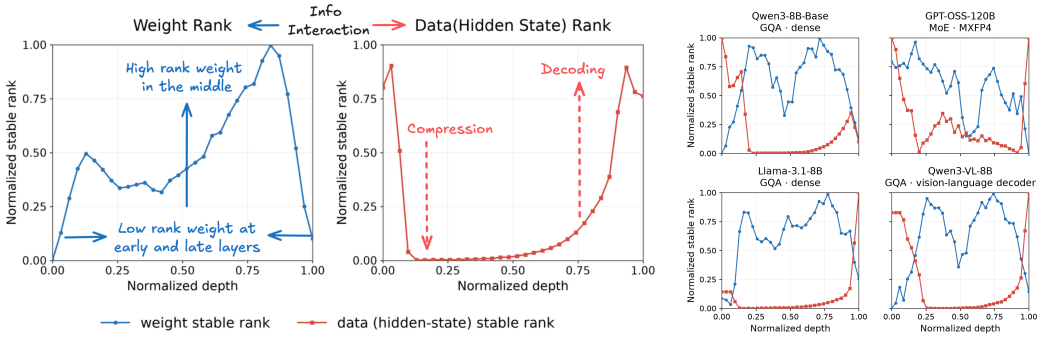
MODEL-AWARE DATA SELECTION FROM IN-AND-OUT INFORMATION INTERPLAY

Anonymous authors
Paper under double-blind review

ABSTRACT

LLMs are effective representations that assimilate vast amounts of knowledge during pretraining, but post-training is necessary for models to reliably access this knowledge and "know what they know." We observe an interesting rank equilibrium between knowledge stored in the weights and the data stream passing through the model. Across all model layers, we find that the hidden states (data stream) follow a U-shaped pattern, showing substantial compression in early layers and a steep rise during the late-layer decoding phase. In contrast, the weight rank follows an inverted U-shaped pattern, with very low rank in the early and late layers and high rank in the middle. We interpret this as an *in-and-out information interplay*: intermediate activations do not need to carry content that the weights can supply later, so they primarily preserve what the weights cannot provide. Motivated by this observation, we propose a model-aware data selection method, CAP (Counterfactual Assimilation Profile), which can determine whether a data candidate contains information accessible to the current model by utilizing the divergence gap in early- and late-layer representations between model-generated and reference responses. Across math, code, and science domains, CAP delivers **35.4%** greater average improvement over the base model than the strongest baseline under different selection budgets. With only **10%** of the data pool, CAP surpasses or matches full-pool training on math and science. We further show that CAP transfers to multimodal data selection and is robust to response horizon and noise.



(a) Rank equilibrium between weights and data. (b) Cross-model consistency.

Figure 1: **The in-and-out information interplay.** **(a)** Across model layers, weight stable rank (blue, circles) follows an inverted U-shape, while hidden-state stable rank (red, squares) follows the complementary U-shape: activations compress through the early and middle layers and re-expand near the output for decoding, as weight rank declines. **(b)** This complementary pattern recurs across model families, scales, and architectures; the full 12-model survey is in Appendix B.

1 INTRODUCTION

A modern language model acquires knowledge in two very different regimes. Pretraining assimilates a web-scale corpus into model weights (Delétang et al., 2024): it determines what is *stored*, but storage does not imply use, since a base model often fails at tasks whose ingredients are demonstrably

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.

represented in its parameters. Post-training changes much less about what is stored and much more about what is *accessible*: which internal competencies can be surfaced, in what format, and under what instruction. Figure 2 makes this distinction explicit: the latent set is mainly expanded through (continued) pretraining, while post-training primarily expands the accessible set and can occasionally introduce new knowledge, with some knowledge also lost through forgetting (Kirkpatrick et al., 2017).

Supervised fine-tuning (SFT) is the simplest and most widely used post-training stage, making the choice of SFT data a practical lever for improving accessibility. With strong open-weight base models, a few thousand curated examples can already produce competitive domain-specific assistants. This setting is particularly attractive in domains such as finance and medicine, where sensitive data often cannot leave the organization. Under limited compute and a large candidate pool of uneven utility, a small, well-chosen subset can outperform a much larger indiscriminate one (Zhou et al., 2023). Which examples to keep therefore becomes a central question.

This decision is difficult because the value of an example depends on the model. An example the model can already reproduce provides little new signal, while one far outside its competence may provide mostly noise. Yet, most existing selection signals largely ignore the underlying model. Learned quality raters (Wettig et al., 2024) score text in isolation, while diversity methods (Sener & Savarese, 2018) cover embedding space without asking what the model has already absorbed. Loss-based scores such as perplexity (Marion et al., 2023) and IFD (Li et al., 2024b) consider the model, but reduce its response to a scalar that cannot distinguish *knowledge I have not yet assimilated* from *text that is intrinsically difficult, or simply junk*.

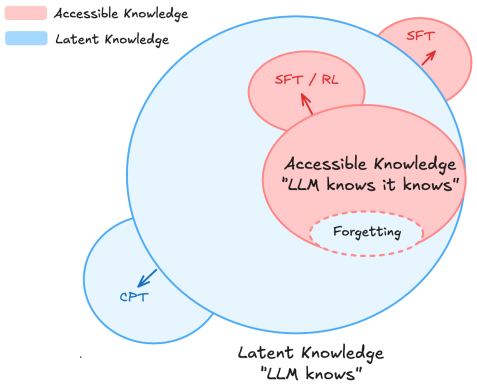


Figure 2: Latent and accessible knowledge.

We identify this boundary of accessible knowledge inside the model through a pattern in how information is distributed between weights and activations. Measured layer by layer, trained decoders place high-rank weights in the middle of the stack, while hidden states compress through the middle and re-expand near the output (Figure 1). Neither profile is present at initialization; both emerge over pretraining and recur across model families, scales, attention mechanisms, dense and mixture-of-experts computation, and text-only and vision-language decoders. We interpret this as an *in-and-out information interplay*: middle layers compress what comes *in* against what is already stored, while late layers re-expand as stored knowledge is read *out* for decoding. Deep layers are therefore where information brought by the data meets knowledge already stored in the weights.

This observation motivates our approach, *Counterfactual Assimilation Profile* (CAP). For a candidate (x, y), we run the model on the real pair $[x; y]$ and on the counterfactual pair $[x; \hat{y}]$, where $\hat{y}$ is the model's own generation, and measure the per-layer divergence between the two runs. Shallow-layer divergence captures surface mismatch such as format and style, while deep-layer divergence captures mismatch at locations where stored knowledge is retrieved. CAP residualizes deep divergence on shallow divergence, isolating information that the model has not yet assimilated, precisely the information SFT should make accessible. Structureless noise contains no retrievable content, so it provably falls to the bottom of the CAP ranking rather than the top, where loss-based scores may place it. Building on these insights, we make the following contributions:

1. **Phenomenon and theory.** We identify a coupled rank equilibrium between weights and hidden states that emerges over pretraining and holds across architectures, scales, and modalities. In a memory-augmented computation model, we show that the observed U-shaped and inverted-U-shaped profiles arise as optimal capacity profiles, obey a per-depth conservation law, and make the trough of the activation U-shape a measure of assimilated knowledge (Section 3).

2. **Performance and data efficiency.** Across math, code, and science and across selection budgets on Marin-8B-Base, CAP delivers 35.4% greater average improvement over the base

model than the strongest baseline, and nearly $1.9\times$ the gain of random selection. (Sections 4 and 5).

3. **Generality and robustness.** CAP transfers unchanged to a vision-language decoder and to finance and medicine, without requiring modality-specific scoring machinery. It remains effective across nearly an order of magnitude of response horizons and rejects injected noise that perplexity ranks above the entire clean distribution (Sections 4.4 and 5).

2 Related Work

Data selection for post-training. Work on choosing post-training data splits into roughly three directions, distinguished by where the signal comes from. *External scoring* judges an example without considering the current model: learned quality raters such as QuRating (Wettig et al., 2024), LLM-as-judge pipelines such as AlpaGasus (Chen et al., 2024), and complexity/quality composites such as DEITA (Liu et al., 2024). *Geometric and distributional* methods select for coverage or for match to a target distribution: k-center coreset selection (Sener & Savarese, 2018), DSIR-style importance resampling (Xie et al., 2023), and deduplication or clustering pipelines. *Model-internal* methods read the current target model: perplexity and loss-based pruning (Marion et al., 2023), instruction-following difficulty (Li et al., 2024b), and gradient-influence matching such as LESS (Xia et al., 2024). CAP belongs to the third family but differs from others in what it reads. Loss-based scores compress the model's state into a single likelihood number, and gradient methods require a target eval set and backward passes. CAP instead contrasts two forward passes, the reference response against the model's own continuation, and localizes the contrast by depth, which is what lets it separate unassimilated content from surface difficulty. That separation is also what our theory predicts, and what makes noise rejection provable rather than incidental.

Compression, rank, and structure across depth in LLMs and VLMs. Our observation sits in a line of work on the low-dimensional and highly structured nature of transformer computation. Language modeling as compression (Delétang et al., 2024) provides the framing; fine-tuning within a low intrinsic dimension (Aghajanyan et al., 2021) and low-rank adaptation (Hu et al., 2022) show that task-relevant updates occupy few directions; the low-rank simplicity bias of gradient descent (Huh et al., 2023) explains why; and post-training compression methods such as SparseGPT (Frantar & Alistarh, 2023) and Wanda (Sun et al., 2024b) exploit the resulting redundancy, empirically showing that layers differ sharply in how much compression they tolerate. Complementary phenomenological work shows that activations are far from generic: attention sinks concentrate mass on a few positions (Xiao et al., 2024) and massive activations dominate a handful of hidden dimensions (Sun et al., 2024a), both varying systematically with depth. Interpretability work in the same vein reads intermediate layers out directly, as in logit and tuned lenses (Belrose et al., 2023), and localizes factual recall to middle-layer MLPs (Meng et al., 2022; Dai et al., 2022), consistent with our finding that the interior is where stored knowledge is matched against the input. The same depth-wise structure appears in vision-language decoders (Deitke et al., 2024). What is new here is the *coupling*: weight rank and hidden-state rank form complementary profiles, the coupling strengthens over pretraining, and it can be converted into a practical, model-aware data selection signal.

3 In-and-Out Information Interplay: Rank Equilibrium Between Weights and Data

3.1 The Observation

We measure the stable rank $\mathrm{srank}(A) = \|A\|_F^2/\|A\|_2^2$ of weight matrices and token hidden-state matrices at each layer, normalized by dimension. Three facts emerge: (i) *The profiles are coupled:* middle layers hold high-rank weights and low-rank activations, while late layers reverse the pattern (Figure 1a). (ii) *The coupling is learned:* we observe that the complementary profiles emerge progressively over pretraining rather than being present at initialization. (iii) *The coupling is not model-specific:* the same qualitative structure appears across all models we examine. Figure 1b spans dense, mixture-of-experts, and vision-language decoders, while the full survey covers 12 models from 3.8B to 120B parameters (Figure 7). Activation compression and late re-expansion are the most consistent features; weight profiles are noisier but generally concentrate rank in the interior.

We interpret this as an interplay between what flows *in* through the activations and what is stored in the weights. Middle layers compress activations while stored information can be supplied later, and late layers re-expand as that information is retrieved for decoding. This suggests a model-aware signal for data selection: assimilated content should induce little deep-layer change relative to the model's own generation, while unassimilated content should induce substantially more. Shallow layers should primarily reflect surface mismatch. This motivates a counterfactual comparison, a depth split, and residualization of deep divergence on shallow divergence.

3.2 Theoretical Analysis

We formalize this equilibrium with a simple capacity model of computation against memory. The full model, assumptions, mapping to transformer measurements, and proofs are given in Appendix A.

A depth-L computation carries activation capacity r_ℓ and spends weight capacity w_ℓ at each layer. A memory holds K units of task-relevant knowledge, and a retrieval schedule k_ℓ determines when that knowledge is injected. Activations must carry content that will not be supplied later, together with an address for content that will be retrieved later. We assume that addressing stored content is cheaper than carrying it directly. The training compression bias is represented by minimizing $\sum_\ell r_\ell$, with total weight capacity as a tiebreaker.

Theorem 1 (Shape of the optimal profiles)**.** *Under Assumptions 1–3 (Appendix A.1), the optimal computation exists, is unique, and satisfies, up to one fractional boundary layer:*

- *(i)* ***Deferred retrieval:*** $K_{>\ell} = \min\{K, (L-\ell)B_r\}$*, so retrieval is pushed as late as the bandwidth B_r permits.*
- *(ii)* ***U-shaped activations:*** $r_\ell = \max\{H - B_t\ell,\ g(K_{>\ell})\}$ *descends at the maximal rate B_t from H to the trough $r^\star = g(K) = c(R-K) + \varphi(K)$, holds there, and rises back to cR at the output.*
- *(iii)* ***Interior-concentrated matching:*** *the matching component of w_ℓ vanishes at both boundaries and attains its maximum exactly under the activation trough, at a value strictly increasing in K; in the matching-dominated regime of Assumption 4, the full weight profile w_ℓ peaks in the interior.*

Parts (i) and (ii) arise because the computation compresses as quickly as permitted and delays retrieval until needed. Part (iii) additionally requires matching cost to grow with both the amount of stored content and the degree of compression, yielding an interior-peaked weight profile in the matching-dominated regime.

Theorem 2 (In-and-out conservation law)**.** *For every feasible computation and every depth ℓ, $r_\ell + c\,K_{>\ell} \geq cR$, and consequently $r_\ell + \frac{c}{\sigma}\sum_{j>\ell} w_j \geq cR$. On the retrieval arm of the optimal computation,*

$$r_\ell + c\,K_{>\ell} = cR + \varphi(K_{>\ell}).$$

Thus, at each depth, information carried by the activations plus information that can still be supplied from memory must meet the task requirement.

Corollary 1 (The trough is a knowledge meter)**.** *Let $\psi(K) = cK - \varphi(K)$. Then $\psi(0) = 0$, $\psi' \geq \eta > 0$, and*

$$cR - r^\star(K) = \psi(K).$$

Hence the trough deepens monotonically as retrievable knowledge grows, and at $K = 0$ the optimal profile is non-increasing in depth.

This is consistent with our observation that the coupled profiles emerge over pretraining and predicts that domain-specific continued pretraining should deepen the trough on in-domain data.

The assimilation gap. For a candidate (x, y), let K_y denote the storable content of the reference response and $K_y^{\cap} \leq K_y$ the part already retrievable from the weights given x. We define

$$G(x,y) \ :=\ \psi(K_y) - \psi(K_y^{\cap}) \ \geq\ 0. \tag{1}$$

The gap is zero both when the model already retrieves the relevant content of y and when y contains no storable structure. Thus G captures unassimilated but learnable content, which scalar loss alone cannot isolate.

From the gap to a measurable score. Run the model on $[x; y]$ and on the counterfactual $[x; \hat{y}]$, and let D_ℓ denote their divergence at depth ℓ (Section 4.1). The deferred-retrieval structure motivates an exclusion restriction: shallow divergence should primarily reflect surface mismatch, while deep divergence can additionally reflect the assimilation gap. We model

$$D_{\text{early}} = aS + \varepsilon_e, \qquad D_{\text{late}} = a'S + bG + \varepsilon_l,$$

where S is surface mismatch between the reference and the model's own generation. Assumption 5 defines the factor model and the resulting surface-orthogonal gap $G_\perp$.

Theorem 3 (Identification of the surface-orthogonal gap)**.** *Under Assumption 5, the population regression residual*

$$\text{CAP} = D_{\text{late}} - \alpha - \beta\, D_{\text{early}}$$

has slope

$$\beta = (1-\delta)(a' + b\gamma)/a$$

and satisfies

$$\text{CAP} = b\, G_\perp + \delta\,(a' + b\gamma)\,(S - \mathbb{E}S) - \beta\,\varepsilon_e + \varepsilon_l.$$

Consequently: (i) $\text{Cov}(\text{CAP}, G_\perp) = b\,\text{Var}(G_\perp) > 0$*; (ii) if* $\sigma_{\varepsilon_e} = 0$*, then* $\text{CAP} = b\, G_\perp + \varepsilon_l$*; and (iii) surface difficulty leaks into CAP only through the attenuation term* δ*.*

CAP therefore ranks by unassimilated content after removing the component predictable from surface difficulty.

Corollary 2 (Noise rejection)**.** *Let a contaminating example have* $G = 0$ *and surface value* s*, and suppose that it follows the same structural equations as the clean corpus. Then*

$$\mathbb{E}\left[\text{CAP} \mid S = s,\ G = 0\right] = \left[\,\delta\, a' - (1-\delta)\, b\gamma\,\right](s - \mathbb{E}S) - b\,\mathbb{E}G.$$

If $\gamma > 0$ *and* $\delta < b\gamma/(a' + b\gamma)$*, this expectation decreases with* s*, so high-surface-difficulty noise receives a low CAP score.*

This makes noise rejection a consequence of residualization rather than a particular distance metric. Proposition 1 further shows that top-m CAP selection retains sufficiently high-gap examples with failure probability exponentially small in the gap margin.

The theory characterizes the qualitative structure of the observed profiles and their connection to data selection. The activation U-shape and deferred retrieval schedule depend on the addressing and bandwidth assumptions, while the interior-peaked weight profile additionally requires the matching assumptions. Stable rank is a measurable proxy for capacity, and Theorem 3 is conditional on the factor model. The theory does not determine the empirical depth split, response horizon R, or downstream effect size; Appendix A.10 discusses these points in detail.

4 CAP: Counterfactual Assimilation Profile for Model-Aware Data Selection

4.1 Method

Inputs. For a candidate with prompt x and reference response y, we form the real sequence $[\,x; y\,]$ and the counterfactual $[\,x; \hat{y}\,]$, where $\hat{y}$ is the model's own greedy generation for the same prompt. Both responses are truncated to R tokens ($R = 48$ by default; ablated in Section 5.2). The generated response reflects what the model can produce from its current knowledge, so differences between the two runs isolate information introduced by the reference response.

Divergence and depth split. Let H_ℓ^{real} and H_ℓ^{gen} denote the layer-ℓ hidden states at aligned response positions in an L-layer model. Guided by the deferred-retrieval structure of Theorem 1, we summarize the per-layer divergence with shallow and deep averages:

$$D_\ell = 1 - \cos\!\big(H_\ell^{\text{real}},\, H_\ell^{\text{gen}}\big), \qquad D_{\text{early}} = \text{mean}_{\ell \le 0.33L}\, D_\ell, \qquad D_{\text{late}} = \text{mean}_{\ell \ge 0.66L}\, D_\ell, \quad (2)$$

with D_ℓ averaged over the R aligned response positions. Shallow divergence primarily captures surface mismatch, while deep divergence additionally reflects differences where stored knowledge is retrieved.

Method	Signal family	Signal
Random	Heuristic	none
PPL (Marion et al., 2023)	Model-internal (loss)	absolute model loss
IFD (Li et al., 2024b)	Model-internal (difficulty)	cond. vs. uncond. loss ratio
QuRating (Wettig et al., 2024)	External scoring	learned quality rater
LESS (Xia et al., 2024)	Model-internal (gradient)	target-set gradient influence
k-center (Sener & Savarese, 2018)	Geometric coverage	embedding geometry
CAP (ours)	Model-internal (repr.)	depth-split counterfactual divergence

Table 1: Data-selection baselines spanning heuristic, model-internal, external-scoring, and geometric signals.

Score and selection. Surface difficulty can inflate both quantities, so we residualize:

$$\mathrm{CAP} = D_{\mathrm{late}} - (\alpha + \beta\, D_{\mathrm{early}}), \tag{3}$$

where α and β are fit by ordinary least squares of D_{late} on D_{early} over the candidate pool. By Theorem 3, CAP captures the component of deep divergence not explained by surface difficulty, corresponding to the surface-orthogonal assimilation gap up to scale and noise. We z-score CAP within each domain and select the highest-scoring examples under the budget. Scoring requires one greedy generation and two forward passes per candidate, with no auxiliary rater, backward pass, or target set.

4.2 Experimental Setup

We use Marin-8B-Base (Marin Community, 2025) for all key experiments. Marin is developed fully in the open: its training code, checkpoints, training process, and the sources of its complete pretraining corpus are publicly released. This lets us construct SFT candidate pools from data sources outside its pretraining mixture, substantially reducing pretraining contamination as a confounder in evaluating data selection.

We fine-tune Marin-8B-Base with full-parameter SFT for one epoch using AdamW, a peak learning rate of 1×10^{-5}, a cosine schedule, and a 2048-token context. Every method shares the same candidate pool, budget, data order, seed, optimizer, and training configuration, so only the selected subset differs (Appendix H).

Each domain uses a deduplicated pool of 80k examples. *Math* selects from NuminaMath-CoT (Li et al., 2024a) and evaluates GSM8K strict-match accuracy (Cobbe et al., 2021); *code* selects from OpenCoder opc-sft-stage2 (Huang et al., 2024) and evaluates HumanEval pass@1 (Chen et al., 2021); and *science* selects from the StackExchange split of WebInstructSub (Yue et al., 2024) and evaluates MMLU-Pro STEM with 5-shot CoT (Wang et al., 2024). Baselines span the major signal families in Section 2: random selection, perplexity (PPL; Marion et al., 2023), IFD (Li et al., 2024b), QuRating (Wettig et al., 2024), gradient-based LESS (Xia et al., 2024), and k-center (Sener & Savarese, 2018).

4.3 Main Results

	Math (GSM8K)				**Code** (HumanEval)				**Science** (MMLU-Pro-5)				**Avg. @8k**
Method	1k	2k	4k	8k	1k	2k	4k	8k	1k	2k	4k	8k	
Base model	61.11				30.49				32.55				41.38
Random	**63.99**	65.88	68.01	69.37	28.05	**32.93**	32.32	30.49	33.89	35.50	37.57	38.19	46.02 (+4.63)
PPL	62.02	64.97	65.88	68.92	30.49	28.05	28.05	32.93	**34.83**	34.87	36.41	37.72	46.52 (+5.14)
IFD	61.41	62.70	65.05	66.34	25.00	28.66	26.83	28.66	33.85	34.56	37.48	38.27	44.42 (+3.04)
QuRating	61.11	63.76	67.78	66.11	29.27	28.66	31.71	32.93	34.14	35.06	**38.76**	38.51	45.85 (+4.47)
LESS	60.27	64.29	68.16	70.81	**31.71**	32.32	**33.54**	**35.37**	34.57	36.03	36.59	38.39	48.19 (+6.81)
k-center	63.31	64.44	66.11	70.58	30.49	27.44	29.27	32.32	33.78	35.07	36.29	37.08	46.66 (+5.28)
CAP (ours)	62.32	**67.78**	**74.30**	**76.04**	**31.71**	29.88	**33.54**	**35.37**	34.66	**36.23**	37.26	**39.05**	**50.15 (+8.77)**

Table 2: Accuracy after SFT on subsets of the 80k-example pools. The final column reports the macro-average accuracy at the 8k budget, with improvement over the base model shown in parentheses. **Best** and second best are marked per column. Results with standard deviations are in Appendix C.

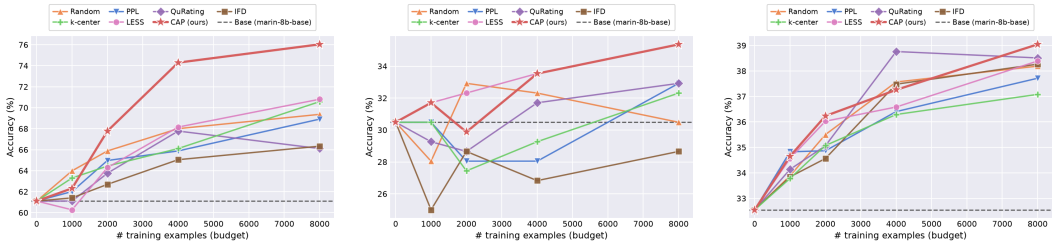


Figure 3: Mean accuracy versus selection budget on math (left), code (middle), and science (right). Shaded regions show one standard deviation.

Across math, code, and science and across all four selection budgets, CAP provides the strongest overall performance. Averaged over all 12 domain-budget settings, its improvement over the base model is 35.4% larger than that of LESS, the strongest single baseline on average. CAP wins or ties for first in 8 of the 12 settings (Table 2, Figure 3).

At the 8k budget, CAP gains an average of $+8.77$ points over the base model, compared with $+6.81$ for LESS and $+4.63$ for random selection. Thus, at this budget, CAP achieves nearly $1.9\times$ the gain of random selection. On math, CAP reaches $+14.93$ points over the base model, compared with $+9.70$ for the strongest baseline, with the margin growing to more than five points at 4k and 8k. On code, CAP ties LESS at 8k, while IFD ends below the base model, illustrating that difficulty alone is not a reliable measure of training value. Science is the most competitive domain: CAP leads at 2k and 8k, while PPL and QuRating lead at 1k and 4k, respectively.

Data efficiency relative to full-pool training. With only 8k examples, or 10% of the candidate pool, CAP reaches 76.04 on GSM8K, exceeding full-pool training at 72.48. On science, the same 10% subset reaches 39.05, within one point of full-pool training at 39.82. The 8k CAP models also outperform Marin-8B-Instruct on both math and science, despite the instruct model being trained on a substantially broader mixture. Code is the exception: full-pool training reaches 43.29, indicating that this domain benefits more from data volume. Even there, CAP maintains a consistent advantage over random and PPL as the budget increases to 60k examples (Appendix F).

4.4 Generalizability: Multimodal and Further Domains

Because CAP uses only hidden states and the model's own generation, it does not assume text-only inputs. Several baselines do: QuRating relies on a text-trained rater, IFD's unconditioned loss is not directly defined when the instruction contains an image, and k-center requires an embedding space that represents the visual component.

We apply CAP unchanged, including $R = 48$, to the *pretrain* checkpoint of Molmo2-O-7B (Clark et al., 2026), ensuring that the target domain is not already part of its post-training mixture. The candidate pool contains 20k deduplicated document-understanding examples from Molmo2-SynMultiImageQA (Appendix H). CAP trails random at 1k, where all methods operate in a high-variance regime, but leads from 2k onward. At 8k it reaches 46.15, outperforming the runner-up by 4.39 points and random selection by 5.49 points (Table 3).

Method	1k	2k	4k	8k
Base (no SFT)		0.00		
Random	**30.77**	29.12	35.71	40.66
PPL	26.92	32.97	35.71	41.76
CAP (ours)	26.92	**34.62**	**40.66**	**46.15**

Table 3: Multimodal selection on Molmo2-O-7B: held-out document-split accuracy from a 20k candidate pool.

Further specialized domains (finance, medicine). We further evaluate CAP on finance and medicine to test whether its model-aware signal remains effective in more specialized domains. CAP transfers unchanged to both settings and leads on every reported metric, achieving a 13.8% relative improvement over PPL on finance and an 8.2% improvement on medicine. The gains span both domain knowledge and reasoning-oriented evaluations (Appendix E).

5 Analysis and Ablation

5.1 Selection Cost versus Performance

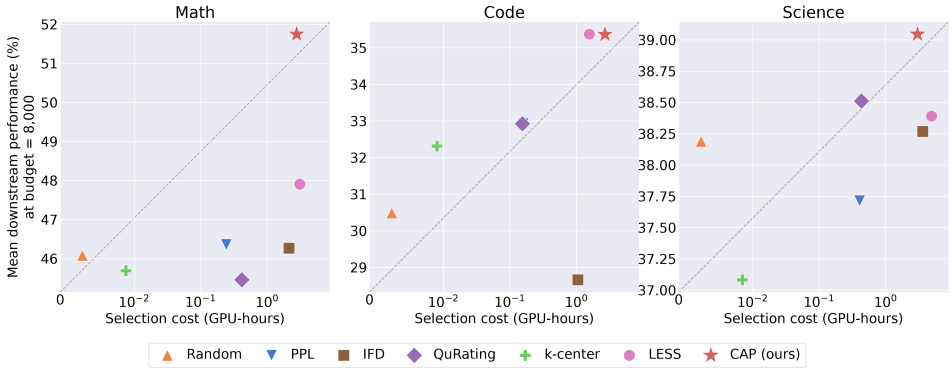


Figure 4: Accuracy at the 8k budget versus selection cost in utilization-weighted GPU-hours. The symmetric log scale retains the zero-cost random baseline.

CAP requires one greedy generation and two forward passes per candidate, with no auxiliary rater training, backward pass, or target set. Figure 4 places the methods on the cost–performance plane at the 8k budget, with cost measured in utilization-weighted GPU-hours to discount allocated time during which GPUs are idle (Appendix D). Averaged across math, code, and science, CAP uses 2.814 GPU-hours, compared with 3.175 for LESS and 2.259 for IFD. CAP is therefore 11.4% cheaper than LESS but 24.6% more expensive than IFD, and is also more expensive than PPL, QuRating, and k-center. This additional scoring cost is offset by stronger data efficiency: random selection requires 8k examples to reach 69.37 on GSM8K, whereas CAP reaches the same level at roughly 2.5k examples by interpolation between its 2k and 4k results, corresponding to a 3.2× reduction in selected data.

5.2 Ablation: The Response-Length Parameter R

Beyond the depth split, CAP has one numerical hyperparameter: the response horizon R over which the real and counterfactual runs are compared. Table 4 reruns the full pipeline at three horizons spanning nearly an order of magnitude. Every horizon outperforms random selection in all three domains, while CAP exceeds the strongest baseline in six of the nine horizon-domain settings and matches it in one more. The default $R = 48$ performs best on math and code, exceeding the strongest math baseline by 5.23 points and matching LESS on code. Science instead continues to improve with a longer horizon, reaching 39.72 at $R = 144$. Short horizons may miss reasoning expressed later in the response, while long horizons can include divergence caused by accumulated generation differences rather than assimilation. We therefore use $R = 48$ as the balanced shared setting across domains.

R	Math	Code	Science
16	71.95	32.32	38.88
48	**76.04**	**35.37**	39.05
144	72.93	31.71	**39.72**
Random	69.37	30.49	38.19
Best baseline	70.81	35.37	38.51

Table 4: Effect of response horizon R at the 8k budget.

5.3 Noise Rejection

High loss can arise for two different reasons: an example may contain useful knowledge the model has not yet assimilated, or it may simply be difficult because it lacks learnable structure. Difficulty-based scores cannot distinguish these cases, since both can produce high loss. To test whether CAP separates them, we inject 5% random junk into a clean pool and score the resulting mixture with Marin-8B-Base. Perplexity ranks the junk above the entire clean distribution, whereas CAP ranks it below 99.7% of clean examples (Figure 5). This behavior is consistent with Corollary 2: structureless noise has no assimilation gap, so although it appears difficult at the surface level, its deep divergence

falls below what the corpus regression predicts from that difficulty. CAP therefore suppresses such contamination without requiring a separately curated clean reference set, while difficulty-based ranking can assign it high priority.

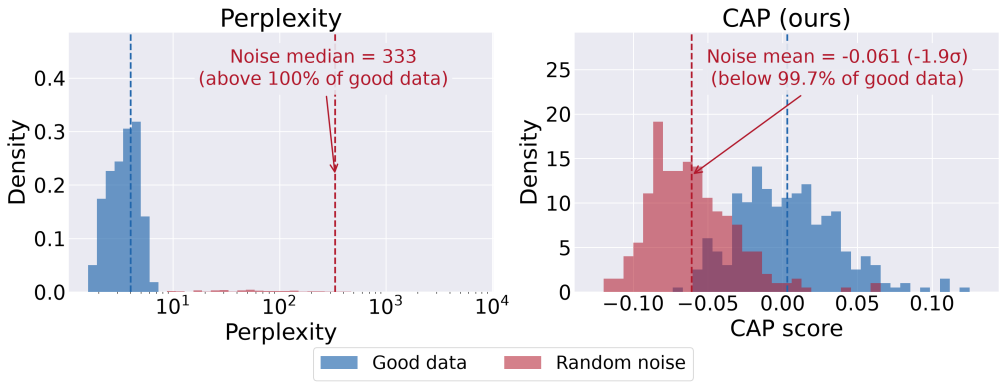


Figure 5: CAP separates unassimilated information from unstructured difficulty. We inject 5% random junk into a clean pool and score 300 clean and 300 noise examples with Marin-8B-Base. Left: perplexity ranks the noise above the clean distribution. Right: CAP ranks the same noise below 99.7% of clean examples.

5.4 The Optimization Path

Selection can improve performance either by changing the final solution or by changing the route taken during optimization. We project the 8k math SFT trajectories of CAP and random selection onto the top two principal components of their joint parameter path and evaluate them over a shared held-out GSM8K loss surface. In this projection, the two runs converge to distinct regions rather than different points along the same trajectory. CAP descends more quickly and reaches a lower final held-out loss (0.572 versus 0.584), while random selection begins to degrade late in training (Figure 6; full version in Appendix G). The difference appears early in training and persists through convergence, suggesting that CAP changes the optimization signal rather than merely improving the final subset average. While the two-dimensional projection cannot establish distinct basins in the full parameter space, it provides complementary evidence that model-aware selection affects not only final performance but also the training dynamics that lead to it.

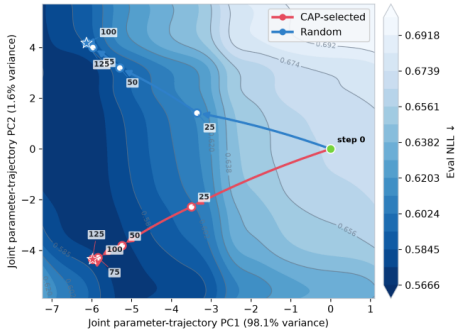


Figure 6: Optimization trajectories under CAP and random selection. Joint-trajectory PCA projection over a shared held-out eval loss surface. The full figure, including held-out loss versus optimizer step, is in Appendix G.

6 Conclusion

We identify an in-and-out information interplay in which weight and hidden-state ranks form complementary profiles across layers, and use this structure to define the *assimilation gap*: information the model has not yet made accessible. This motivates CAP, a lightweight model-aware selection score based on one counterfactual generation and two forward passes. Across math, code, and science, CAP delivers 35.4% greater average improvement than the strongest baseline, surpasses full-pool training on math with 10% of the data, and transfers to multimodal and specialized domains. CAP also remains robust across response horizons and rejects unstructured noise that difficulty-based selection can prioritize. Overall, our results suggest that effective post-training data should target what the model has not yet assimilated, rather than what is merely difficult, providing a model-aware view of data value for post-training.

Ethics Statement

All authors have read and adhere to the ICLR Code of Ethics. This work does not involve human subjects or personal data; all models, datasets, and benchmarks are publicly available and used under their respective licenses.

Reproducibility Statement

The theoretical model, assumptions, and complete proofs are given in Section 3.2 and Appendix A. The CAP score is specified in Section 4.1, and the scoring, training, pool construction, baseline, and evaluation details are in Section 4.2 and Appendix H. Full results with standard deviations and cost measurements are reported in Appendices C and D.

AI Use Statement

We used generative AI tools to assist with part of the experiment code and to edit the paper for readability. Generative AI was not used for the theory, proofs, research ideas, experimental design, or interpretation of results. The authors have reviewed all AI-assisted content and take full responsibility for this paper.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the Association for Computational Linguistics*, 2021.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. AlpaGasus: Training a better alpaca with fewer data. In *International Conference on Learning Representations*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Rohun Tripathi, Sangho Lee, Mohammadreza Salehi, Jason Ren, Chris Dongjoo Kim, Yinuo Yang, et al. Molmo2: Open weights and data for vision-language models with video understanding and grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28652–28668, 2026.

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, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Proceedings of the Association for Computational Linguistics*, 2022.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. *arXiv preprint arXiv:2409.17146*, 2024.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression. In *International Conference on Learning Representations*, 2024.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 2023.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Siming Huang, Tianhao Cheng, J. K. Liu, Jiaran Hao, Liuyihan Song, Yang Xu, J. Yang, J. H. Liu, Chenchen Zhang, Linzheng Chai, et al. OpenCoder: The open cookbook for top-tier code large language models. *arXiv preprint arXiv:2411.04905*, 2024.

Minyoung Huh, Hossein Mobahi, Richard Zhang, Brian Cheung, Pulkit Agrawal, and Phillip Isola. The low-rank simplicity bias in deep networks. *Transactions on Machine Learning Research*, 2023.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. FinanceBench: A new benchmark for financial question answering. *arXiv preprint arXiv:2311.11944*, 2023.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, et al. NuminaMath: The largest public dataset in AI4Maths with 860k pairs of competition math problems and solutions. `https://huggingface.co/datasets/AI-MO/NuminaMath-CoT`, 2024a.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In *Proceedings of NAACL*, 2024b.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *International Conference on Learning Representations*, 2024.

Marin Community. Marin: An open lab for building foundation models. `https://marin.community`, 2025. Marin-8B-Base and Marin-8B-Instruct models.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. When less is more: Investigating data pruning for pretraining LLMs at scale. *arXiv preprint arXiv:2309.04564*, 2023.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Sujet AI. Sujet-finance-instruct-177k. `https://huggingface.co/datasets/sujet-ai/Sujet-Finance-Instruct-177k`, 2024.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *Conference on Language Modeling*, 2024a.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, 2024b.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. QuRating: Selecting high-quality data for training language models. In *International Conference on Machine Learning*, 2024.

Kevin Wu, Eric Wu, Rahul Thapa, Kevin Wei, Angela Zhang, Arvind Suresh, Jacqueline J Tao, Min Woo Sun, Alejandro Lozano, and James Zou. Medcasereasoning: Evaluating and learning diagnostic reasoning from clinical case reports. *arXiv preprint arXiv:2505.11733*, 2025.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning*, 2024.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. Data selection for language models via importance resampling. In *Advances in Neural Information Processing Systems*, 2023.

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. MAmmoTH2: Scaling instructions from the web. In *Advances in Neural Information Processing Systems*, 2024. WebInstruct dataset.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less is more for alignment. In *Advances in Neural Information Processing Systems*, 2023.

A Formal Model, Assumptions, and Proofs

A.1 The Memory-Augmented Computation Model

Definition 1 (Memory-augmented layered computation). *A computation of depth L consists of activation capacities $r_0, \dots, r_L \geq 0$, weight capacities $w_1, \dots, w_L \geq 0$, and a retrieval schedule $k_1, \dots, k_L \geq 0$ with $\sum_\ell k_\ell = K$, where $0 \leq K \leq R$ is the total knowledge retrievable from memory for the task. Write $K_{>\ell} = \sum_{j>\ell} k_j$. The task requires delivering R units of predictive information at the output from an input carrying $H \geq cR$ units. The parameters are a content rate $c > 0$, an addressing function φ, a storage rate $\sigma > 0$, a transport rate $\tau \geq 0$, a matching rate $\sigma_m > 0$, a transport bandwidth $B_t > 0$, and a retrieval bandwidth $B_r > 0$. A computation is* feasible *if*

(C1) $r_0 = H$ *and* $r_L \geq cR$ *(boundary);*

(C2) $|r_\ell - r_{\ell-1}| \leq B_t$ *for all* ℓ *(transport bandwidth);*

(C3) $0 \leq k_\ell \leq B_r$ *for all* ℓ *(retrieval bandwidth);*

(C4) $r_\ell \;\geq\; g(K_{>\ell}) \;:=\; c\,(R - K_{>\ell}) + \varphi(K_{>\ell})$ *for all* ℓ *(information budget);*

(C5) $w_\ell \;\geq\; \underbrace{\sigma k_\ell}_{\text{injection}} + \underbrace{\tau\,|r_\ell - r_{\ell-1}|}_{\text{transport}} + \underbrace{m_\ell}_{\text{matching}}$, *where* $m_\ell := \sigma_m\,\mu(K_{>\ell})\,\nu(H - r_\ell)$ *(weight cost).*

An optimal *computation minimizes $\sum_{\ell=1}^{L} r_\ell$ over all feasible (r, k, w) and, among the minimizers, minimizes $\sum_{\ell=1}^{L} w_\ell$.*

The budget (C4) is the heart of the model: at depth ℓ the activations must carry the content that no later layer will inject, $c(R - K_{>\ell})$, plus an address for the content that later layers will inject, $\varphi(K_{>\ell})$. The primary objective encodes the compression bias of training, namely that representations are as small as the constraints allow; the weight tiebreaker encodes parameter economy, which is second-order and is used only to select a minimal w once (r, k) is fixed.

Assumption 1 (Addressing gain). *$\varphi : [0, K] \to \mathbb{R}_{\geq 0}$ is concave, non-decreasing, differentiable, $\varphi(0) = 0$, and $\varphi'(0) \leq c - \eta$ for some $\eta > 0$.*

Assumption 1 states that pointing at stored content is strictly cheaper than carrying it, which is the defining property of a memory. Concavity gives $\varphi'(x) \leq \varphi'(0) < c$ everywhere, so g in (C4) is strictly decreasing with slope at most $-\eta$: every unit of retrievable knowledge lowers the information the activations must carry. Since $K \leq R$ we also have $g \geq 0$ throughout.

Assumption 2 (Sufficient depth). *$L \;\geq\; \big(H - g(K)\big)/B_t + \lceil K/B_r \rceil + 1$ and $c\,B_r \leq B_t$.*

Assumption 3 (Localized matching cost). *$\mu, \nu : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ are non-decreasing with $\mu(0) = \nu(0) = 0$, and at least one of them is strictly increasing.*

The reading is that converting a compressed activation into an address requires the layer to supply the structure that compression discarded: the cost grows with the number of undelivered items to be discriminated, $K_{>\ell}$, and with how far the activation has been compressed below the raw input, $H - r_\ell$. This is the substantive modeling input behind the weight profile. Nothing below depends on the functional form beyond monotonicity; we use the bilinear case $\mu(K') = K'$, $\nu(u) = u/H$ only to state a closed-form ratio.

Assumption 4 (Matching-dominated regime). *$\sigma_m\,\mu(K)\,\nu(H - r^\star) \;>\; \big(\sigma + \tau c\big)B_r$, where $r^\star = g(K)$.*

Assumption 4 says that indexing the store costs more per layer than reading one bandwidth-unit out of it. It is the condition under which the model predicts the interior-peaked weight profile of Figure 1. When it fails, the model predicts the opposite, a weight profile peaking at the readout boundary, which makes the prediction falsifiable rather than automatic.

A.2 The Semantic Factor Model

Assumption 5 (Semantic factor model)**.** *Over the corpus, $(S, G, \varepsilon_e, \varepsilon_l)$ have finite second moments with $\mathrm{Var}(S) = \sigma_S^2 > 0$, and:*

- *(F1)* Exclusion. *G does not load on D_{early}. This is inherited from Theorem 1(i): the shallow block lies strictly before the first retrieval layer $t_r = L - \lceil K/B_r \rceil$, where the memory channel has injected nothing.*
- *(F2)* Linearity. *$D_{\mathrm{early}} = a\,S + \varepsilon_e$ and $D_{\mathrm{late}} = a'S + b\,G + \varepsilon_l$ with $a, a', b > 0$.*
- *(F3)* Noise. *$\varepsilon_e, \varepsilon_l$ are mean-zero and uncorrelated with each other and with (S, G).*
- *(N)* Contaminant homogeneity *(used only in Corollary 2). Contaminating examples obey (F2) with the same loadings a, a' and with $G = 0$.*

Define the surface-gap coupling $\gamma = \mathrm{Cov}(S, G)/\sigma_S^2$, the surface-orthogonal gap $G_\perp = (G - \mathbb{E}G) - \gamma\,(S - \mathbb{E}S)$, and the attenuation $\delta = \sigma_{\varepsilon_e}^2/(a^2\sigma_S^2 + \sigma_{\varepsilon_e}^2) \in [0, 1)$.

Only (F1) is supplied by the capacity model; (F2), (F3) and (N) are modeling assumptions about the corpus and the measurement. Note also that γ and δ are not identified from $(D_{\mathrm{early}}, D_{\mathrm{late}})$ alone, since (F2) gives a single equation in the latent G and the loadings a', b. The sufficient conditions of Corollary 2 are therefore not directly measurable; Section 5.3 tests its observable implication instead.

Proposition 1 (Selection consistency)**.** *Assume in addition that S is independent of $G_\perp$, and that $\nu_i := \mathrm{CAP}_i - b\,G_{\perp,i}$ are independent across examples and sub-Gaussian with proxy ς^2 conditionally on the gaps. Then for any two examples with $G_{\perp,i} > G_{\perp,j} + \Delta$,*

$$\Pr\left[\mathrm{CAP}_i < \mathrm{CAP}_j\right] \;\le\; \exp\!\left(-\frac{b^2\Delta^2}{4\varsigma^2}\right),$$

and top-m selection by CAP retains every example whose gap exceeds the m-th largest gap in the pool by margin Δ, except with probability at most $n\,e^{-b^2\Delta^2/(4\varsigma^2)}$ over a pool of size n.

The independence of S and $G_\perp$ is strictly stronger than the uncorrelatedness that defines $G_\perp$, and is needed because ν contains the residual surface term $\delta(a' + b\gamma)(S - \mathbb{E}S)$ of Theorem 3. It holds automatically if (S, G) is jointly Gaussian.

A.3 Mapping the Model to Transformers

Three modeling choices connect Definition 1 to the measurements of Section 3.1.

(M1) *Activation capacity is measured by hidden-state stable rank.* For representations with approximately isotropic per-direction energy, log-volume, and hence transmissible information, scales with the number of directions carrying non-negligible energy, of which srank is the robust surrogate. The approximation degrades for strongly anisotropic spectra, which is one reason we read the profiles qualitatively.

(M2) *Weight capacity is measured by weight stable rank*, read as the number of directions a layer can read from or write to. This is a coarser proxy than (M1), since $\mathrm{srank}(W)$ conflates the injection, transport and matching components of w_ℓ into one number; it is consistent with the empirical observation that weight profiles are noisier than activation profiles.

(M3) *The compression objective is the implicit bias of training*, namely the information-bottleneck tendency of deep representations combined with the low-rank bias of gradient descent (Huh et al., 2023).

(M4) *The factor model* takes D_ℓ as a capacity-weighted representational divergence and the depth split $0.33L/0.66L$ as a conservative estimate of the phase boundaries ℓ_d and t_r of Theorem 1.

A.4 Proof of Theorem 1

Step 0: reduction to (r, k). The weight variables appear only in (C5), as a lower bound, and only in the tiebreaker. Hence for any (r, k) satisfying (C1)–(C4) there is a feasible w, namely (C5) with

equality, and the lexicographic optimum is obtained by first solving for (r, k) and then setting w to that value. It therefore suffices to minimize $F(r, k) = \sum_{\ell=1}^{L} r_\ell$ subject to (C1)–(C4).

Existence. The feasible set is nonempty (the profile of Step 2 is feasible under Assumption 2), closed, and restricting to $r_\ell \le H$ is without loss of optimality, so the feasible set may be taken compact; the objective is continuous, so a minimizer exists.

Step 1: pointwise lower bound for a fixed schedule. Fix k. By (C1) and induction on (C2), $r_\ell \ge H - B_t\ell$. Combining with (C4),

$$r_\ell \;\ge\; \underline{r}_\ell(k) \;:=\; \max\{\, H - B_t\ell,\; g(K_{>\ell})\,\}.$$

Step 2: the bound is attained. Set $r_\ell = \underline{r}_\ell(k)$. Then (C4) holds by construction, and (C1) holds since $\underline{r}_0 = \max\{H, g(K)\} = H$ (because $g(K) \le g(0) = cR \le H$) and $\underline{r}_L \ge g(K_{>L}) = g(0) = cR$. For (C2), note that for any two sequences f, h,

$$\big|\max(f,h)_\ell - \max(f,h)_{\ell-1}\big| \;\le\; \max\big\{|f_\ell - f_{\ell-1}|,\; |h_\ell - h_{\ell-1}|\big\}.$$

The descent sequence has increments of size exactly B_t, and on the g sequence, by the mean value theorem and Assumption 1,

$$g(K_{>\ell-1}) - g(K_{>\ell}) = \big(c - \varphi'(\xi_\ell)\big)k_\ell \in [0,\, c\,k_\ell] \subseteq [0,\, cB_r] \subseteq [0,\, B_t],$$

using (C3) and Assumption 2. Hence $\underline{r}(k)$ is feasible, and by Step 1 it is the unique optimal profile for the schedule k.

Step 3: the optimal schedule is uniquely the maximally deferred one. It remains to minimize $F(k) := \sum_\ell \underline{r}_\ell(k)$. For any feasible k and any ℓ, (C3) gives $K_{>\ell} = \sum_{j>\ell} k_j \le (L-\ell)B_r$, and trivially $K_{>\ell} \le K$, so

$$K_{>\ell} \;\le\; K^\star_{>\ell} \;:=\; \min\{K,\; (L-\ell)B_r\}.$$

The bound is attained pointwise by the schedule $k^\star_\ell = K^\star_{>\ell-1} - K^\star_{>\ell}$, which satisfies $0 \le k^\star_\ell \le B_r$ and telescopes to $K^\star_{>0} - K^\star_{>L} = K$ under Assumption 2. Since g is strictly decreasing, $\underline{r}(k^\star) \le \underline{r}(k)$ pointwise for every feasible k, so $k^\star$ is optimal.

For uniqueness, write $\ell_d := \lceil (H - r^\star)/B_t \rceil$, so that Assumption 2 gives $\ell_d \le L - \lceil K/B_r \rceil$. Let k be optimal. For $\ell \ge \ell_d$ we have $H - B_t\ell \le r^\star = g(K) \le g(K^\star_{>\ell}) \le g(K_{>\ell})$, so both $\underline{r}_\ell(k)$ and $\underline{r}_\ell(k^\star)$ lie on the g branch; if $K_{>\ell} < K^\star_{>\ell}$ for some such ℓ, then $\underline{r}_\ell(k) > \underline{r}_\ell(k^\star)$ while $\underline{r}(k^\star) \le \underline{r}(k)$ everywhere else, giving $F(k^\star) < F(k)$ and contradicting optimality. Hence $K_{>\ell} = K^\star_{>\ell}$ for all $\ell \ge \ell_d$. For $\ell < \ell_d$ we have $K^\star_{>\ell} = K$ and $K \ge K_{>\ell} \ge K_{>\ell_d} = K^\star_{>\ell_d} = K$, so equality holds there too. Since k is determined by $\ell \mapsto K_{>\ell}$, we get $k = k^\star$, which is (i).

Step 4: the activation profile. Substituting $K^\star_{>\ell}$ into $\underline{r}(k^\star)$ gives (ii): a descent at rate B_t from H to the trough $r^\star = g(K) = c(R-K) + \varphi(K)$ reached at ℓ_d, a flat segment on $[\ell_d,\, \ell_r]$ with $\ell_r := L - K/B_r$, and an ascent along $g(K^\star_{>\ell})$ to $g(0) = cR$. The segment $[\ell_d, \ell_r]$ is nonempty by Assumption 2. The profile is quasi-convex as the pointwise maximum of a non-increasing and a non-decreasing function of ℓ.

Step 5: the weight profile. By Step 0, w_ℓ equals the right-hand side of (C5) at the optimum. Write $m_\ell = \sigma_m\, \mu(K^\star_{>\ell})\, \nu(H - r_\ell)$. On the descent, $k_\ell = 0$, $|r_\ell - r_{\ell-1}| = B_t$, $K^\star_{>\ell} = K$ and $H - r_\ell = B_t\ell$, so

$$w_\ell = \tau B_t + \sigma_m\, \mu(K)\, \nu(B_t\ell), \qquad 1 \le \ell \le \ell_d,$$

with m_ℓ non-decreasing in ℓ. On the trough, $k_\ell = 0$ and $r_\ell - r_{\ell-1} = 0$, so

$$w_\ell = m_\ell = \sigma_m\, \mu(K)\, \nu(H - r^\star) =: M, \qquad \ell_d < \ell \le \ell_r,$$

constant and equal to the largest value m takes on the descent. On the retrieval arm, $k_\ell = B_r$ and $r_\ell - r_{\ell-1} = (c - \varphi'(\xi_\ell))B_r > 0$, so

$$w_\ell = \sigma B_r + \tau\big(c - \varphi'(\xi_\ell)\big)B_r + \sigma_m\, \mu(K^\star_{>\ell})\, \nu(H - r_\ell), \qquad \ell_r < \ell \le L,$$

and here $K^\star_{>\ell}$ is strictly decreasing while r_ℓ is strictly increasing, so m_ℓ is strictly decreasing, reaching $m_L = \sigma_m\mu(0)\nu(H - cR) = 0$. Together with $m_0 = \sigma_m\mu(K)\nu(0) = 0$ this gives

the quasi-concavity of m with $\arg\max = [\ell_d, \ell_r]$, and M is strictly increasing in K because μ is non-decreasing, ν is non-decreasing, at least one strictly, and $r^\star(K)$ is strictly decreasing by Assumption 1.

For the full profile, the boundary values are $w_1 = \tau B_t + \sigma_m \mu(K)\nu(B_t)$ and $w_L = \sigma B_r + \tau(c - \varphi'(\xi_L))B_r \le (\sigma + \tau c)B_r$. Under Assumption 4, $M > (\sigma + \tau c)B_r \ge w_L$, and M likewise exceeds every value on the retrieval arm beyond the first, so the maximum of w is attained in the interior. In the bilinear case $\mu(K') = K'$, $\nu(u) = u/H$,

$$\frac{M}{w_1} = \frac{\sigma_m K(H - r^\star)/H}{\tau B_t + \sigma_m K B_t/H} = \frac{\ell_d}{1 + \tau H/(\sigma_m K)}, \qquad \ell_d = \frac{H - r^\star}{B_t},$$

which exceeds one whenever the compression phase spans more than $1 + \tau H/(\sigma_m K)$ layers. This is (iii). □

Remark 1. *The transport term τB_t is paid on the descent and on the retrieval arm but not on the trough, so w itself is quasi-concave only up to that term; the matching component alone is exactly quasi-concave. We state (iii) in terms of m for this reason. The distinction is immaterial whenever $\tau B_t \ll M$, which is implied by Assumption 4 together with $B_t \ge cB_r$.*

A.5 Proof of Theorem 2

By (C4) and $\varphi \ge 0$, $r_\ell \ge c(R - K_{>\ell}) = cR - cK_{>\ell}$, which is the first inequality. For the second, (C5) gives $w_j \ge \sigma k_j$, so $\sum_{j>\ell} w_j \ge \sigma K_{>\ell}$ and hence $K_{>\ell} \le \sigma^{-1} \sum_{j>\ell} w_j$; substituting into the first inequality gives the claim. On the retrieval arm of the optimal computation, Theorem 1(ii) gives $r_\ell = g(K_{>\ell}) = c(R - K_{>\ell}) + \varphi(K_{>\ell})$, which rearranges to the stated identity. □

Remark 2. *The weight-side inequality is loose by exactly the transport and matching components of $\sum_{j>\ell} w_j$. The exact conservation is between activation capacity and* retrievable knowledge*; the version in w is a corollary, and should not be read as a one-for-one trade between hidden-state rank and weight rank.*

A.6 Proof of Corollary 1

Let $\psi(K) = cK - \varphi(K)$. By Assumption 1, $\psi'(K) = c - \varphi'(K) \ge \eta > 0$, so ψ is a strictly increasing bijection onto its range with $\psi(0) = 0$, and $cR - r^\star(K) = cR - c(R-K) - \varphi(K) = \psi(K)$, giving the meter and its inverse. The map is affine exactly when φ is affine. Monotone deepening along a trajectory follows by composition with $K(t)$. At $K = 0$, (C4) reads $r_\ell \ge cR$ for all ℓ, so $\underline{r}_\ell = \max\{H - B_t\ell,\ cR\}$ is non-increasing in ℓ: there is no interior minimum below the output value and no re-expansion, hence no U. □

A.7 Proof of Theorem 3

Write $\hat{S} = S - \mathbb{E}S$ and absorb means into α. Substituting $G = \mathbb{E}G + \gamma\hat{S} + G_\perp$ into (F2),

$$D_{\text{late}} = \text{const} + (a' + b\gamma)\,\hat{S} + b\,G_\perp + \varepsilon_l,$$

with $G_\perp$ uncorrelated with S by construction. By (F3) the population slope is

$$\beta = \frac{\text{Cov}(D_{\text{late}}, D_{\text{early}})}{\text{Var}(D_{\text{early}})} = \frac{a\,(a' + b\gamma)\,\sigma_S^2}{a^2\sigma_S^2 + \sigma_{\varepsilon_e}^2} = (1 - \delta)\,\frac{a' + b\gamma}{a}.$$

The residual is therefore

$$\text{CAP} = (a' + b\gamma)\hat{S} + bG_\perp + \varepsilon_l - \beta(a\hat{S} + \varepsilon_e) = \delta(a' + b\gamma)\hat{S} + bG_\perp + \varepsilon_l - \beta\varepsilon_e,$$

which is the stated identity. Claims (i)–(iii) follow by taking covariances against $G_\perp$ and S and using (F3). □

A.8 Proof of Corollary 2

By (N) a contaminant satisfies (F2) with $G = 0$, so $G_\perp = -\mathbb{E}G - \gamma(s - \mathbb{E}S)$. Substituting into the identity of Theorem 3 and taking expectations over the noise terms,

$$\begin{aligned}\mathbb{E}[\text{CAP} \mid S = s, G = 0] &= \delta(a' + b\gamma)(s - \mathbb{E}S) - b\gamma(s - \mathbb{E}S) - b\,\mathbb{E}G \\ &= \big[\delta a' - (1 - \delta)b\gamma\big](s - \mathbb{E}S) - b\,\mathbb{E}G.\end{aligned}$$

The bracket is negative iff $\delta(a' + b\gamma) < b\gamma$, i.e. iff $\delta < b\gamma/(a' + b\gamma)$, and the expression is affine in s with that slope. Any score that is a monotone increasing function of s or of absolute loss ranks the same points in its upper tail by definition. □

Remark 3. *The prediction is affine in the latent surface value s. Regressing the observed* CAP *on the observed D_{early} within a contaminant cluster estimates the slope $\left[\delta a' - (1-\delta)b\gamma\right](1-\delta_{\text{noise}})/a$, where δ_{noise} is the attenuation within the cluster; only the sign and the affine shape transfer to measurables without further assumptions.*

A.9 Proof of Proposition 1

$\text{CAP}_i - \text{CAP}_j = b(G_{\perp,i} - G_{\perp,j}) + (\nu_i - \nu_j)$, and by the stated independence $\nu_i - \nu_j$ is sub-Gaussian with proxy $2\varsigma^2$. Thus

$$\Pr[\text{CAP}_i < \text{CAP}_j] \le \Pr[\nu_j - \nu_i > b\Delta] \le \exp\left(-\frac{b^2\Delta^2}{4\varsigma^2}\right),$$

and the top-m claim follows from a union bound over the at most n comparisons formed with the threshold example. □

A.10 Scope of the Theory

This section expands the scope discussion of Section 3.2. Because the argument spans a capacity model, a mapping to measurements, and a factor model, it is worth stating in full what each step does and does not carry.

The model is an accounting model, not a model of transformer computation. It predicts the *shape* of capacity profiles over depth, not their values, and it is silent about attention, tokenization, and everything else that determines the constants. Comparisons between predicted and measured curves should be read qualitatively.

Three claims have different strengths. The activation U and the deferred retrieval schedule (Theorem 1(i)–(ii)) follow from the addressing gain assumption alone, which is close to a definition of what a memory is. The weight inverted U (Theorem 1(iii)) additionally requires Assumption 3 on the cost of matching, and holds only in the regime of Assumption 4; it is the weakest link on the phenomenon side. The conservation law (Theorem 2) is exact in K and only an inequality in w: $\sum_{j>\ell} w_j$ also pays for transport and matching, so the exact trade is between activations and retrievable knowledge, not between hidden-state rank and weight rank.

Stable rank is a surrogate for capacity, not capacity. For activations this is the standard argument, exact only when the per-direction energy is approximately isotropic. For weight matrices we read $\text{srank}(W)$ as the number of directions the layer can read from or write to, which is a coarser proxy; the empirical weight profiles are correspondingly noisier than the activation profiles (Section 3.1).

The bridge to CAP is an exclusion restriction plus a factor model. Theorem 1 supplies the first, and Theorem 3 is an identification result conditional on the second, not evidence for the factor model. It nevertheless carries a testable implication that does not depend on the factor model being right: on a decoder without the rank equilibrium, at initialization for instance, the shallow block no longer precedes retrieval, the exclusion restriction fails, and CAP should carry no selection signal. We state this as a prediction rather than a result, since testing it requires scoring, fine-tuning, and evaluating with a randomly initialized decoder. Likewise, the sharpest form of Corollary 2 is stated in terms of the latent surface value s; regressing observed CAP on observed D_{early} within a noise cluster attenuates the slope by a further factor, so only the sign and the affine shape are cleanly predicted at the level of measurables. The empirical case for CAP rests on Section 4.3, not on the theory alone.

Finally, the theory predicts a sign and a mechanism, not effect sizes. It does not determine the depth split fractions, the response horizon R, or how much accuracy any of this should buy.

B Full Cross-Model Rank Survey

Figure 7 extends the four-model comparison of Figure 1b to 12 models. Table 5 describes the checkpoints in the survey. Parameter counts follow the nominal model sizes in the released checkpoint

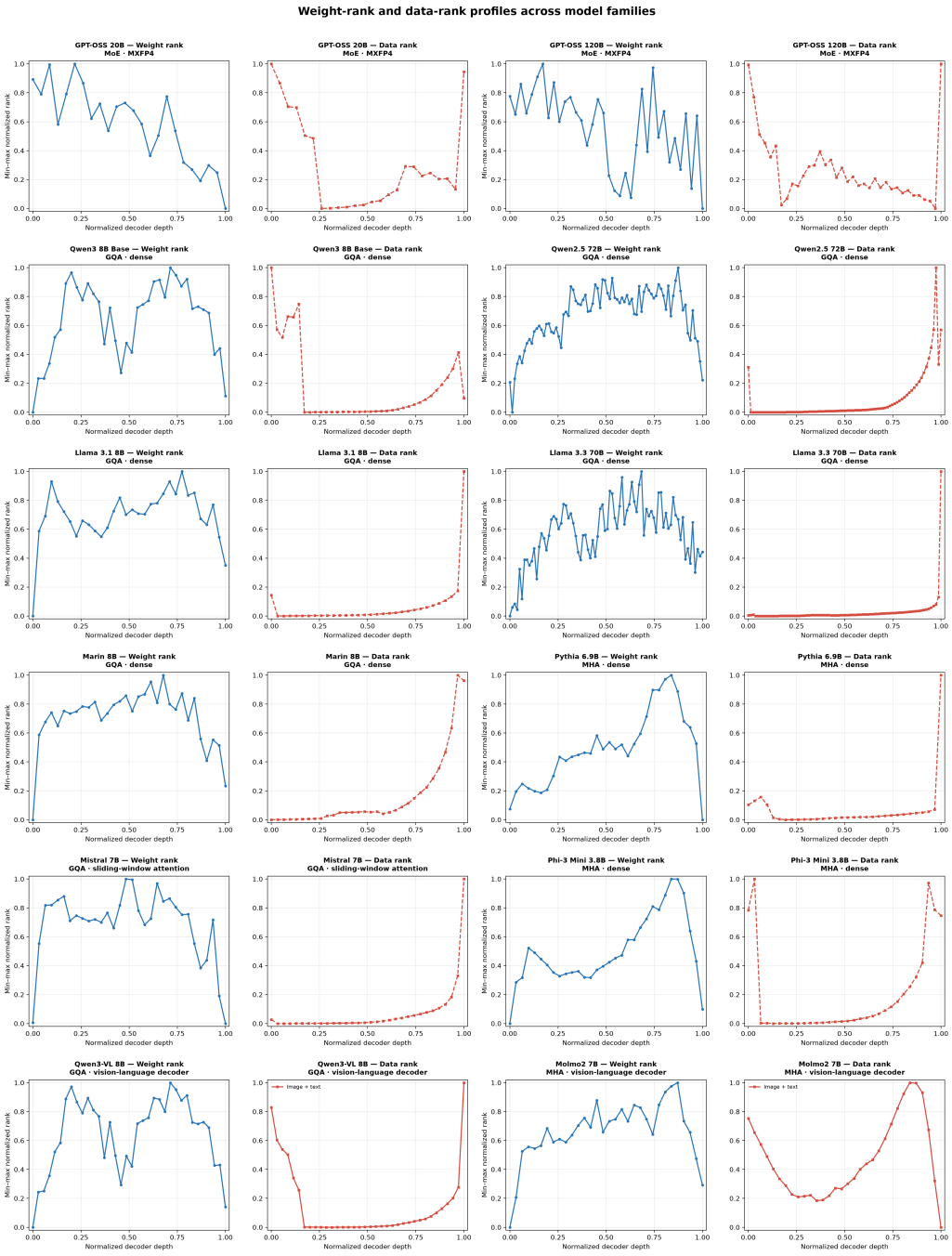


Figure 7: Weight and data stable rank over normalized decoder depth for 12 models. Table 5 lists their sizes, attention mechanisms, and architectures.

Model	Size	Decoder attention	Type	Modality
GPT OSS 20B	20B	GQA, alternating full/window	MoE	Text
GPT OSS 120B	120B	GQA, alternating full/window	MoE	Text
Qwen3 8B Base	8B	GQA	Dense	Text
Qwen2.5 72B	72B	GQA	Dense	Text
Llama 3.1 8B	8B	GQA	Dense	Text
Llama 3.3 70B Instruct	70B	GQA	Dense	Text
Marin 8B Base	8B	GQA	Dense	Text
Pythia 6.9B	6.9B	MHA	Dense	Text
Mistral 7B v0.3	7B	GQA, sliding window	Dense	Text
Phi 3 Mini 4K Instruct	3.8B	MHA	Dense	Text
Qwen3 VL 8B Instruct	8B	GQA	Dense	VL
Molmo2 O 7B	7B	MHA	Dense	VL

Table 5: The 12-model rank survey. Size is the nominal parameter count; GPT OSS sizes are total rather than active parameters. Full/window denotes alternating full and sliding-window attention. VL denotes vision-language.

Method	1k	2k	4k	8k
Base model		61.11		
Random	**63.99 ± 1.32**	65.88 ± 1.31	68.01 ± 1.28	69.37 ± 1.27
PPL	62.02 ± 1.34	64.97 ± 1.31	65.88 ± 1.31	68.92 ± 1.27
IFD	61.41 ± 1.34	62.70 ± 1.33	65.05 ± 1.31	66.34 ± 1.30
QuRating	61.11 ± 1.34	63.76 ± 1.32	67.78 ± 1.29	66.11 ± 1.30
LESS	60.27 ± 1.35	64.29 ± 1.32	68.16 ± 1.28	70.81 ± 1.25
k-center	63.31 ± 1.33	64.44 ± 1.32	66.11 ± 1.30	70.58 ± 1.26
CAP (ours)	62.32 ± 1.33	**67.78 ± 1.29**	**74.30 ± 1.20**	**76.04 ± 1.18**

Table 6: Full math results on GSM8K.

names. For GPT OSS, 20B and 120B denote total parameters, with approximately 3.6B and 5.1B active per token, respectively. GQA denotes grouped-query attention, MHA denotes multi-head attention, and MoE denotes mixture of experts. For vision-language models, the attention description refers to the decoder.

C Full Main-Domain Results

Tables 6–8 report the complete numerical results underlying Table 2 and Figure 3.

D Selection Cost Details

Table 9 reports both resource allocation and measured usage for scoring each 80k-example domain pool. We treat utilization-weighted GPU-hours as the primary compute measure because it integrates observed GPU utilization over wall time instead of charging equally for active and idle allocation.

E Further Text Domains

CAP is applied unchanged to two additional domains, with the same protocol as the main experiments and a 2k budget. Table 10 reports finance and Table 11 reports medicine.

Method	1k	2k	4k	8k
Base model		30.49		
Random	28.05 ± 3.52	**32.93 ± 3.68**	32.32 ± 3.66	30.49 ± 3.61
PPL	30.49 ± 3.61	28.05 ± 3.52	28.05 ± 3.52	32.93 ± 3.68
IFD	25.00 ± 3.39	28.66 ± 3.54	26.83 ± 3.47	28.66 ± 3.54
QuRating	29.27 ± 3.56	28.66 ± 3.54	31.71 ± 3.64	32.93 ± 3.68
LESS	**31.71 ± 3.64**	32.32 ± 3.66	**33.54 ± 3.70**	**35.37 ± 3.74**
k-center	30.49 ± 3.61	27.44 ± 3.49	29.27 ± 3.56	32.32 ± 3.66
CAP (ours)	**31.71 ± 3.64**	29.88 ± 3.59	**33.54 ± 3.70**	**35.37 ± 3.74**

Table 7: Full code results on HumanEval.

Method	1k	2k	4k	8k
Base model		32.55		
Random	33.89 ± 0.75	35.50 ± 0.76	37.57 ± 0.76	38.19 ± 0.77
PPL	**34.83 ± 0.76**	34.87 ± 0.75	36.41 ± 0.76	37.72 ± 0.76
IFD	33.85 ± 0.75	34.56 ± 0.75	37.48 ± 0.76	38.27 ± 0.76
QuRating	34.14 ± 0.75	35.06 ± 0.75	**38.76 ± 0.76**	38.51 ± 0.77
LESS	34.57 ± 0.75	36.03 ± 0.76	36.59 ± 0.76	38.39 ± 0.76
k-center	33.78 ± 0.75	35.07 ± 0.76	36.29 ± 0.76	37.08 ± 0.76
CAP (ours)	34.66 ± 0.75	**36.23 ± 0.76**	37.26 ± 0.76	**39.05 ± 0.77**

Table 8: Full science results on MMLU-Pro-5.

F Scaling the Budget on Code

Code is the one main-paper domain where a 10% budget does not match full-pool training (Section 4.3), so we extend the sweep to 60k, i.e., 75% of the pool. Table 12 and Figure 8 show that CAP holds a consistent margin over random and PPL at every budget, reaching $+15.24$ over the base model. Selection therefore does not merely reorder the head of the pool: it induces a useful total order over all of it.

G The Optimization Path

Selection quality can show up in two ways: a better final solution, or a faster route to it. To separate them we take the two 8k math runs that differ only in which examples were selected, save parameters every 25 optimizer steps, and project both paths onto the top two principal components of their joint trajectory, which capture 89% of its variance. On that plane we evaluate a single shared objective, the negative log-likelihood of held-out 5-shot GSM8K responses, so both runs are measured against the same surface rather than against their own training losses (Figure 9).

The two paths leave the shared base model, at loss 0.666, in opposite directions along the second component and never return to each other: the runs end in different basins, not at different points of one basin. CAP descends faster and further. After 25 steps it is already at 0.577, below the best value random selection reaches at any point of its run (0.582 at step 75), and it settles at 0.572 against 0.584. Random selection also degrades late, rising from 0.582 at step 75 to 0.584 by step 125, whereas the CAP path is monotone to within noise. Both observations are consistent with what the score selects. A set chosen for high assimilation gap poses one coherent demand on the weights, so its per-example gradients plausibly reinforce rather than cancel, and the run walks into a lower basin instead of oscillating inside a shallow one.

Domain	Method	Wall min.	GPUs	Allocated GPU-h	Avg. GPU util. (%)	Util.-weighted GPU-h
Math	Random	0.001	0	0.000	0.0	0.000
	PPL	4.476	4	0.298	81.6	0.243
	IFD	34.159	4	2.277	93.8	2.130
	QuRating	11.749	4	0.783	53.2	0.415
	k-center	1.321	1	0.022	10.1	0.009
	LESS	60.173	4	4.012	77.2	3.086
	CAP	63.476	4	4.232	66.0	2.780
Code	Random	0.001	0	0.000	0.0	0.000
	PPL	2.926	4	0.195	81.7	0.159
	IFD	20.222	4	1.348	78.1	1.049
	QuRating	2.716	4	0.181	85.0	0.154
	k-center	1.215	1	0.020	11.3	0.009
	LESS	52.862	4	3.524	44.6	1.566
	CAP	64.047	4	4.270	62.7	2.668
Science	Random	0.001	0	0.000	0.0	0.000
	PPL	7.769	4	0.518	78.2	0.403
	IFD	57.617	4	3.841	94.0	3.599
	QuRating	7.307	4	0.487	89.2	0.432
	k-center	1.704	1	0.028	7.6	0.009
	LESS	79.019	4	5.268	92.7	4.873
	CAP	65.685	4	4.379	68.7	2.995

Table 9: Selection cost for each method and domain. The highlighted column is the primary compute measure used in Figure 4 and the main-text analysis. Values are rounded for readability.

Method	FinanceBench	MMLU-Pro Bus.	MMLU-Pro Econ.	Macro avg
PPL	19.33	34.60	47.39	33.77
Random	18.67	<u>37.01</u>	<u>48.82</u>	<u>34.83</u>
CAP (ours)	**22.00**	**37.14**	**49.05**	**36.06**

Table 10: Finance: 2k selected from 80k of Sujet-Finance-Instruct (Sujet AI, 2024), evaluated on FinanceBench (Islam et al., 2023) and MMLU-Pro splits.

H Training and Evaluation Details

H.1 Supervised Fine-Tuning Protocol

All text experiments fine-tune `marin-community/marin-8b-base` with full-parameter SFT using LlamaFactory, driven by a single sweep script so that every selection method is trained by an identical pipeline. The only thing that varies between cells of a table is the list of selected example indices; the pool, the budget, the shuffling order, the random seed, the optimizer state, the schedule, and the number of epochs are held fixed. Table 13 lists the configuration.

Two choices deserve comment because they affect how the results should be read. First, we train for a single epoch at every budget. This means that the number of optimizer steps grows linearly with the budget, so a budget curve conflates "more data" with "more updates"; the comparison between methods at a fixed budget is nevertheless exact, since all methods take the same number of steps on the same schedule. Second, the maximum sequence length of 2048 tokens covers the large majority of responses in all three pools, but it truncates a small tail of long chain-of-thought solutions in the math pool; truncation is applied identically to every method.

H.2 Pool Construction

Normalization and sampling. We use the same candidate pool for every selection method within a domain. Text-pool deduplication applies Unicode NFKC normalization, Unicode case folding, whitespace collapse, and trimming. The key is the normalized instruction together with the optional

Method	Diagnosis accuracy	Reasoning recall
PPL	<u>14.38</u>	32.71
Random	13.60	<u>34.72</u>
CAP (ours)	**15.16**	**35.79**

Table 11: Medicine: 2k selected from the 13k MedCaseReasoning pool (Wu et al., 2025).

Method	2k	8k	20k	40k	60k	Max gain
Random	29.27	32.93	34.76	38.41	42.07	+11.59
PPL	30.49	31.71	34.15	37.80	43.90	+13.41
CAP (ours)	**32.32**	**32.93**	**36.59**	**40.24**	**45.73**	**+15.24**

Table 12: Code at enlarged budgets (HumanEval pass@1, base 30.49).

input, not the response. We retain the first occurrence of each key and sample without replacement with seed 0. Each of the three main domains contains 80,000 sampled examples.

Main text domains. For math, the source is the 859,494-example training split of NuminaMath-CoT (Li et al., 2024a). For code, we use the 118,278-example `educational_instruct` split of OpenCoder opc-sft-stage2 (Huang et al., 2024); prompt deduplication leaves 92,813 unique examples, from which we sample 80,000. This removes 25,465 duplicates, or 21.53% of the source split. For science, we restrict the 2,335,220-example training split of WebInstructSub (Yue et al., 2024) to its 317,209-example Science StackExchange subset covering physics, biology, chemistry, and computer science, and sample 80,000 examples. Separate post-deduplication counts for the math and science sources are not available in the pool summary, so we do not report duplicate-removal rates for those two domains.

Additional text pools. The filtered Sujet-Finance source (Sujet AI, 2024) contains 86,217 valid and unique prompts before sampling 80,000 examples. MedCaseReasoning (Wu et al., 2025) contains 13,092 valid and unique prompts. The MedMCQA pool contains 182,822 valid rows and 182,788 unique prompts, removing 34 duplicates, or 0.019% of valid rows. These counts distinguish source filtering and deduplication from the subsequent selection budget.

Multimodal document pool. We presample 23,000 document-domain candidates. The source is Molmo2-SynMultiImageQA. Deduplication on normalized questions retains 22,984 candidates, removing 16 duplicates, or 0.070%, after which we uniformly sample 20,000 examples. Image bytes are not part of this deduplication key. Consequently, identical normalized questions paired with different documents can be merged by this protocol.

H.3 Scoring Configuration

CAP. The text scoring pipeline uses the raw tokenizer rather than the SFT chat template. We strip the instruction and, when an input is present, append a newline followed by the stripped input. The reference branch appends the stripped response without adding response special tokens. We retain the last 256 prompt tokens and the first $R = 48$ response tokens, excluding examples with fewer than two prompt tokens or six response tokens. The counterfactual branch generates from the same truncated prompt using greedy decoding, with sampling disabled, one beam, and at most 48 new tokens. Both branches run in bfloat16. Hidden states are collected from decoder-block outputs at response positions; only blocks in the first and final thirds are retained. Specifically, D_{early} uses layers up to $0.33L$ and D_{late} uses layers from $0.66L$. The residualizing regression is fitted separately on each domain pool. Examples are sorted by tokenized length and scored in batches of 16. The response-horizon ablation varies R as described in Section 5.2.

PPL. Perplexity scoring uses the same 256-token prompt limit, 48-token response limit, six-token response minimum, bfloat16 model, and batch size of 16 as CAP.

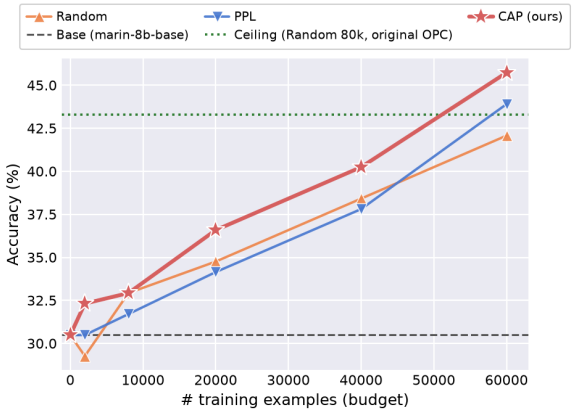


Figure 8: Code accuracy versus enlarged budgets.

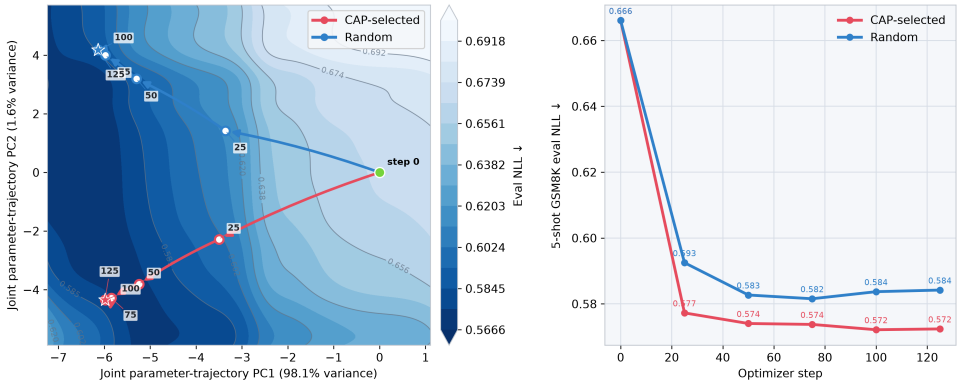


Figure 9: SFT trajectories of CAP and random selection in the plane spanned by the top two PCs of the joint parameter path, over a shared held-out 5-shot GSM8K loss surface.

QuRating. We score the full example text. The input concatenates the instruction, optional input, and response. The rater is `princeton-nlp/QuRater-1.3B`, and the selection signal is the `educational_value` logit. The maximum sequence length is 512 tokens and the batch size is 16.

IFD. We score one example at a time on each GPU. Four strided scoring shards run concurrently across four GPUs, giving a per-GPU batch size of 1 and an aggregate concurrent batch size of 4. Each example is truncated to a maximum sequence length of 1,024 tokens. The score is the ratio of response perplexity conditioned on the instruction to response perplexity without the instruction. Prompt tokens are masked from the conditional loss, and examples with an IFD ratio greater than or equal to one are excluded from selection.

k-center. We embed the concatenated prompt and response. The encoder is `sentence-transformers/all-MiniLM-L6-v2`; we mean-pool token representations and normalize each embedding to unit length. The embedding maximum length is 256 tokens and the batch size is 256. Thus the geometric baseline uses both the prompt and response, rather than instruction-only embeddings.

LESS. The selection cost comparison uses LESS 8192 without gradient warmup. The target set contains at most 2,000 examples from the downstream domain: the first 2,000 GSM8K training examples for math, all 164 HumanEval problems with their canonical solutions for code, and the complete 285-example MMLU validation split for science, comprising five examples from each of 57 subjects. The science target is drawn from MMLU, not MMLU-Pro. The code target overlaps the

Setting	Value
Base model	`marin-community/marin-8b-base`
Fine-tuning type	full parameter (no adapters)
Chat template	`llama3`
Max sequence length	2048 tokens
Sequence packing	disabled
Optimizer	AdamW (`adamw_torch`)
Peak learning rate	1×10^{-5}
LR schedule	cosine decay
Warmup ratio	0.03
Epochs	1
Per-device batch size	2
Gradient accumulation	8
Effective batch size	64 ($2 \times 8 \times 4$ GPUs)
Precision	bf16
Gradient checkpointing	enabled
Distributed backend	DeepSpeed ZeRO stage 3
Hardware	4× NVIDIA A100 SXM

Table 13: SFT configuration, shared by every method and budget.

entire HumanEval evaluation set and includes its reference solutions; this cost configuration therefore does not use a disjoint target set for code.

LESS attaches rank-8 LoRA modules with scaling factor 16 and zero dropout to the query, key, value, and output projection matrices. For each candidate and target example, it computes the gradient of the response-only causal language modeling loss with respect to all trainable LoRA parameters, using a maximum sequence length of 1,024 tokens. The full LoRA gradient is compressed to an 8,192-dimensional vector using a fixed signed CountSketch projection and then normalized to unit length. Target vectors are averaged and normalized, and each candidate is scored by cosine similarity to this aggregate target gradient.

Candidate and target gradients are computed one example at a time on each GPU, with four strided shards running concurrently across four GPUs. Thus the LESS scoring batch size is 1 per GPU and the aggregate concurrent batch size is 4.

H.4 Evaluation

Shared configuration. The three main text benchmarks use lm-evaluation-harness version `0.4.13.dev0`, with bfloat16 inference, a model context length of 4,096 tokens, and evaluation batch size 16.

GSM8K. We evaluate all 1,319 test examples with five demonstrations from the training split. Each item is formatted as `Question:` {`question`} followed by a newline and `Answer:`; demonstrations are separated by two newlines. Decoding is greedy with temperature 0 and at most 1,024 generated tokens, stopping before a new `Question:` or an end token. We report strict exact match after extracting the number following `####` and removing commas, dollar signs, and a final period.

HumanEval. We evaluate all 164 test problems with zero demonstrations, using the canonical function prefix supplied by the benchmark directly. Decoding is greedy with at most 1,024 generated tokens and stops on a new class definition, function definition, comment, `if`, or `print` statement. We report pass@1 using the official execution-based tests.

MMLU-Pro STEM. The reported score is the macro-average over biology, chemistry, computer science, engineering, and physics. We evaluate the complete test subsets of 717, 1,132, 410, 969, and 1,299 examples, respectively, for a subsampling fraction of 1.0 in every subject. Each subject uses the first five chain-of-thought demonstrations from its validation split. The prompt asks the model to reason step by step and finish with `the answer is (X)`. Decoding is greedy with temperature 0,

at most 512 generated tokens, and `Question:` as the stop sequence. Answers are extracted using the case-insensitive regular expression

```
answer is \(?([ABCDEFGHIJ])\)?
```

and exact match ignores case and punctuation.