventions. Discrete prompt search (Shin et al., 2020; Prasad et al., 2023), prompt tuning (Lester et al., 2021), and prefix tuning (Li & Liang, 2021) steer frozen models through optimized input-side variables. Black-Box Tuning optimizes continuous prompts by derivative-free population search in a random low-dimensional subspace (Sun et al., 2022). Activation engineering and inference-time intervention add directions to internal states (Turner et al., 2023; Li et al., 2023); function vectors and in-context vectors extract task information from demonstrations and replay it in latent space (Todd et al., 2024; Liu et al., 2024). These methods establish task-directed context interventions and gradient-free continuous-prompt optimization. Our source-free setting instead holds the textual prefix fixed, scores independent perturbations of its realized KV cache, and selects candidates without updating the sampling distribution.

Direct KV-cache interventions. The closest methods also modify attention memory rather than model weights. KV Cache Steering derives one-shot K/V shifts from teacher or annotated reasoning traces (Belitsky et al., 2025); differentiable cache augmentation trains a coprocessor to append latent KV states (Liu et al., 2025); Memory Inception inserts text-derived KV banks at selected layers (Liu et al., 2026); and CachePrune uses learned attribution to suppress instruction-following features in the cache (Wang et al., 2026). These methods construct or learn cache interventions, whereas our source-free setting evaluates independently sampled random fields without adapting the sampling distribution. RippleKV is closer in mechanism: it injects norm-adaptive value-cache noise, but uses the resulting output divergence only to allocate a compression budget across layers (Xu et al., 2026). Our fixed-scale perturbations instead form the candidates themselves and jointly alter prefix keys and values. A second route uses a selected weight candidate to set blockwise magnitudes before sampling fresh KV coordinates.

Multi-expert model serving. PagedAttention enables efficient request-local cache management (Kwon et al., 2023), while S-LoRA shows that heterogeneous low-rank adapters can share base-model computation in one serving system (Sheng et al., 2024). Our vLLM extension evaluates candidate-specific KV perturbations in one offline generation batch while keeping a single model resident; each request's candidate slot follows it through scheduler reordering to the attention-time gather.

3 Methodology

KV Thickets searches for reusable task candidates by independently sampling prefix-KV perturbations and retaining those with the highest task scores. It follows the sampling, selection, and voting procedure of Neural Thickets (Gan & Isola, 2026), but represents each candidate by a prefix-local

field under unchanged model weights. We first describe source-free search and then examine weight-induced fields and magnitude profiles.

3.1 From in-context adaptation to KV search

The implicit-adaptation view of in-context learning motivates treating a prefix's internal states as a space for adaptation (Dai et al., 2023). Let θ denote the pretrained weights and p^* a task prefix of L tokens, fixed before candidate search without task-score optimization. A candidate field Δ is added to the keys and values computed for prefix positions at each layer, after key positional encoding and before attention and cache update. Only prefix positions receive direct additions. Earlier-layer interventions propagate through subsequent computation, so the injected field need not equal the final cache displacement. We denote the resulting candidate by $T(\Delta)$ and the unperturbed baseline by $P_0 = T(0)$.

Write $f_\theta(\Delta; x)$ for the next-token logits at a fixed task input and continuation history x, suppressing the fixed prefix. At a differentiable baseline, weight and KV perturbations have first-order responses

$$\begin{aligned} f_{\theta+\delta\theta}(0; x) - f_\theta(0; x) &\approx J_\theta(x)\,\delta\theta, \\ f_\theta(\Delta; x) - f_\theta(0; x) &\approx J_{\mathrm{KV}}(x)\,\Delta. \end{aligned} \tag{1}$$

The Jacobians are with respect to the weights and injected field, respectively, evaluated at $(\theta, \Delta = 0)$, and include downstream propagation. Equation (1) describes two local routes to changing predictions. KV Thickets searches through the second route while keeping the weights and prefix text fixed, sampling and scoring fields directly without computing Jacobians.

3.2 Sampling a source-free population

Let $C_\theta(p^*)$ be the clean prefix cache, and let C_b denote one layer's key or value block, including all prefix positions, KV heads, and feature entries. For a tensor X, define $\mathrm{RMS}(X) = \|X\|_F/\sqrt{|X|}$, where $|X|$ is its number of entries. At relative scale $\alpha > 0$, independently sample tensors Z_b of the same shapes as C_b, with standard Gaussian entries, and set

$$\Delta_b = \alpha\ \mathrm{RMS}(C_b)\frac{Z_b}{\mathrm{RMS}(Z_b)}. \tag{2}$$

Each block has an independent random direction and a prescribed RMS magnitude. A candidate uses the same relative scale α across blocks, while their absolute magnitudes follow the clean-cache profile. Pooling independent draws at prespecified scales gives a population $\{\Delta^{(i)}\}_{i=1}^N$. Every field is sampled once and reused unchanged across selection and subsequent inputs. The sampling distribution is fixed before candidate scoring. This *source-free* construction requires no weight-space expert; candidate selection uses labeled task examples.

3.3 Candidate selection and population characterization

Let D_{sel} be a labeled selection set and $S_{\mathrm{sel}}(\Delta)$ the task score of $T(\Delta)$ on this set. We rank the full population by selection score, jointly across scales, and retain the top K candidates, where $1 \le K \le N$. Their fields and identities remain fixed for subsequent inputs. The candidates and P_0 share the weights, prefix text, decoding rule, and task evaluator. For prediction, the retained candidates' normalized, nonempty answers enter a plurality vote; $K = 1$ gives a single-candidate predictor. Task-specific ranking and answer-handling conventions are discussed in appendices A and B.

The same scores characterize the sampled population separately from the retained predictor. Define the *selection-split improving fraction*, also called *selection-split expert density*, by

$$\widehat{q} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left\{S_{\mathrm{sel}}\left(\Delta^{(i)}\right) > S_{\mathrm{sel}}(0)\right\}. \tag{3}$$

This fraction uses all N sampled candidates and does not depend on K. *Coverage* counts model–task settings containing at least one candidate that improves the selection score over P_0. Both quantities characterize observed candidate accessibility on D_{sel}; the performance of the retained predictor is assessed separately on examples disjoint from D_{sel}.

3.4 FROM WEIGHT-INDUCED CHANGES TO KV EXPLORATION

The source-free construction sets perturbation magnitudes from the clean prefix cache. A weight-space expert provides another reference through the cache change it induces. We use this change in two ways: as a particular intervention and as a magnitude profile for sampling new fields.

Let W be a candidate selected by Neural Thickets, with weights $\theta_W = \theta + \delta\theta_W$. Using the same prefix tokens and cache layout, define its cache difference Δ^W. Let $J_C(p^*)$ denote the derivative of the clean prefix cache with respect to the weights, evaluated at θ. The measured change and its subsequent injection have the local interpretation

$$\begin{aligned} \Delta^W &:= C_{\theta_W}(p^*) - C_\theta(p^*) \approx J_C(p^*)\,\delta\theta_W, \\ f_\theta(\Delta^W; x) - f_\theta(0; x) &\approx J_{\mathrm{KV}}(x)\,\Delta^W. \end{aligned} \tag{4}$$

We compute Δ^W by evaluating the two caches directly. Injecting this field into the base model defines $T_{\mathrm{img}} = T(\Delta^W)$, called *Image*. It uses the layer-wise intervention of Section 3.1: the source weights construct the field, while all Image predictions use θ.

Next, retain only the blockwise RMS magnitudes of Δ^W and sample fresh independent Gaussian tensors as in Section 3.2:

$$\Delta_b = \mathrm{RMS}(\Delta_b^W)\frac{Z_b}{\mathrm{RMS}(Z_b)}. \tag{5}$$

This *difference-calibrated search* changes the magnitude profile in Equation (2) while retaining the random-direction sampler. Its population is centered at zero injection, not at Image. Candidates are ranked by the task-based selection in Section 3.3; T_{iso} denotes the highest-ranked individual candidate. The two constructions distinguish applying a measured cache field from using its blockwise magnitudes to define a search population.

3.5 SHARED-WEIGHT POPULATION EVALUATION

The KV candidates constructed in Sections 3.2 and 3.4 share the base weights and differ through prefix-local fields. For n_L layers, H_{kv} KV heads per layer, and head dimension d, each field contains $2n_L L H_{\mathrm{kv}} d$ entries. Its dense storage grows linearly with prefix length L, independently of the length of subsequent generation. This stored intervention is distinct from the runtime KV state of an individual request.

Our vLLM extension batches requests carrying different fields under one resident model. Each request uses its associated field and maintains its own runtime KV state. The same execution path supports scoring a sampled population and generating predictions with the retained candidates, without switching the shared model weights.

Once the blockwise magnitudes are available, sampling a random field requires no model forward pass. Evaluating that field still requires candidate-conditioned prefix computation followed by its own generation trajectory. Implementation details and perturbation-storage accounting are given in Appendix E.

4 EXPERIMENTS

We first characterize the accessibility of task-improving candidates under source-free random prefix-KV perturbations. We then evaluate the final $N = 256$, $K = 8$ vote on examples disjoint from selection. Finally, we use weight-derived cache changes to investigate useful intervention fields and magnitude profiles.

Evaluation settings. The source-free campaign and the separate source-conditioned atlas each span 49 model–task cells: seven models from the Qwen and Llama families and seven benchmarks, GSM8K (Cobbe et al., 2021), Countdown (HuggingFaceTB, n.d.), ROCStories (Mostafazadeh et al., 2016), USPTO50K (Schneider et al., 2016), MBPP (Austin et al., 2021), MATH-500 (Lightman et al., 2023), and OlympiadBench (He et al., 2024) (section A). Within each study, P0 and candidate arms share the prefix text, greedy decoding rule, task evaluator, and comparison items. Reported gains are paired differences from same-cell P0 in task-score points, averaged equally across the relevant cells.

Source-free protocol. Each cell uses one bank of $N = 256$ candidates cycling through $\alpha \in \{0.01, 0.03, 0.05\}$. We characterize the full bank on $\mathcal{D}_{\text{sel}}$, rank candidates jointly across scales, and retain the top $K = 8$ for plurality voting. Candidate fields and selected identities are frozen before evaluation on selection-disjoint examples. Population density uses all 256 candidates; vote evaluation uses the same bank's eight selected candidates. Selection and evaluation rules are detailed in Appendices A, B, and C.

4.1 Can random prefix-KV perturbations find thickets?

We call a sampled candidate an expert when its selection-split task score exceeds same-cell P0. We measure expert density, the fraction of such candidates in the bank, and coverage, the number of cells containing at least one such candidate. These statistics use the entire bank and do not depend on the number of candidates retained for voting. The expert label denotes an observed score difference, not a per-candidate significance test.

Population accessibility. The bank contains an improving candidate in all 49 model–task cells, with a mean expert density of 28.7%. Improving candidates recur across the tested models and tasks, although three cells contain only one improving candidate each. Full coverage therefore does not imply uniformly dense experts. Across the nested checkpoints, coverage rises from 45/49 cells at $N = 32$ to 49/49 at $N = 256$, while mean density stays between 28.7% and 29.1% (table 1).

Table 1: **Source-free selection across 49 model–task cells.** Expert density is the mean fraction of candidates above same-cell P0; coverage counts cells with at least one such candidate. Both use selection rows. Each checkpoint keeps $K = N/32$.

(N, K)	Expert density	Coverage
$(32, 1)$	29.0%	45/49
$(64, 2)$	29.0%	46/49
$(96, 3)$	29.0%	47/49
$(128, 4)$	29.1%	48/49
$(192, 6)$	28.7%	48/49
$(256, 8)$	28.7%	49/49

Variation across scales and models. Improving candidates occur at all three tested scales, but their frequency and coverage differ (table 2). The scales contribute 86, 85, and 85 candidates per cell to the same bank. Their mean densities are 35.0%, 26.9%, and 24.2% at $\alpha = 0.01$, 0.03, and 0.05, respectively. These population statistics are distinct from the evaluation performance of the globally selected vote.

Table 2: **Per-scale selection at $N = 256$ across 49 cells.** The three scales contribute 86/85/85 candidates per cell. Expert density and coverage use selection rows; top-eight selection remains global.

α	Cand./cell	Expert density	Coverage
0.01	86	35.0%	47/49
0.03	85	26.9%	35/49
0.05	85	24.2%	37/49

Figure 2 compares the three fixed scales by task. Its evaluation panel describes selection-split experts within each scale, with different contributing cells; the mixed-scale vote is evaluated in the next subsection.

Across the seven models, nominal parameter count is positively associated with seven-task mean density (Spearman $\rho = 0.80$). This exploratory association includes model-level reversals and does not isolate model size from architecture or training (fig. 5 and table 6). Neural Thickets reports a similar model-scale tendency for random weight perturbations (Gan & Isola, 2026); here it appears in prefix-KV space.

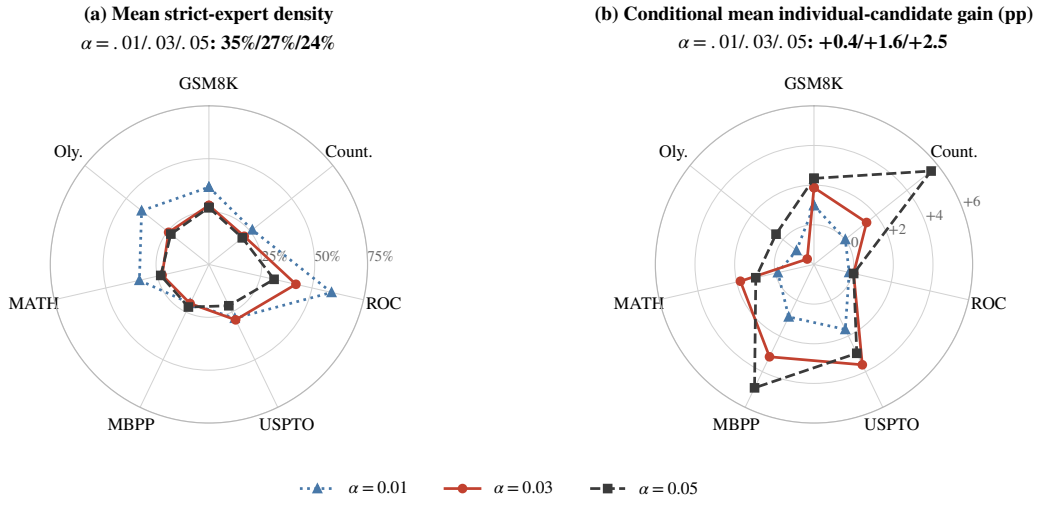


Figure 2: **Source-free task-wise means at $N = 256$.** (a) Selection-split expert density, averaging seven models per task. (b) Conditional individual-candidate gain over same-row P0: within each cell, we average the evaluation scores of selection-split experts among the top eight at that scale, then average the paired gains across contributing models. Cells without a qualifying expert are excluded; the contributing cell counts for $\alpha = 0.01/0.03/0.05$ are 47/35/37 of 49. This post-hoc diagnostic uses registered evaluation rows and reports individual-candidate scores rather than the global mixed-scale vote.

4.2 Does the selected KV vote improve evaluation performance?

The frozen $N = 256$, $K = 8$ mixed-scale vote is evaluated on registered selection-disjoint rows in all 49 model–task cells, paired with same-row P0. Candidate identities are fixed using $\mathcal{D}_{\text{sel}}$ before these rows are read (Appendix D).

The selected vote improves over P0 in 41 cells, ties in two, and falls below P0 in six. Its mean paired gain is $+4.83$ task-score points (95% cell-bootstrap interval $[+2.91, +7.09]$), and its median gain is $+1.60$ points. Mean gains are positive on all seven tasks, ranging from $+0.31$ points on ROCStories to $+10.93$ on Countdown (table 3). This result concerns the global mixed-scale vote, whereas fig. 2 reports a conditional diagnostic of individual candidates selected within each scale. Selection and evaluation rows are disjoint within each task. Per-model outcomes appear in table 6.

Table 3: **Selected source-free vote on registered evaluation rows.** Each task row averages the same seven models. P0 and vote scores are percentages; gain is the mean paired difference in task-score points. P/Z/N counts model cells with positive, zero, and negative gain.

Task	P0	Vote	Gain (pp)	P/Z/N
GSM8K	68.94	74.27	+5.33	7/0/0
Countdown	11.29	22.21	+10.93	6/0/1
ROCStories	50.70	51.01	+0.31	5/0/2
USPTO50K	16.26	21.39	+5.13	7/0/0
MBPP	32.43	37.09	+4.65	5/1/1
MATH-500	37.04	42.86	+5.82	7/0/0
OlympiadBench	17.25	18.87	+1.62	4/1/2
All 49 cells	33.42	38.24	+4.83	41/2/6

The evaluation scores also distinguish selected individuals from their vote. The first selection-ranked field gains $+2.70$ points over P0 on average, improving 30 cells, tying four, and declining in 15. Averaging the eight selected members' scores within each cell gives a $+1.99$-point mean gain; the

vote exceeds that mean-member score by 2.84 points. This difference is an exact accounting of the frozen predictor's scores, not an estimate of the causal effect of voting (Appendix C).

4.3 What connects weight-space experts and KV populations?

Source-free search finds useful KV candidates without a weight expert. We now use selected weight experts as probes, asking whether their induced prefix-cache changes are useful as particular intervention fields or as blockwise magnitude profiles for further search.

The separate 49-cell atlas selects W from 32 weight candidates, tests its measured cache difference directly as Image, and samples a bank of 128 KV candidates at that difference's blockwise RMS magnitudes. It selects one KV candidate by selection score; unlike the source-free predictor, this arm does not vote across candidates. Frozen arms are compared with same-session P0 on identical registered evaluation items. Their paired results are descriptive (Appendices A–F). The atlas retains its original ROCStories configurations and recomputes their scores under the fractional ordering metric without reselection.

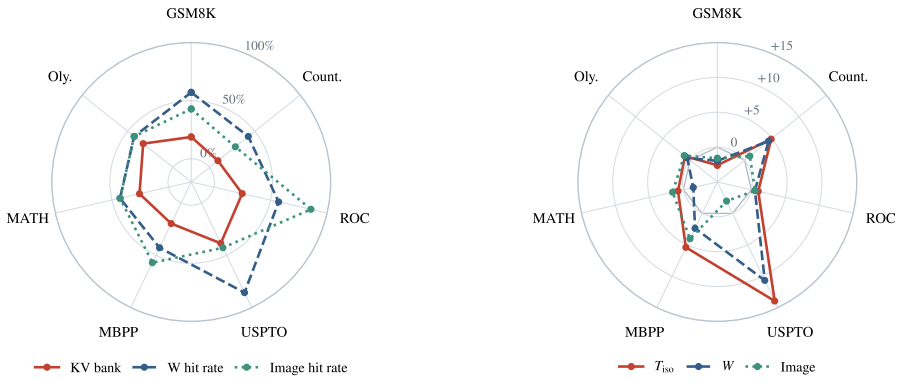


Figure 3: **Weight-derived KV interventions across 49 paired cells.** (a) Selection-split expert density within the calibrated KV bank, alongside the fractions of models whose single W or Image configuration beats P0. W and Image hit rates are not population densities. (b) Paired registered-test gains over P0 in task-score points. Each vertex averages seven models; cell-level values appear in fig. 8.

Direct weight-derived fields. Image tests whether the prefix-cache change induced by a selected weight candidate can itself serve as a useful context intervention. We inject this measured field into the frozen base model, without further search or selection in KV space. Image improves over P_0 in 24 of 49 model–task cells, with 5 ties and 20 regressions, yielding a mean gain of $+0.67$ task-score points (figs. 3 and 8). These improvements show that weight-derived cache fields can produce task gains under the original model weights, providing an empirical link from parameter-space candidates to useful context interventions. We next retain the measured fields' blockwise RMS magnitudes and sample fresh directions to explore this connection through population search.

Search at measured magnitudes. The difference-calibrated bank contains an improving candidate in 44/49 cells on selection rows. Its selected KV predictor, T_{iso}, gains $+3.48$ points over P0 on the registered evaluation rows, with 33 positive, 3 tied, and 13 negative outcomes. As a secondary comparison, W gains $+2.16$ points, and the selected KV predictor matches or exceeds W in 33/49 cells (figs. 3 and 8). These are outcomes under the atlas's respective search procedures, not an equal-budget comparison of weight and KV search.

Interpretation. The two tests distinguish a particular weight-derived field from a magnitude profile for exploration: direct Image effects are mixed, whereas random search at the measured magnitudes yields useful selected predictors in many atlas cells. The results support these magnitudes as one viable KV-search profile, not stable source-specific transfer. Because the source-conditioned and

source-free campaigns use different search budgets and aggregation procedures, their gains do not establish an advantage of one profile over the other.

5 LIMITATIONS

Evaluation and uncertainty. Each source-free model–task cell uses one sampled bank and a fixed task prefix. Expert density describes observed improvements on the selection split, not per-candidate statistical significance or the fraction of candidates that would improve on unseen examples. The selected vote is evaluated on examples disjoint from selection, with candidate identities frozen beforehand. This evaluation assesses performance on separate examples within the studied task pools. Cell-bootstrap intervals summarize variation among the observed, related cells, conditional on the sampled banks and fixed prefixes; they do not quantify variability across independently sampled banks or prefixes, or establish generalization beyond the evaluated models and task pools.

Search-protocol scope. The post-hoc scale curves condition on selection-split experts and average different contributing cells, so their ordering does not isolate a scale effect. Nested checkpoints vary N and K jointly and therefore do not isolate the effect of bank size on vote performance. The exploratory pilot chose $\alpha = 0.01$ after inspecting its grid, without establishing a held-out fixed-scale winner (Appendix G.1.1). The source-conditioned prefix study varies length and content together; source-free prefix sensitivity remains unmeasured.

Weight-derived evidence. Image provides feasibility evidence for constructing useful context interventions from weight-space candidates. Neither these improvements nor the ancillary controls establish stable source-specific capability transfer (Appendix F.1). The difference-calibrated atlas also uses previously examined evaluation pools. Its single-candidate predictor differs from the source-free vote in search budget and aggregation, precluding a head-to-head claim that either magnitude profile is superior.

Deployment scope. Candidate selection requires labeled task examples, and shared weights do not eliminate candidate-specific inference or the multiple generation trajectories used for voting. Our vLLM extension demonstrates offline batching under one resident model; online mixed-expert capacity, throughput, and latency remain unmeasured. Dense-field payload ratios compare stored perturbations, excluding runtime request caches and engine overhead.

6 CONCLUSION

KV Thickets shows that random search over fixed-prefix KV interventions can discover reusable task-improving candidates while leaving model weights and prefix text unchanged. Across 49 model–task settings, the source-free banks contain selection-split improvers, and the frozen eight-candidate vote improves 41 settings with a mean gain of 4.83 task-score points over the unperturbed greedy baseline. The first-ranked candidate also gains 2.70 points on average, showing that the observed improvements are not restricted to the ensemble.

Weight-induced cache differences provide a complementary probe: their direct effects are mixed, while their blockwise magnitudes define another useful sampling profile. The resulting candidate fields admit offline evaluation under shared weights, as our vLLM implementation shows. Because we previously examined some evaluation pools and each source-free setting uses one bank and prefix, these findings remain descriptive. Matched-budget controls and replication across new banks, prefixes, and evaluation examples are needed to establish when this shared-weight representation offers a practical alternative to weight-space expert populations.

AI USE STATEMENT

The author(s) used ChatGPT for minor grammatical adjustments, and the author(s) carefully reviewed and finalized all resulting edits.

Reproducibility Statement

The task pools, row selection, candidate ranking, scoring, and evaluation settings are specified in Appendices A, B and E.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Max Belitsky, Dawid J. Kopiczko, Michael Dorkenwald, M. Jehanzeb Mirza, James R. Glass, Cees G. M. Snoek, and Yuki M. Asano. KV cache steering for controlling frozen LLMs. *arXiv preprint arXiv:2507.08799*, 2025. URL https://arxiv.org/abs/2507.08799.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2005.14165.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 4005–4019, 2023. doi: 10.18653/v1/2023.findings-acl.247. URL https://aclanthology.org/2023.findings-acl.247/.

Yulu Gan and Phillip Isola. Neural thickets: Diverse task experts are dense around pretrained weights. *arXiv preprint arXiv:2603.12228*, 2026. URL https://arxiv.org/abs/2603.12228.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

HuggingFaceTB. Countdown-Task-GOLD. Hugging Face Datasets, n.d. URL https://huggingface.co/datasets/HuggingFaceTB/Countdown-Task-GOLD. Accessed September 2026.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. 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*, 2023.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/81b8390039b7302c909cb769f8b6cd93-Abstract-Conference.html.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, 2021. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Andy Zeyi Liu, Michael Zhang, Ilana Greenberg, Adam Alnasser, Lucas Baker, and John Sous. Memory inception: Latent-space KV cache manipulation for steering LLMs. *arXiv preprint arXiv:2605.06225*, 2026. URL https://arxiv.org/abs/2605.06225.

Luyang Liu, Jonas Pfeiffer, Jiaxing Wu, Jun Xie, and Arthur Szlam. Deliberation in latent space via differentiable cache augmentation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 39261–39274. PMLR, 2025. URL https://proceedings.mlr.press/v267/liu25bc.html.

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32287–32307. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24bx.html.

Alireza Mehrtash, Purang Abolmaesumi, Polina Golland, Tina Kapur, Demian Wassermann, and William Wells. PEP: Parameter ensembling by perturbation. *Advances in Neural Information Processing Systems*, 33:8895–8906, 2020.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 839–849, 2016. doi: 10.18653/v1/N16-1098.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3845–3864, 2023. doi: 10.18653/v1/2023.eacl-main.277.

Nadine Schneider, Nikolaus Stiefl, and Gregory A. Landrum. What's what: The (nearly) definitive guide to reaction role assignment. *Journal of Chemical Information and Modeling*, 56(12): 2336–2346, 2016. doi: 10.1021/acs.jcim.6b00564.

Ying Sheng, Shiyi Cao, Dacheng Li, Coleman Hooper, Nicholas Lee, Shuo Yang, Christopher Chou, Banghua Zhu, Lianmin Zheng, Kurt Keutzer, Joseph E. Gonzalez, and Ion Stoica. SLoRA: Scalable serving of thousands of LoRA adapters. In *Proceedings of Machine Learning and Systems*, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/906419cd502575b617cc489a1a696a67-Abstract-Conference.html.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 4222–4235, 2020. doi: 10.18653/v1/2020.emnlp-main.346.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. Black-box tuning for language-model-as-a-service. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 20841–20855. PMLR, 2022. URL https://proceedings.mlr.press/v162/sun22e.html.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Rui Wang, Junda Wu, Yu Xia, Tong Yu, Ruiyi Zhang, Ryan A. Rossi, Subrata Mitra, Lina Yao, and Julian McAuley. CachePrune: Teaching LLMs what not to follow via KV-cache editing. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 1551–1570, 2026. doi: 10.18653/v1/2026.acl-long.70. URL https://aclanthology.org/2026.acl-long.70/.

Dongjie Xu, Kai Qian, Julius, Weijie Shi, Yuxuan Sun, Minghua Tang, Fenglei Jin, Hanchi Dong, and Jiajie Xu. RippleKV: Cross-layer KV cache allocation via perturbation propagation. *arXiv preprint arXiv:2608.08684*, 2026. URL https://arxiv.org/abs/2608.08684.

APPENDICES

Appendices A–B specify the study protocol, data partitions, scoring, and candidate-selection rules. Appendices C–D provide the complete source-free results and selected-member analyses; Appendix E describes the implementation and storage accounting; and Appendix G specifies prompt and prefix construction.

A STUDY PROTOCOL AND EVALUATION DESIGN

The active source-free study and the difference-calibrated atlas answer different questions. The source-free study samples one cumulative bank per model–task cell, measures the fraction of candidates above same-cell P_0 on selection rows, and evaluates a frozen $N = 256$, $K = 8$ vote on separate rows. The atlas selects a weight candidate from 32 trials and one KV candidate from a separately calibrated bank of 128; its direct Image arm does not vote. Neither atlas gains nor its selection hit rates are added to the source-free population statistics.

Registered row pools. The source-free campaign uses seven Qwen and Llama models crossed with the seven tasks in table 4. Every cell uses the same task's 128 selection row IDs and registered evaluation row IDs. The row sets are disjoint within each task; "evaluation" here refers to the study's registered readout, not necessarily a benchmark's canonical test partition.

Table 4: **Source-free task pools and row counts.** Seven models per task; all arms use greedy decoding. m is the number of few-shot demonstrations in the fixed task prefix. "Max." gives newly generated tokens per answer.

Task	Source pool	$\lvert D_{\text{sel}}\rvert$	$\lvert D_{\text{eval}}\rvert$	m	Max.
GSM8K	train	128	1000	4	512
Countdown	train	128	1000	4	1024
ROCStories	train	128	1000	4	64
USPTO50K	train	128	1000	2	64
MBPP	combined full set	128	396	4	2048
MATH-500	test partition	128	108	4	2048
OlympiadBench	math/en/COMP	128	106	1	2048

The MBPP source concatenates its original train, validation, and test partitions before this study's row sampling. MATH-500 rows come from its 500-item test partition. OlympiadBench uses the text-only English competition mathematics subset `OE_TO_maths_en_COMP`. ROCStories is a five-sentence ordering adaptation. USPTO50K rows come from the training partition, so its evaluation is not a

canonical test read. The first 200 pool rows were excluded before sampling. Selection and evaluation IDs do not overlap in any task. This row separation alone does not provide an independent test on new task pools.

Candidate selection and generation settings. Candidates are ranked by selection task score descending, then the fraction of nonempty normalized answers descending, then zero-based candidate ID ascending. The $N = 256$ population cycles through $\alpha \in \{0.01, 0.03, 0.05\}$ with seed 20260924. All arms use BF16 and temperature zero. The maximum model context length varies by model and task; the answer-length limits in table 4 give the maximum number of newly generated tokens.

B Task Scoring, Normalization, and Voting

Each candidate and P_0 is scored on identical row IDs with the same task handler. Selection ranks the mean of per-row task scores, using the nonempty-answer fraction only to break equal scores. That fraction is a parseability statistic, not the correctness rate and not proof that a nonempty answer obeys every task constraint. Maximum generation lengths and row counts appear in table 4; the full messages and chat-template convention are in section H.

Arithmetic and mathematics. GSM8K extracts the last `####` number in the final 300 characters, falling back to the last numeric string, and scores exact string agreement with the reference after the scorer's comma handling. MATH-500 and OlympiadBench extract a final boxed answer where possible, then apply the registered fallback extraction. Their scorer accepts normalized string and limited LaTeX, numeric, fraction, and coordinate equivalences; it is not a general symbolic-equivalence oracle. OlympiadBench here is the text-only English competition-mathematics subset in section A.

Countdown scores an individual answer only when its arithmetic expression uses the supplied numbers exactly once and reaches the target. During voting, a permitted expression is evaluated and contributes its numeric result; distinct correct expressions can therefore share one vote. An expression with invalid characters, a wrong number multiset, an invalid equation, or failed evaluation abstains. The voted numeric result is correct when it matches the target within 10^{-5}. The scoring path's format reward is not used as the selection accuracy: the campaign uses the binary answer-correctness result.

Classification, code, and ordering. USPTO50K removes atom-map indices from reaction SMILES in the prompt. Its scorer extracts a class number from the last `<answer>` tag, or from a fallback scan, and compares integer classes 1–10. A nonempty but wrong class can participate in the vote. MBPP exposes the first three provided tests in the prompt and evaluates generated Python code against the full provided `test_list` with setup code. The registered score is all-tests-pass, not a fraction of tests. Code execution and each test use a five-second timeout in the scoring worker; a failed or timed-out row scores zero. Voting compares extracted code strings without executing them first, so a nonempty but failing program can win a lexical tie.

ROCStories requires one permutation of labels A–E. For prediction π and gold order g, its fractional score is

$$s(\pi, g) = 0.6\,\frac{1}{5}\sum_{j=1}^{5}\mathbf{1}[\pi_j = g_j] + 0.4\,\frac{1}{4}\sum_{j=1}^{4}\mathbf{1}[\mathrm{pos}_\pi(g_j) < \mathrm{pos}_\pi(g_{j+1})]. \tag{6}$$

An invalid or repeated-label order scores zero and is excluded from voting. This sentence-ordering score is distinct from the original story-cloze task. The weight-derived atlas retained candidates selected under its historical binary metric and rescored them under this fractional metric without reselection (section F).

Deterministic plurality. For each row, only nonempty handler-normalized answers count. The answer with the largest count wins; tied largest counts resolve to the lexicographically smaller string. For example, one vote for class `10` and one for class `2` selects `10`, because the comparison is lexical rather than numeric. When every member abstains, the vote is the empty string and scores zero. Except where a handler explicitly rejects invalid answers during extraction, nonempty answers that later score zero still participate. The vote rule uses no evaluation labels to normalize or break ties.

C Complete Source-Free Results

The source-free campaign uses one cumulative bank per model–task cell with $(N, K) = (32, 1), (64, 2), (96, 3), (128, 4), (192, 6), (256, 8)$ checkpoints. These are total candidate counts, not per-scale counts. Stable zero-based IDs cycle through $\alpha \in \{0.01, 0.03, 0.05\}$, yielding 11/11/10 candidates per scale at $N = 32$, 22/21/21 at $N = 64$, 32/32/32 at $N = 96$, 43/43/42 at $N = 128$, 64/64/64 at $N = 192$, and 86/85/85 at $N = 256$. The $N = 192$ bank adds IDs 128–191 in one 64-candidate extension; its $K = 6$ vote is an intermediate nested checkpoint. The $N = 256$ checkpoint adds IDs 192–255 and freezes the global top eight. The reported source-free predictor is this final $N = 256$, $K = 8$ vote; smaller checkpoints serve the nested-bank sensitivity analysis. Every block is sampled at the candidate's multiplier of its clean-cache RMS, and saved fields are applied without further rescaling. The fixed prefix, greedy decoding, and task evaluator are shared across P0 and candidate arms. At each checkpoint, we rank the cumulative bank by selection task score, then validity, then ascending candidate ID; the global top K are frozen for evaluation. Each task handler normalizes their answers before a row-wise plurality vote. Nonempty answers vote, and a tie goes to the lexicographically smaller normalized answer.

C.1 Readout and Units

The 128 selection rows and the registered evaluation rows are disjoint within each task pool (table 4). Scores are means of item scores in $[0, 1]$, multiplied by 100; gains subtract the same-cell P_0 score in task-score points. The task-specific extraction, scoring, validity, and deterministic vote rules are stated in section B.

C.2 Population-Size Stability

The nested candidate banks have similar aggregate selection-split expert fractions (28.7–29.1% across the six checkpoints in table 1). The pooled candidate-score distributions at all six checkpoints have similar shapes across the seven tasks (fig. 4), but aggregate stability does not hold exactly in every cell. Individual densities shift by up to 15.2 percentage points between $N = 32$ and $N = 256$, and larger banks recover four rare-expert cells that have no expert at $N = 32$. The density uses every candidate and is independent of the selected vote size K; the changing (N, K) checkpoints do not isolate an effect of N on selected-vote performance. At $N = 256$, the three fixed scales have densities of 35.0%, 26.9%, and 24.2% and coverage of 47/49, 35/49, and 37/49 cells, respectively (table 2).

Selection and registered-evaluation rows are disjoint in every cell. The primary $N = 256$, $K = 8$ vote uses each cell's registered evaluation rows; its candidate identities were frozen on selection rows.

For the scale-restricted diagnostic in fig. 2, we additionally freeze the top eight candidates within each scale using the same selection ordering. For the expert diagnostic, we retain only candidates whose selection score exceeds same-cell selection P0, average their registered-evaluation scores within the cell, and subtract same-row evaluation P0. Cells with no such candidate are excluded from that scale's mean, rather than assigned P0. The figure averages the remaining paired gains across models by task. In task order GSM8K, Countdown, ROCStories, USPTO50K, MBPP, MATH-500, and OlympiadBench, the numbers of contributing models for $\alpha = 0.01/0.03/0.05$ are 7/5/5, 7/5/5, 7/5/5, 7/6/7, 6/5/5, 7/5/5, and 6/4/5, respectively. Thus the curves compare conditional means over different sets of model–task cells, not a matched-scale effect. This post-hoc analysis does not change the globally selected vote and shares its evaluation rows.

The $N = 256$ bank's per-scale selection statistics appear in table 2. The final $N = 256$, $K = 8$ vote and intermediate checkpoint gains appear in table 5, with final-checkpoint per-model detail in table 6. These nested readouts share registered evaluation rows and do not measure variation across independently sampled banks or prefixes.

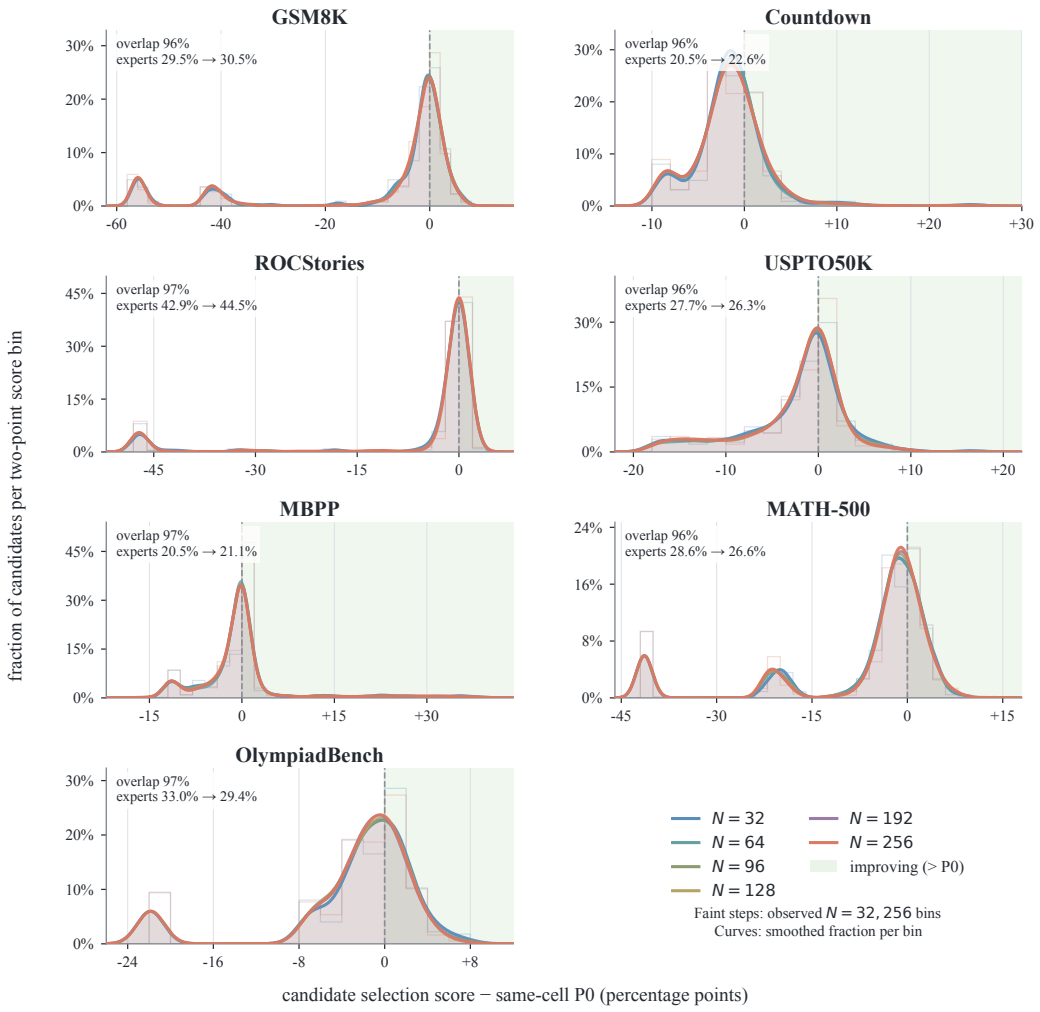


Figure 4: **Pooled candidate-score distributions across six bank sizes.** Each task panel pools seven models at $N = 32, 64, 96, 128, 192, 256$, with $7N$ candidate selection scores per curve. The horizontal axis is candidate score minus its own model–task P0 in percentage points; green marks strict improvement. The vertical axis is the fraction of candidates in a two-point score bin. Faint steps show observed endpoint-bin fractions; curves are Gaussian-smoothed guides with a shared task-specific bandwidth. Insets give exact expert fractions and smoothed distribution overlap between $N = 32$ and $N = 256$. Candidates from smaller banks are nested in larger ones.

Table 5: **Registered-row evaluation across 49 cells.** Gains are from frozen votes over rows disjoint from candidate selection; intervals bootstrap cells. P/Z/N counts positive, zero, and negative gains.

(N, K)	Gain [95% CI] (pp)	Median (pp)	P/Z/N
$(64, 2)$	$+3.24$ $[+1.47, +5.38]$	$+0.94$	36/2/11
$(96, 3)$	$+3.89$ $[+2.18, +5.92]$	$+0.94$	37/4/8
$(128, 4)$	$+4.28$ $[+2.51, +6.38]$	$+1.42$	39/3/7
$(192, 6)$	$+4.57$ $[+2.71, +6.82]$	$+1.85$	44/0/5
$(256, 8)$	$+4.83$ $[+2.91, +7.09]$	$+1.60$	41/2/6

At $N = 256$, expert density varies strongly across tasks and model families (fig. 5). Across the seven models, the rank correlation between nominal parameter count and seven-task mean density is $\rho = 0.80$. An exact two-sided permutation test gives $p = 208/5040 \approx 0.041$: it reassigns whole-model size labels while retaining each model's seven task densities together. The task-wise rank

correlations are all positive (0.06–0.87), but the curves are not monotone; in particular, Qwen2.5-1.5B has lower density than Qwen2.5-0.5B on five of seven tasks. This exploratory association does not isolate model size from architecture or training and is not a scaling law.

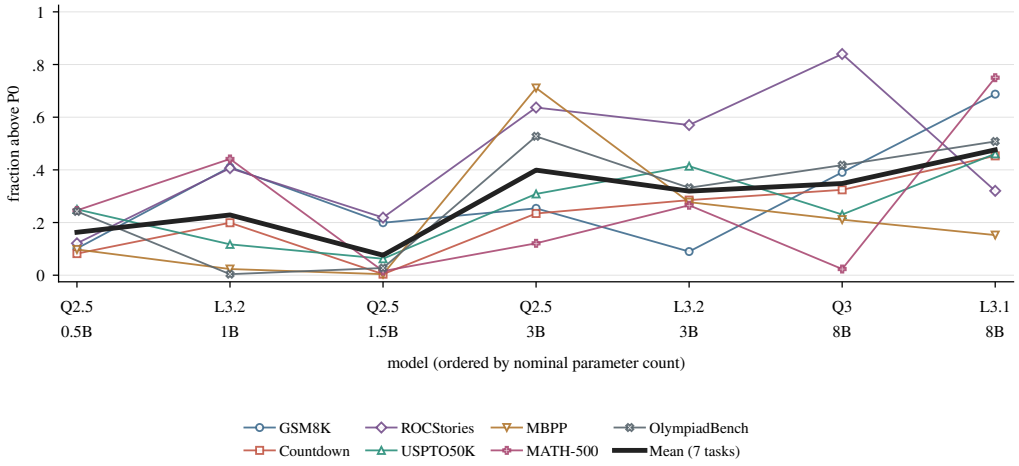


Figure 5: **Task-level source-free expert density at $N = 256$.** Each colored curve follows one task across seven models ordered by nominal parameter count; every marker is one observed model–task cell. The bold dark line is the mean density across the seven tasks for each model. Lines guide the eye and do not represent a fitted scaling law.

Table 6: **Source-free $N = 256$, $K = 8$ outcomes by model.** Coverage and density use selection rows; vote gains use registered rows disjoint from candidate selection. P/Z/N counts gain signs versus P0.

Model	Coverage	Mean density	Mean gain (pp)	P/Z/N
Qwen2.5-0.5B	7/7	16.3%	+3.91	4/1/2
Qwen2.5-1.5B	7/7	7.6%	+3.02	6/0/1
Qwen2.5-3B	7/7	39.9%	+10.12	7/0/0
Llama-3.2-1B	7/7	22.9%	+1.29	6/0/1
Llama-3.2-3B	7/7	31.9%	+4.65	7/0/0
Qwen3-8B	7/7	34.8%	+6.33	6/0/1
Llama-3.1-8B	7/7	47.6%	+4.47	5/1/1
All 49 cells	49/49	28.7%	+4.83	41/2/6

The complete cell-level selection fractions and paired vote gains appear in table 7. The selected IDs for each cell appear in table 8; they are zero based and listed in selection-rank order. These are the same expanding banks, not independent replications of field sampling.

Table 7: **Source-free results in all 49 model–task cells at $N = 256$, $K = 8$.** Panel A gives the fraction of candidates exceeding same-cell P_0 on selection rows. Panel B gives the paired gain of the frozen eight-member vote over P_0 on evaluation rows.

	Qwen2.5			Llama 3.2		Qwen3	Llama 3.1
Task	0.5B	1.5B	3B	1B	3B	8B	8B
A. Improving candidates on selection rows (%)							
GSM8K	10.2	19.9	25.4	41.0	9.0	39.1	68.8
Countdown	8.2	0.4	23.4	19.9	28.5	32.4	45.3
ROCStories	12.1	21.9	63.7	40.6	57.0	84.0	32.0
USPTO50K	25.0	6.2	30.9	11.7	41.4	23.0	46.1
MBPP	9.8	0.4	71.1	2.3	27.7	21.1	15.2
MATH-500	24.6	1.6	12.1	44.1	26.6	2.3	75.0
OlympiadBench	24.2	2.7	52.7	0.4	33.2	41.8	50.8
B. Frozen vote gain on evaluation rows (task-score points)							
GSM8K	+8.70	+5.40	+16.80	+3.40	+0.80	+0.60	+1.60
Countdown	-0.70	+0.60	+1.10	+1.40	+18.00	+33.90	+22.20
ROCStories	-0.32	-0.29	+0.90	+0.11	+1.27	+0.06	+0.44
USPTO50K	+12.50	+2.10	+10.90	+0.20	+6.30	+3.30	+0.60
MBPP	+2.53	+0.25	+29.04	+0.25	+1.52	-1.01	+0.00
MATH-500	+4.63	+7.41	+9.26	+4.63	+3.70	+3.70	+7.41
OlympiadBench	+0.00	+5.66	+2.83	-0.94	+0.94	+3.77	-0.94

Table 8: **Selected candidate IDs at $N = 256$, $K = 8$.**

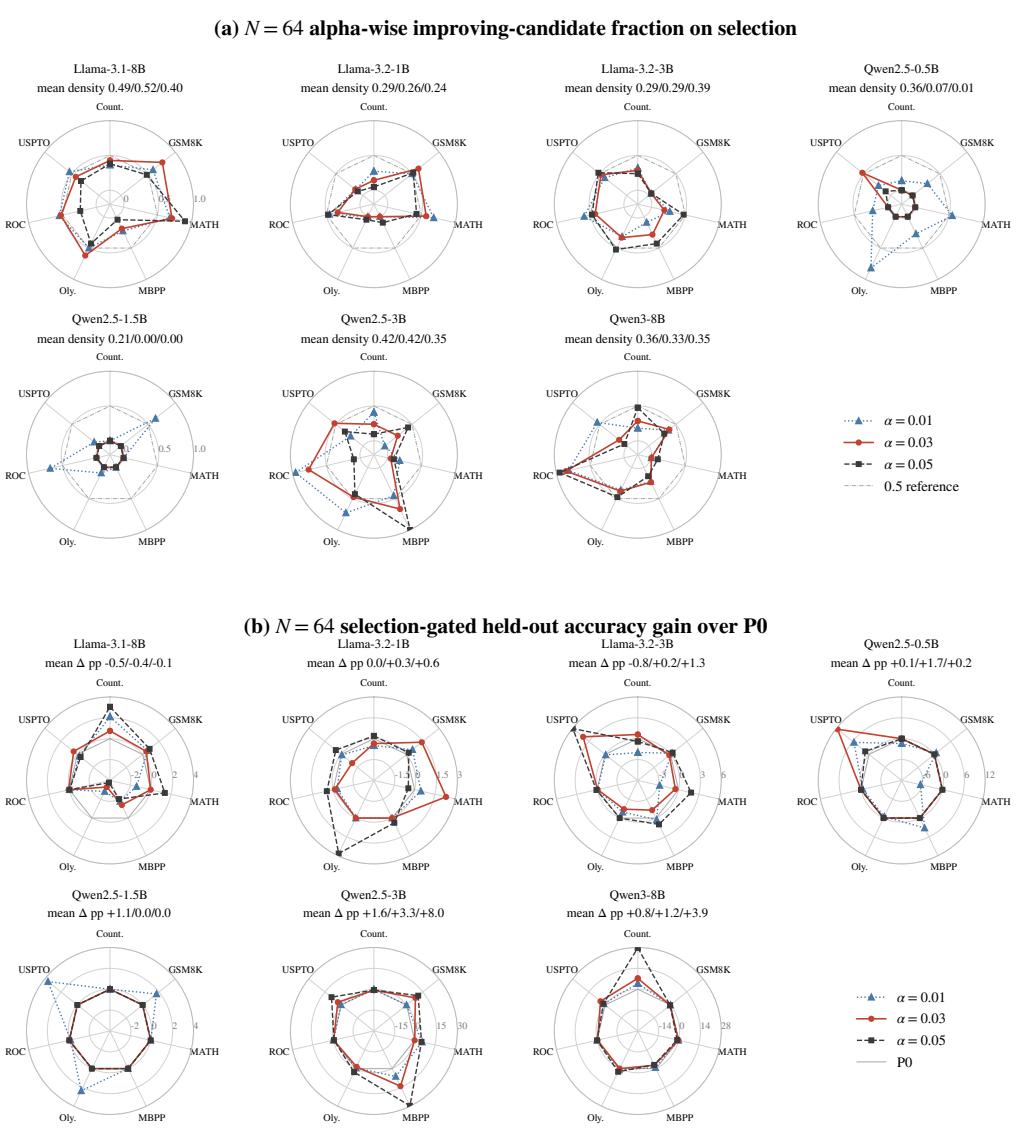
Model	Task	Selected IDs
Q2.5-0.5	GSM8K	9, 243, 144, 21, 45, 159, 186, 207
Q2.5-0.5	Countdown	192, 69, 183, 210, 144, 162, 99, 138
Q2.5-0.5	ROCStories	162, 240, 69, 66, 129, 126, 174, 42
Q2.5-0.5	USPTO50K	1, 82, 244, 103, 4, 63, 93, 97
Q2.5-0.5	MBPP	60, 153, 225, 174, 12, 186, 204, 138
Q2.5-0.5	MATH-500	231, 207, 111, 186, 132, 252, 72, 150
Q2.5-0.5	OlympiadBench	168, 3, 228, 129, 123, 132, 150, 237
Q2.5-1.5	GSM8K	72, 204, 18, 81, 84, 117, 147, 33
Q2.5-1.5	Countdown	231, 201, 111, 12, 3, 21, 216, 57
Q2.5-1.5	ROCStories	159, 162, 252, 255, 18, 225, 72, 135
Q2.5-1.5	USPTO50K	21, 98, 54, 168, 186, 141, 150, 162
Q2.5-1.5	MBPP	111, 56, 221, 53, 115, 104, 80, 35
Q2.5-1.5	MATH-500	81, 141, 189, 174, 15, 72, 246, 201
Q2.5-1.5	OlympiadBench	9, 99, 249, 159, 72, 207, 57, 39
Q2.5-3	GSM8K	191, 86, 143, 26, 65, 137, 209, 227
Q2.5-3	Countdown	130, 137, 127, 169, 91, 0, 186, 23
Q2.5-3	ROCStories	229, 38, 34, 73, 133, 54, 1, 67
Q2.5-3	USPTO50K	26, 161, 29, 101, 104, 59, 95, 122
Q2.5-3	MBPP	116, 215, 11, 56, 137, 68, 80, 218
Q2.5-3	MATH-500	247, 227, 212, 24, 109, 166, 23, 135
Q2.5-3	OlympiadBench	21, 99, 45, 135, 220, 87, 0, 164
L3.2-1	GSM8K	20, 47, 94, 126, 229, 232, 46, 56
L3.2-1	Countdown	197, 213, 222, 112, 108, 150, 200, 191
L3.2-1	ROCStories	215, 32, 89, 29, 80, 245, 197, 179
L3.2-1	USPTO50K	38, 112, 238, 125, 2, 6, 7, 60
L3.2-1	MBPP	251, 119, 5, 41, 74, 179, 116, 23
L3.2-1	MATH-500	155, 238, 145, 161, 20, 17, 25, 117
L3.2-1	OlympiadBench	47, 110, 185, 14, 120, 213, 66, 241
L3.2-3	GSM8K	218, 113, 221, 16, 106, 153, 154, 201
L3.2-3	Countdown	224, 59, 97, 197, 119, 77, 104, 125

Table continued from previous page

Model	Task	Selected IDs
L3.2-3	ROCStories	206, 239, 44, 179, 149, 218, 107, 203
L3.2-3	USPTO50K	134, 38, 104, 227, 10, 31, 125, 170
L3.2-3	MBPP	67, 170, 148, 121, 35, 64, 76, 83
L3.2-3	MATH-500	67, 44, 107, 171, 14, 145, 98, 154
L3.2-3	OlympiadBench	11, 182, 46, 73, 142, 229, 205, 211
Q3-8	GSM8K	207, 1, 11, 13, 145, 220, 243, 244
Q3-8	Countdown	44, 2, 101, 173, 128, 140, 137, 56
Q3-8	ROCStories	14, 32, 74, 170, 209, 230, 5, 140
Q3-8	USPTO50K	209, 110, 164, 1, 13, 21, 184, 66
Q3-8	MBPP	217, 10, 13, 21, 26, 35, 39, 90
Q3-8	MATH-500	196, 14, 118, 41, 147, 179, 51, 72
Q3-8	OlympiadBench	10, 62, 35, 15, 121, 134, 7, 145
L3.1-8	GSM8K	25, 40, 46, 43, 44, 88, 142, 190
L3.1-8	Countdown	90, 44, 230, 79, 85, 18, 88, 84
L3.1-8	ROCStories	119, 101, 32, 31, 202, 7, 55, 197
L3.1-8	USPTO50K	155, 143, 78, 85, 167, 23, 31, 34
L3.1-8	MBPP	214, 252, 202, 3, 2, 9, 48, 37
L3.1-8	MATH-500	41, 217, 14, 215, 167, 34, 40, 116
L3.1-8	OlympiadBench	71, 230, 64, 5, 247, 17, 32, 31

The cumulative continuation resolves both Qwen2.5-1.5B cells that had no selection expert at $N = 96$. MBPP first gains candidate 111 at $N = 128$; Countdown first gains candidate 231 at $N = 256$. Each has one expert among 256 candidates. On registered evaluation rows, the frozen top-eight vote gains $+0.6$ points on Countdown and $+0.25$ points on MBPP, whose P0 score is zero. These small evaluation differences do not turn the selection-density exception into a general size trend.

The $N = 64$, $K = 2$ post-hoc scale diagnostic ranks each scale subset by the same selection-only rule and reads its selected candidates separately on registered evaluation rows. A subset with no improving selection candidate reports P0; a singleton reports that candidate; otherwise the policy averages the highest-ranked improving candidates up to two. Negative evaluation gain from a selection improver is retained. This is a mean of available candidate scores with a P0 fallback, not the primary row-wise vote. Cell-bootstrap 95% intervals for the 0.01, 0.03, and 0.05 scale-specific gains are $[-0.27, +0.92]$, $[+0.07, +1.82]$, and $[+0.51, +3.85]$ points, respectively. These descriptive curves do not redefine the global selection arm.



Means follow $\alpha = .01/.03/.05$; (b) uses P0 if none improve on selection, otherwise the top two available candidates.

Figure 6: **Per-model source-free scale diagnostic at $N = 64$.** Seven model panels disaggregate the per-scale diagnostic. (a) Scale-specific selection improving-candidate fraction for each model and task. (b) Selection-gated gain on registered evaluation rows over same-row P0: an empty improving set falls back to P0, a singleton uses that candidate, and a set of at least two uses the selection-ranked top two. Panel-title values follow $\alpha = 0.01/0.03/0.05$ and average the model's seven tasks. The gain panels use model-specific symmetric percentage-point axes. This is a post-hoc read of registered-evaluation rows, not the primary global top-2 vote.

D Selected-Member Analysis and Uncertainty

Selected members and plurality outcomes. We read the registered-evaluation scores of the eight selection-ranked members of the final $N = 256$, $K = 8$ vote without selecting any candidate on those rows. For the selected set I, let $S_{\text{member}} = |I|^{-1} \sum_{i \in I} S(T_i)$, $T_i = T(\Delta^{(i)})$, V_i denote the plurality predictor over I, and S denote the evaluation task score, where all scores use the same cell's evaluation items. The identity

$$S(V_I) - S(P_0) = \big[S_{\text{member}} - S(P_0)\big] + \big[S(V_I) - S_{\text{member}}\big] \tag{7}$$

separates the observed mean-member gain from the aggregation increment; it does not assign a causal effect to either term. Across 49 cells, these terms average $+1.99$ and $+2.84$ task-score points, respectively. The first-ranked field averages $+2.70$ points and improves 30 cells, ties four, and declines in 15. Its rank was fixed by selection score alone.

Across 32,270 evaluation rows, the vote also has two boundary cases. Among them, 2,853 have multiple answers tied for the largest nonempty plurality and therefore use lexical tie-breaking; 1,788 have no nonempty member answer and receive an empty vote, scored as zero. These are row counts across model–task cells, not independent examples. Every reported vote was checked against the stated plurality rule.

Table 9: **Selected-member and vote gains by task.** Gains are task-score points over same-row P_0, averaged over seven models. Tie rows count non-unique largest nonempty pluralities across cells.

Task	First	Mean member	Vote	Tie rows
GSM8K	+2.14	+1.79	+5.33	341
Countdown	+4.14	+3.75	+10.93	537
ROCStories	-0.16	-0.10	+0.31	464
USPTO50K	+7.10	+4.29	+5.13	294
MBPP	+4.22	+3.95	+4.65	757
MATH-500	+0.26	+0.78	+5.82	167
OlympiadBench	+1.21	-0.54	+1.62	293
All 49 cells	+2.70	+1.99	+4.83	2,853

Uncertainty and control scope. The reported 95% cell-bootstrap interval $[+2.91, +7.09]$ points resamples the 49 observed cell gains. It describes variation across these related model–task cells, conditional on their fixed prefixes and banks; it does not measure uncertainty over new banks, prefixes, or candidate selection. We did not perform a new paired-example rescore. The positive vote gain establishes the performance of the frozen predictor, without isolating field identity from scale choice or multi-answer diversity.

E Systems and Implementation Details

Cache fields are applied without rescaling. Weight candidates include merged QKV/gate–up and tied-output tensors. Fields contain only prefix perturbations; a runtime position mask keeps the intervention zero after the prefix. The model–task context limit varies as described in Appendix A.

Candidate-slot and reference gathers, serial and parallel draws, and reconstructed image banks were checked for bit equality. Selection and test use the same generation settings. Clean generations before and after each field audit its removal.

E.1 Per-Candidate Perturbation Storage

We compare the dense BF16 payload of a materialized candidate in the source-free KV bank with that of the registered Neural Thickets weight perturbation. For a prefix of L tokens, the KV field contains $2n_L L H_{\mathrm{kv}} d$ values (keys and values), or $B_{\mathrm{KV}} = 4n_L L H_{\mathrm{kv}} d$ bytes in BF16. The weight payload is $B_W = 2\sum_{w \in \mathcal{W}_{\mathrm{lin}}} \mathrm{numel}(w)$ bytes, where $\mathcal{W}_{\mathrm{lin}}$ contains the distinct linear weights changed by the weight-search implementation, counting a tied output weight once. We obtained these sizes from checkpoint tensor shapes and the registered prefix lengths in all 49 model–task cells. Task means appear in fig. 7.

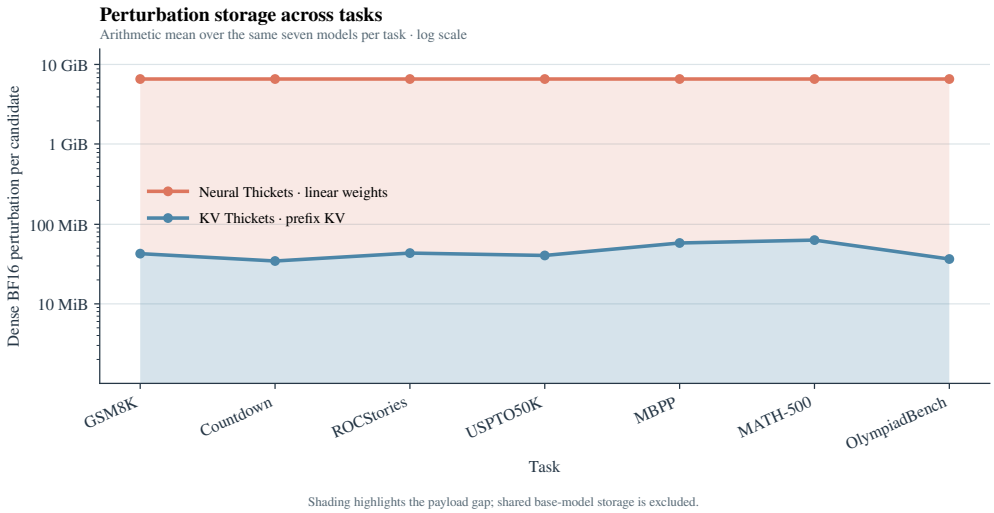


Figure 7: **Dense perturbation payload per candidate across tasks.** Each line is the arithmetic mean over the same seven BF16 models; the vertical axis is logarithmic. The weight payload is constant across tasks because the models are fixed, while the prefix-KV payload varies with prefix length. Filled regions highlight the gap. Shared base weights, request caches, and engine overhead are excluded.

Across individual cells, B_W/B_{KV} ranges from 62.5 to 328.3 (median 149.4). This compares materialized candidate tensors; storing only random seeds or serving candidates live has different memory costs.

F Weight-Derived Atlas and Source Controls

The difference-calibrated evaluation spans 49 cells: seven tasks across seven Qwen and Llama models. In each cell, W is selected from 32 weight candidates and T_{iso} is the single highest-ranked candidate from 128 difference-calibrated KV fields. Both selections rank by task score, then validity, then lowest candidate ID; the frozen arms are compared on identical registered-test rows. There is no vote in this atlas. KV-bank density is measured on $\mathcal{D}_{\mathrm{sel}}[0{:}128]$, whereas W and Image each contribute one binary selection-split hit rather than a population density.

ROCStories retains historically frozen candidates while recomputing their scores and density under the registered fractional metric, without reselection; the candidates were frozen under the binary ordering metric.

Across the 49 cells, Spearman rank correlations between selection density and the KV test gain ($\rho = 0.52$) and KV–W gap ($\rho = 0.54$) are positive, but density does not determine either outcome. Same-split maxima are in-sample oracle references over the sampled bank, not generalization bounds. These correlations are descriptive, and the source-conditioned atlas does not provide a matched-budget comparison with source-free search.

Paired registered-test atlas. Figure 8 shows selection density and paired test gains for each model–task cell.

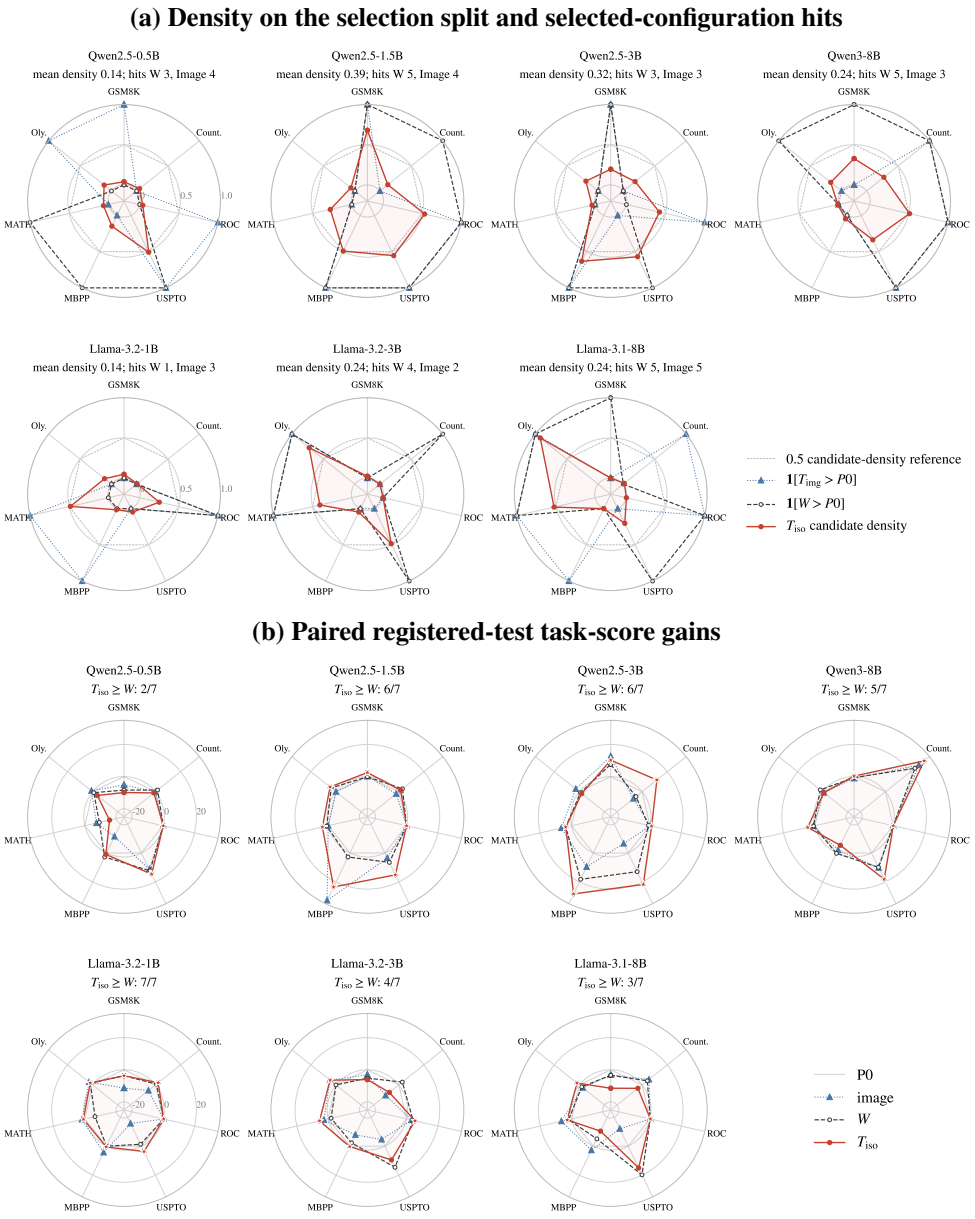


Figure 8: **Difference-calibrated selection density and paired test gains.** (a) Red shows the fraction of an $N = 128$ KV bank above P0 on selection rows; the 0.5 reference applies only to this curve. Dashed W and dotted Image curves are binary hits for one frozen configuration each. (b) Registered-test gains over P0; stars mark $T_{\text{iso}} \geq W$.

F.1 Direct Image Source-Specificity Diagnostic

This ancillary check asks whether a single selected target weight candidate's measured Image outperforms fields with the same blockwise scale but altered coordinates or source candidate. For target task t, let W_t be its selected weight candidate and W_u the selected candidate for a fixed wrong task u. We measure both differences on the *target* prefix p_t^*: $\Delta_t = C_{W_t}(p_t^*) - C_\theta(p_t^*)$ and $\Delta_{u\to t} = C_{W_u}(p_t^*) - C_\theta(p_t^*)$. For each layer–stream block $B = (\ell, S)$, the wrong-source control uses

$$\widetilde{\Delta}_{u\to t,B} = \Delta_{u\to t,B}\,\frac{\mathrm{RMS}(\Delta_{t,B})}{\mathrm{RMS}(\Delta_{u\to t,B})}, \qquad T_{\text{wrong}} = T(\widetilde{\Delta}_{u\to t}). \tag{8}$$

When both block RMS values are zero, the implementation writes a zero block. If the target RMS is positive but the wrong-source RMS is zero, the matched field is undefined and the run raises an error; the equation is used only for nonzero denominators. The coordinate control $T^{(s)}_{\text{shuffle}}$ independently permutes entries of $\Delta_{t,B}$ within each block for four fixed seeds. It does not mix layers, keys with values, or positions outside the target prefix. Both controls retain the target Image's blockwise RMS. They compare single fields without task-score selection, so they do not test either population search or whether the target expert's RMS profile is useful for calibration.

Six target/wrong-source pairs were fixed before running the controls (table 10). The four seeds are 104729, 130363, 155921, and 196613. All arms share target items, prefix, greedy decoding, and scoring. P0 was repeated after the controls and matched its initial per-item outputs in all six cells. Field audits verified permutations, prefix support, and block norms; the largest wrong-source RMS mismatch was 0.19%. The unadjusted 95% intervals in table 10 are percentile intervals from 10,000 paired bootstrap resamples of the 128 target rows (seed 20260924). Each resample uses the same row indices for both arms; for the shuffle contrast, the four shuffle scores are averaged within each row before resampling.

Table 10: **Measured Image specificity in six task pairs.** Entries are paired task-score differences in points. Each estimate is followed by its 95% interval on the next line.

Model	Target ← wrong	vs. P_0	vs. shuffles	vs. wrong	Shuffles
Q2.5-3B	GSM8K ← USPTO	$+10.9$	$+17.6$	$+3.9$	56.2–82.8 (11.3)
		$[+3.1, +18.8]$	$[+12.1, +23.0]$	$[-3.1, +10.9]$	
	USPTO ← GSM8K	-7.8	-15.6	-2.3	7.8–26.6 (8.6)
		$[-14.1, -2.3]$	$[-21.1, -10.0]$	$[-8.6, +3.1]$	
Q2.5-1.5B	MBPP ← Count.	$+33.6$	$+33.2$	$+15.6$	0.0–1.6 (0.8)
		$[+25.8, +42.2]$	$[+25.2, +41.4]$	$[+7.0, +24.2]$	
	Count. ← MBPP	-0.8	-2.5	0.0	0.0–12.5 (6.1)
		$[-3.9, +1.6]$	$[-4.3, -0.6]$	$[-2.3, +2.3]$	
L3.2-3B	ROC ← USPTO	-6.6	$+27.1$	-0.8	0.3–36.5 (17.6)
		$[-10.6, -2.9]$	$[+24.0, +30.2]$	$[-4.5, +2.9]$	
	USPTO ← ROC	-2.3	-6.8	$+2.3$	10.9–21.9 (4.8)
		$[-8.6, +3.9]$	$[-13.3, -0.2]$	$[-4.7, +9.4]$	

Note. Shuffles gives the score range (SD) of four coordinate permutations; vs. shuffles compares the real Image with their mean. The vs. wrong column compares it with the wrong-source Image. Q2.5 and L3.2 denote Qwen2.5 and Llama-3.2 Instruct. USPTO, Count., and ROC abbreviate USPTO50K, Countdown, and ROCStories; ROCStories uses its fractional score.

The real Image exceeds the wrong-source image and all four permutations in only two of six cells by point estimate. On GSM8K, its $+3.91$-point advantage over the wrong-source image has a paired interval crossing zero; only MBPP has positive unadjusted paired intervals against P0 and every control. The other four cells do not meet the point-estimate criterion. This six-pair diagnostic does not establish stable expert-source-specific transfer. Exact model identity across the source and control sessions was not established. A cross-run GSM8K P0 read also differs from this diagnostic's same-session P0, so we use only within-run paired contrasts. Four shuffle seeds are within-cell controls, not independent model–task cells.

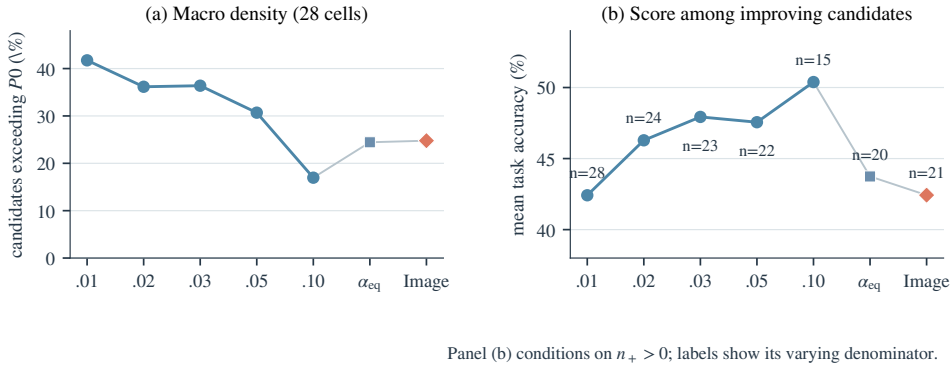
G Secondary Studies and Historical Pilots

G.1 Scale and Prefix Ablations

G.1.1 Exploratory KV-Scale Pilot

This 28-cell calibration pilot compared fixed scales with Image-derived controls in the measured-Image pipeline. Its fixed-scale arms use clean-cache RMS. The aggregate comparison appears in

fig. 9; tables 11 and 12 report the cell-level fractions and conditional scores separately from the 49-cell source-free campaign.



Panel (b) conditions on $n_+ > 0$; labels show its varying denominator.

Figure 9: **Fixed and weight-derived profiles in an exploratory 28-cell pilot.** (a) Mean improving-candidate fraction across all cells. (b) Mean task score among improving candidates, conditional on a nonempty set; labels give the number of contributing cells. This pilot is separate from the 49-cell source-free campaign.

Table 11: **Exploratory KV-scale pilot: improving-candidate fraction (%).** Each entry is the fraction of sampled candidates above same-cell P_0 on selection rows; Image denotes the Image-matched profile.

Task	0.01	0.02	0.03	0.05	0.10	α_{eq}	Image-RMS
Qwen2.5-0.5B-Instruct							
USPTO50K	34.4	37.5	68.8	12.5	0.0	46.9	53.1
GSM8K	18.8	0.0	0.0	0.0	0.0	0.0	0.0
MBPP	18.8	9.4	0.0	0.0	0.0	3.1	12.5
Countdown	15.6	0.0	0.0	0.0	0.0	0.0	0.0
MATH500	65.6	0.0	0.0	0.0	0.0	0.0	0.0
OlympiadBench	78.1	0.0	0.0	0.0	0.0	0.0	15.6
ROCStories	90.6	87.5	56.2	3.1	0.0	84.4	90.6
Qwen2.5-3B-Instruct							
USPTO50K	3.1	21.9	21.9	43.8	81.2	18.8	43.8
GSM8K	25.0	31.2	40.6	62.5	0.0	56.2	34.4
MBPP	68.8	75.0	81.2	100.0	0.0	100.0	75.0
Countdown	21.9	9.4	18.8	3.1	0.0	15.6	3.1
MATH500	56.2	34.4	46.9	59.4	0.0	34.4	34.4
OlympiadBench	9.4	18.8	12.5	12.5	0.0	3.1	0.0
ROCStories	15.6	12.5	31.2	6.2	0.0	0.0	9.4
Qwen3-8B							
USPTO50K	31.2	15.6	12.5	6.2	6.2	6.2	12.5
GSM8K	34.4	50.0	62.5	34.4	25.0	18.8	50.0
MBPP	6.2	6.2	9.4	0.0	3.1	0.0	0.0
Countdown	21.9	28.1	31.2	34.4	43.8	46.9	28.1
MATH500	43.8	46.9	31.2	21.9	9.4	15.6	9.4
OlympiadBench	12.5	12.5	9.4	21.9	15.6	6.2	9.4
ROCStories	68.8	71.9	81.2	78.1	59.4	62.5	53.1
Llama-3.1-8B-Instruct							
USPTO50K	21.9	12.5	18.8	21.9	12.5	21.9	28.1
GSM8K	59.4	59.4	56.2	43.8	9.4	0.0	0.0
MBPP	78.1	75.0	75.0	46.9	37.5	3.1	3.1
Countdown	68.8	75.0	62.5	50.0	34.4	21.9	15.6
MATH500	40.6	53.1	31.2	59.4	59.4	62.5	31.2

Table continued from previous page

Task	0.01	0.02	0.03	0.05	0.10	α_{eq}	Image
OlympiadBench	75.0	81.2	84.4	78.1	68.8	56.2	81.2
ROCStories	84.4	87.5	75.0	59.4	9.4	0.0	0.0

Table 12: **Exploratory KV-scale pilot: score among improving candidates (%).** Values condition on a nonempty improving set; – marks cells without an improving candidate.

Task	0.01	0.02	0.03	0.05	0.10	α_{eq}	Image-RMS
Qwen2.5-0.5B-Instruct							
USPTO50K	26.0	25.6	26.1	24.2	–	26.0	26.5
GSM8K	46.7	–	–	–	–	–	–
MBPP	15.2	16.7	–	–	–	12.5	13.5
Countdown	7.0	–	–	–	–	–	–
MATH500	26.7	–	–	–	–	–	–
OlympiadBench	9.1	–	–	–	–	–	7.2
ROCStories	31.1	31.3	30.9	30.3	–	31.0	31.1
Qwen2.5-3B-Instruct							
USPTO50K	12.5	12.6	13.2	15.1	18.1	15.5	18.7
GSM8K	81.6	81.6	82.9	84.4	–	84.1	84.1
MBPP	33.5	38.2	44.6	59.6	–	53.6	48.0
Countdown	19.6	19.5	19.7	20.3	–	22.8	19.5
MATH500	63.9	64.3	64.2	63.6	–	64.8	64.4
OlympiadBench	38.3	38.8	38.5	37.9	–	39.1	–
ROCStories	61.0	60.8	61.2	61.0	–	–	61.3
Qwen3-8B							
USPTO50K	27.5	28.4	27.9	28.5	32.8	31.6	29.9
GSM8K	92.3	92.4	92.7	92.7	92.6	92.3	92.7
MBPP	80.5	80.9	80.5	–	80.5	–	–
Countdown	28.5	30.1	31.2	38.8	40.6	41.9	38.5
MATH500	79.5	79.8	79.8	79.6	78.9	79.4	79.7
OlympiadBench	55.7	55.5	55.5	55.9	55.5	56.2	55.5
ROCStories	78.4	78.6	78.8	79.0	79.3	79.4	79.0
Llama-3.1-8B-Instruct							
USPTO50K	17.2	17.4	17.4	17.6	18.6	22.5	22.0
GSM8K	85.8	86.5	85.9	85.8	87.0	–	–
MBPP	62.1	62.5	62.4	62.7	62.2	62.5	60.9
Countdown	14.2	14.6	13.7	14.2	14.1	17.5	14.5
MATH500	29.4	29.6	30.2	29.9	30.3	30.9	32.7
OlympiadBench	10.0	10.5	10.6	10.7	11.2	11.2	11.4
ROCStories	54.3	54.5	54.6	54.6	54.2	–	–

The cache-wide control used unnormalized Gaussian noise

$$\delta_j^{S,\ell} = \alpha \,\mathrm{rms}^{S,\ell} u_j^{S,\ell}, \qquad u_j^{S,\ell} \sim \mathcal{N}(0,1), \tag{9}$$

at every cached position. The paired ablation instead uses the normalized prefix-only sampler in eq. (2), reuses unit draws across conditions, sweeps five fixed scales, and adds an equal-total-norm control.

For the measured Image field Δ^W in eq. (4), let b range over all prefix-cache blocks (one layer and either the key or value stream), and let C_b be the corresponding clean-cache block. The equal-total-norm control uses a single fixed scale

$$\alpha_{\text{eq}} = \left(\frac{\sum_b \|\Delta_b^W\|_F^2}{\sum_b \|C_b\|_F^2} \right)^{1/2}. \tag{10}$$

Because the sampler in eq. (2) fixes each block's RMS, this scale matches the image-matched profile's total squared norm across the prefix blocks while allowing a different allocation among blocks. Mean score among improving candidates is conditional on exceeding P0 and undefined for an empty improving set; the fraction uses every draw.

P0 matches before and after every chunk. The first six cells were preregistered; the rest are exploratory. The shared $\alpha = 0.01$ was chosen after examining this exploratory grid, and the pilot has no fixed-scale held-out winner.

Two Qwen2.5-0.5B ROCStories studies share the metric and 128-row selection readout but use different prefixes. Their P0 scores are 29.94% and 100.00%; each study must be compared with its own P0.

G.1.2 Prefix-Length Sensitivity

This source-conditioned study crosses 1, 2, 4, and 8 shots with three nested content seeds per model (section H.2). W, rows, decoding, and draw IDs stay fixed; Image and its bank are recomputed per cell. Added shots also change content, so the study measures operating-point sensitivity, not token-count causality.

Qwen2.5-3B contracts with longer prefixes; Qwen3-8B has no monotone density trend. Its eight-shot Image mean is driven by one seed (-87.5 points versus $+3.1$ and -0.8 for the others). At $N = 128$, Qwen3's family-mean slope changes from $+0.33$ to -0.06 points per log token and density stays nearly flat. Llama-3.1-8B has zero density at both budgets despite a higher oracle maximum (fig. 10).

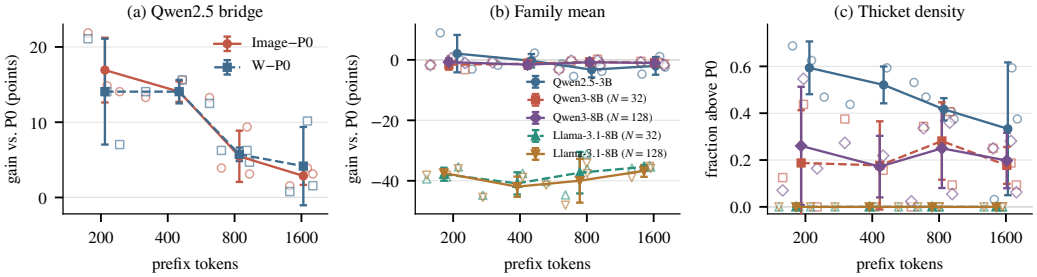


Figure 10: Prefix sensitivity across Qwen and Llama models. Open markers are content seeds; curves show mean±sd across the three seeds.

G.2 Initial Controls and Measured Expert Images

G.2.1 Initial Screening and KV-Space Discovery

Screening. On Qwen2.5-3B-Instruct and GSM8K, a cache-wide $\alpha \cdot \mathrm{rms}$ sweep over $\{0.01, 0.02, 0.03, 0.05, 0.10\}$ selected $\alpha^* = 0.03$ before opening the $n = 1000$ test split. W gained 10.9 points [8.3, 13.5]; the selected cache dose lost 5.4 points [-8.2, -2.5]. This unnormalized, cache-wide control differs from the normalized, prefix-only fixed-scale search in section 3.2; it is not a test of that method.

Discovering the KV intervention space. We measured each scored weight candidate's cache difference from the base on p^*. Four prefix-only fields shared its layer–stream RMS magnitudes: Image, a $K = 16$ covariance-induced draw, an isotropic draw, and a spectrum-matched shuffle. That covariance control is distinct from the within-block coordinate permutations in section F.1. On GSM8K, Image recovered the task gain, while an independently sampled isotropic draw also improved task score.

G.2.2 IMAGE SCALE CONTROLS

For each prefix layer–stream block b, let $r_b = \mathrm{RMS}(C_b)$ and $s_b = \mathrm{RMS}(\Delta_b^W)$. We report the clean-cache-energy-weighted relative magnitude $\alpha_{\text{eff}} = \sum_b w_b(s_b/r_b)$, where $w_b = r_b^2/\sum_{b'} r_{b'}^2$; α_K and α_V use the same average with weights renormalized within the key and value streams, respectively. For Image, $\alpha_{\text{eff}} = 0.038$ is near the failed cache-wide sweep's $\alpha^* = 0.03$, but that sweep edits every cached position whereas Image stops at the prefix. Image has $\alpha_V = 0.17$ versus $\alpha_K = 0.025$.

The five Image doses multiply its field energy by $\{0.25, 0.5, 1, 2, 4\}$, a $16\times$ range, by multiplying field amplitudes by the square roots of these factors. Across those doses, Image scores 0.828–0.852 on the same 128 GSM8K rows, versus 0.773 for same-session P0. For the spectrum-matched shuffled field described above, the paired 95% interval against P0 includes zero at every dose. Two other candidate images also beat P0.

H PROMPTS AND PREFIX CONSTRUCTION

This appendix reports the messages before model-specific chat-template serialization. Braced fields are replaced by the corresponding dataset item; paragraph breaks that affect the prompt structure are shown, while repeated line breaks in enumerations are compacted for print. Tasks without an explicit system message use a single user message. For inference, the model's native chat template is applied with a generation prompt. For prefix construction it is applied without a generation prompt, so the prefix ends at the final demonstration answer. We set `enable_thinking=False`; templates that ignore this argument are unchanged, while Qwen3 emits its registered non-thinking form.

H.1 TASK PROMPTS

Countdown. **System:** You are a helpful assistant. You first think about the reasoning process in your mind and then provide the user with the answer.

User: Using the numbers `{numbers}`, create an equation that equals `{target}`. You can use basic arithmetic operations `(+, -, *, /)` and each number can only be used once. Show your work in `<think> </think>` tags. And return the final answer in `<answer> </answer>` tags, for example `<answer> (1 + 2) / 3 </answer>`.

GSM8K. **User:** `{question}`. Let's think step by step and output the final answer after `####`.

MATH-500. **User:** `{problem}`

Let's think step by step and output the final answer after `####`

OlympiadBench. **User:** `{question}`

Let's think step by step and output the final answer after `####`.

MBPP. **System:** You are a Python programming assistant. Write clean, correct Python code to solve the given problem.

User: `{text}`

Your code should pass these tests:

`{tests}`

Think through your solution in `<think> </think>` tags.

Return your final Python code in `<answer> </answer>` tags, e.g.:

```
<answer>
def solution(x):
   return x + 1
```

</answer>

Here {tests} is the first three unit-test strings joined by newlines.

USPTO-50K. System: You are an expert organic chemist. Your task is to classify chemical reactions into one of 10 standard reaction categories based on the transformation type.

> **Reaction Classes:** 1: Heteroatom alkylation/arylation - N, O, S attacking C (e.g., SN2, ether formation); 2: Acylation - Forming C=O bonds with N, O, S (e.g., amide, ester formation); 3: C-C bond formation - New C-C bonds (e.g., Suzuki, Heck, Grignard); 4: Heterocycle formation - Creating rings with N, O, S; 5: Protections - Adding protecting groups (Boc, Bn, TBS, etc.); 6: Deprotections - Removing protecting groups; 7: Reductions - Adding H, removing O (e.g., ketone → alcohol, nitro → amine); 8: Oxidations - Adding O, removing H (e.g., alcohol → ketone); 9: Functional group interconversion - Changing one FG to another; 10: Functional group addition - Adding new FG to molecule (e.g., halogenation).

User: Classify this reaction:

Reactants >> Product:

{rxn_smiles}

Analyze the key transformation and output the class number (1–10) in <answer>X</answer> tags. Atom-mapping indices are removed from the reaction SMILES before insertion.

ROCStories. System: You are a helpful assistant that excels at story comprehension and logical reasoning. Given shuffled sentences from a story, you carefully analyze the narrative flow and temporal cues to determine the correct chronological order.

User: Below are 5 sentences from a story, but they are in the wrong order. Please arrange them in the correct chronological order.

Title: {title}

Sentence A: {sentence_1}

Sentence B: {sentence_2}

Sentence C: {sentence_3}

Sentence D: {sentence_4}

Sentence E: {sentence_5}

Output the correct order as comma-separated letters (e.g., B,A,D,E,C). Only output the letters, nothing else.

H.2 Few-Shot Prefix and Length-Study Design

The fixed task prefix p^* is a complete few-shot conversation rather than a separate instruction. For task d, shot count Ξ, and content seed s, we construct

$$p^*_{d,\Xi,s} = \mathrm{ChatTemplate}\big([m_{\mathrm{sys}}; u_{i_1}, a_{i_1}; \ldots; u_{i_K}, a_{i_\Xi}]\big), \tag{11}$$

where m_{sys} is included once when the task defines one, u_i is the task's user template above, and a_i is the dataset reference answer—never a model generation. GSM8K, MATH-500, OlympiadBench, and MBPP use their reference solution or code; ROCStories uses the gold letter order; USPTO-50K uses <answer>class</answer>; and Countdown uses a deterministically constructed valid equation. The evaluation item is appended after this fixed prefix as a new request and is not part of p^*.

Rows are sampled without replacement from eligible training examples using a fixed random seed. The eligible pool excludes the pilot head and every registered evaluation split. Selected row IDs are sorted before rendering. A prefix that exceeds its token budget causes the run to fail; it is never truncated.

Registered length grid. The study uses GSM8K, $\Xi \in \{1, 2, 4, 8\}$, content seeds 20260912–20260914, and Qwen2.5-3B, Qwen3-8B, and Llama-3.1-8B-Instruct. Within a seed, exemplar row sets are nested; sorting makes their rendering canonical but means the Ξ-shot token string need not be a literal prefix of the $(\Xi + 1)$-shot string. Across models, row IDs are paired, but tokenization remains model native, so analysis uses measured token length rather than shot count as a shared token budget. Within each model, W, evaluation rows, decoding, and candidate draw IDs remain fixed; the measured Image and random cache family are recomputed per cell. The $N = 128$ repetitions reuse the corresponding $N = 32$ prefixes and evaluation rows.