IN-CONTEXT TASK LEARNING: READING OUT A KERNEL CLASSIFIER

Anonymous authors
Paper under double-blind review

ABSTRACT

Task learning in context requires using demonstrated mappings when pretrained knowledge and other task cues do not determine them. We investigate whether small sets of attention heads can support direct prediction and reuse of demonstration evidence. Starting from Llama-3.1-8B on TREC-fine with 180 demonstrations, we identify mapping-sensitive carrier heads through paired intact/randomized-label contrasts. Their writes support a kernel classifier based on query-dependent retrieval of contextual class votes, characterized through query-weight controls and value interventions. This account motivates *Selective TL Memory*, which caches demonstration-side states while preserving live retrieval and direct readout. Across models, natural-language tasks, and demonstration budgets, thirty-head full-access readouts match or exceed final-model accuracy in 12 of 18 settings. Selective access to additional demonstrations improves a fixed carrier readout in all 18 settings at both tested head budgets; thirty-head selective readout exceeds every adapted vector baseline in 14. Contextual label states often support compact reuse, while access and receiver controls show that the selected readouts depend on states constructed by the surrounding network. In a prefix-cached benchmark, shared memory supports repeated queries with latency close to the smaller prompt and reduced resident storage relative to full access.

1 INTRODUCTION

Since GPT-3 (Brown et al., 2020), large language models (LLMs) have been known to perform tasks from demonstrations without parameter updates, an ability termed in-context learning (ICL). An influential line of work emphasizes *task recognition* (TR): demonstrations identify familiar tasks and activate knowledge acquired during pretraining (Xie et al., 2022; Min et al., 2022; Pan et al., 2023). Longer context windows have complicated this picture. With more demonstrations, models can achieve substantial gains and follow input–output mappings that conflict with their pretrained priors (Agarwal et al., 2024; Wei et al., 2023). These findings motivate studying *task learning* (TL): acquiring a mapping from demonstrations and applying it to unseen inputs when pretrained knowledge and other task cues do not determine that mapping. Pretraining can supply useful representations and inductive biases, while demonstrations supply the missing relationships. A few examples may suffice to recognize a familiar task while providing too little evidence to learn an unfamiliar mapping. This difference motivates examining TL as the amount of in-context evidence grows.

This demonstration demand is visible in our Llama-2 experiment on TREC-fine with 36 relabeled classes: accuracy remains near chance with one demonstration per class and reaches its highest observed value at nine per class, or 324 demonstrations (Figure 1). Our main experiments use arbitrary label assignments: pretrained representations can support semantic classification, while demonstrations supply the class–label correspondences needed for prediction. Larger demonstration sets also increase the cost of attention and its inspection, motivating targeted analysis of how this evidence is used and reused.

Kernel and attention-head accounts already connect ICL to matching demonstrations and retrieving label information (Han et al., 2025; Yu & Ananiadou, 2024). Several closely related head-level analyses develop their main evidence with four to sixteen demonstrations (Yu & Ananiadou, 2024; Yang et al., 2026; Sun et al., 2025). Motivated by the expectation that larger demonstration sets

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

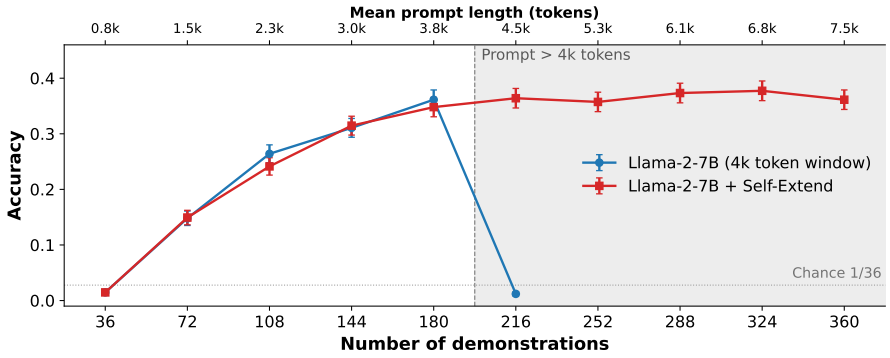


Figure 1: Demonstration demand in a TL setting. Llama-2 accuracy on TREC-fine with 36 relabeled classes as the total number of demonstrations N varies. With a long-context variant (SelfExtend), the highest accuracy is observed at $N = 324$, which exceeds the context window of vanilla Llama-2.

can more fully engage task-learning mechanisms, we focus our mechanism analysis on Llama-3.1-8B/TREC-fine with 180 demonstrations. Our question concerns direct prediction and reuse: *can a small selected set of heads already provide useful class predictions, and can its retrieval of demonstration evidence support selective memory reuse?*

In this setting, we select *carrier heads* by contrasting correct-label writes under intact and randomized demonstration labels, without materializing attention matrices. Matching inputs, label vocabulary, and positions cancels contributions unchanged by relabeling. Their summed class-directed writes support direct prediction, with readouts from as few as eight heads broadly following the model's accuracy trend across demonstration budgets (Figure 4). A kernel-vote decomposition separates query-dependent retrieval weights from contextual class votes. Query-weight and order controls show that predictions depend on query-specific retrieval of demonstration content. Value–output projections reveal class-favoring votes at label positions, and value interventions test local response predictions and the behavioral effects of restoring contextual values. Comparisons with Qwen3 and dbpedia14 examine variation across models and tasks. For a fixed causal prefix, the demonstration-side states are reusable across queries, while their retrieval remains query dependent.

This separation motivates *Selective TL Memory*: prepare contextual demonstration states once and let selected heads retrieve them using live queries. The direct variant predicts from their collected writes and stops after the deepest selected attention sublayer; the anchor variant restricts additional demonstration memory to label positions after full-prefix encoding. The design retains explicit retrieval over cached examples, giving a different reuse rule from fixed-vector activation interventions (Hendel et al., 2023; Todd et al., 2024; Liu et al., 2024b; Li et al., 2025b). Such reuse is useful when many queries share a task-defining demonstration bank, such as labeled support requests specifying an organization's routing rules. Many-shot ICL keeps model parameters fixed (Agarwal et al., 2024), but retaining and attending to demonstrations incurs serving costs even with prefix caching. We evaluate the design across six Llama and Qwen configurations on arbitrary-label natural-language tasks. Selective access improves a fixed carrier readout in all 18 model–task–budget settings at both head budgets. Full access remains more accurate, and receiver and other-head controls show that the selected readout benefits from states constructed by the surrounding network. In a prefix-cached benchmark, selective reuse keeps online latency close to the smaller prompt while reducing resident storage relative to full access. Our contributions are:

- Starting from Llama-3.1-8B/TREC-fine, we identify mapping-sensitive readers whose collected writes support direct classification. Head-selection and budget comparisons evaluate the predictive performance available from small subsets and the task-dependent gains from additional readers.
- We characterize the evidence used by these readouts through query-weight and order controls, contextual vote analysis, and value interventions. Comparisons with Qwen3 and dbpedia14 examine shared patterns and variation in retrieval, intervention responses, and repair.

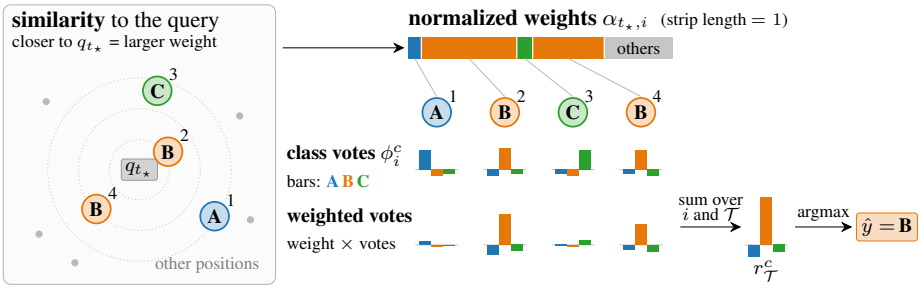


Figure 2: Retrieval and voting in a carrier head. Attention weights aggregate contextual class votes supplied by the value–output pathway. Summing class-directed writes across carriers gives the direct prediction scores. The decomposition holds for any head set; the controls and interventions test its empirical role in the selected carriers.

- We develop Selective TL Memory to reuse contextual evidence through live retrieval, and test the predictive gains retained under selective access, their dependence on receiver computation, and the accuracy and online costs of compact memory reuse.

2 From Mechanism Analysis to Selective TL Memory

We study a fixed pretrained model predicting labels for unseen queries from demonstrations $D = \{(x_i, y_i)\}_{i=1}^N$.

Definition 1 (Task Learning) *Task learning is the ability to acquire input–output relationships from demonstrations and apply them to unseen inputs when pretrained knowledge and the remaining task cues do not determine those relationships.*

In the main experiments, demonstrations specify an arbitrary label assignment over potentially familiar semantic classes (Pan et al., 2023; Wei et al., 2023). Task learning here concerns the demonstrated class–label correspondence and can use pretrained representations; the formalization and dataset construction are in Appendix A.1. Supplementary rule-learning experiments extend the evaluation beyond label reassignment (Appendix E).

2.1 Selecting Heads by Mapping Sensitivity

TSLA relates head outputs to label-subspace geometry (Yang et al., 2026). We identify dependence on demonstrated mappings through a paired output contrast. At the answer position $t_\star$, let $a_{t_\star}^{(\ell,h)}$ be head h's residual write after its output projection (Elhage et al., 2021). Its raw contribution along candidate label $c \in Y$, with unembedding vector W_U^c, is

$$s_c^{(\ell,h)}(q; D) = \left\langle a_{t_\star}^{(\ell,h)}(q; D), W_U^c \right\rangle . \tag{1}$$

For each query's correct label y_q, compare intact demonstrations with $M = 5$ controls whose labels are sampled independently from Y:

$$S_{\ell,h} = \mathbb{E}_{q\in\mathcal{V}}\left[s_{y_q}^{(\ell,h)}(q; D) - \frac{1}{M}\sum_{m=1}^{M} s_{y_q}^{(\ell,h)}(q; D_{\text{rand}}^{(m)}) \right]. \tag{2}$$

Inputs, queries, label vocabulary, and positions remain fixed. Any class, format, or positional contribution unchanged by relabeling cancels, so a stable label preference alone cannot earn a high score. For a fixed head budget, the top-ranked heads form the *selected carrier* set $\mathcal{T}$. Varying the budget gives a family of readouts along the same ranking. Selection uses head outputs without materializing attention matrices; for method evaluation, $\mathcal{V}$ contains validation queries and the ranking is frozen before testing.

Static alignment with label directions does not by itself identify mapping-sensitive writes: some strongly aligned heads have near-zero or negative TL-scores (Appendix C). The paired contrast supplies candidates; query and order controls test their retrieval behavior (Appendix B.2). Section 3.2 evaluates the resulting direct readouts against alternative head-selection criteria.

The primary diagnostic uses Llama-3.1-8B/TREC-fine with 180 demonstrations ($K = 5$; 3,144 prefix tokens), seed 42, and 250 queries, with heads selected and analyzed on that draw. These descriptive diagnostics are separate from the validation-selected method evaluation. Replications cover a second Llama prefix seed, both Qwen3 scales, and Llama/dbpedia14 (Tables 7 and 9).

2.2 Query-Dependent Retrieval and Class Votes

Let $g(h)$ identify the key/value head shared by query head h. At source position i, the value–output pathway supplies the class vote $\phi_i^{c,(\ell,h)} = \left\langle W_O^{(\ell,h)} v_i^{(\ell,g(h))}, W_U^c \right\rangle$. Linearity gives the summed carrier write over visible positions $\mathcal{I}_{t_\star}$:

$$r_{\mathcal{T}}^c(q) := \sum_{(\ell,h)\in\mathcal{T}} s_c^{(\ell,h)}(q;D) = \sum_{(\ell,h)\in\mathcal{T}} \sum_{i\in\mathcal{I}_{t_\star}} \alpha_{t_\star,i}^{(\ell,h)}(q;D)\, \phi_i^{c,(\ell,h)}(q;D). \tag{3}$$

Attention's linear aggregation gives this decomposition for any head set. It separates query-dependent retrieval weights from contextual class votes (Figure 2). For the selected carriers, the controls below examine whether these scores reflect retrieval of the demonstrated mapping. Keys and votes are contextual, and weights can depend on displayed labels. For a fixed causal prefix, demonstration-side votes are reusable across queries, while query-side states remain live.

Retrieval depends on the query and demonstration content. Restricting the readout to demonstration positions gives 50.4% accuracy with each query's own weights, versus 11.6% with mean weights and 6.4% with another query's weights (Figure 3A). Own-weight readout outperforms both controls in all five conditions in Table 7. The early carrier slice in Figure 3A, drawn from the top 128 heads by TL-score, shows the same qualitative query dependence beyond the selected top eight. Reordering the same demonstrations preserves class preference and yields higher retrieval correlations by demonstration identity than by source slot throughout (Appendix B.2). Demonstration-level heatmaps show unequal weights among same-class examples (Appendix B.4).

Retrieved values favor the demonstrated classes. Across the five diagnostic conditions, label positions occupy only 1.35–5.86% of demonstration-block tokens but receive 74.5–88.5% of its carrier attention. Their class-averaged votes favor the demonstrated class in 99.0–100.0% of ordered class comparisons (Table 7). Combining these contextual votes with each query's retrieval weights yields the classifier

$$\hat{y}_{\mathcal{T}}(q) = \arg\max_{c\in Y} r_{\mathcal{T}}^c(q). \tag{4}$$

Once the selected writes have been computed, this classifier produces a prediction without running the remaining model layers. Each reader uses contextual states supplied by the computation preceding it. With full access to five demonstrations per class in the method evaluation, eight heads on Llama-3.1-8B/TREC-fine reach 50.27% accuracy, close to the final model's 51.60% (Table 2). Section 3.2 evaluates prediction across reader budgets and selection criteria; detailed mechanism controls are in Appendix B.

2.3 Intervention Responses and Value Repair

Value interventions characterize the response of the selected readouts and the behavioral role of their contextual inputs. For changed votes at positions $\mathcal{I}_d$ of demonstration d, the original retrieval weights give the local linear term

$$\widehat{\Delta r}^{\,c}(q) = \sum_{(\ell,h)\in\mathcal{T}} \sum_{i\in\mathcal{I}_d} \alpha_{t_\star,i}^{0,(\ell,h)}(q)\, \Delta\phi_i^{c,(\ell,h)}. \tag{5}$$

For fixed queries and keys in a single attention operation, the value response is linear. We test how well the local term tracks aggregate carrier contrasts in a multi-layer forward, where later retrieval and residual states can change, fitting a shared through-origin gain.

Local response fits and averaged-weight controls. Fifty value-only label swaps are evaluated on 250 queries each (Figure 3B). Across five diagnostic conditions, per-query weights improve fitted carrier-response R^2 by 0.227–0.442 over the stronger leave-one-out class mean, grouped by gold or clean-prompt predicted class. Per-query weights also outperform global and class means for

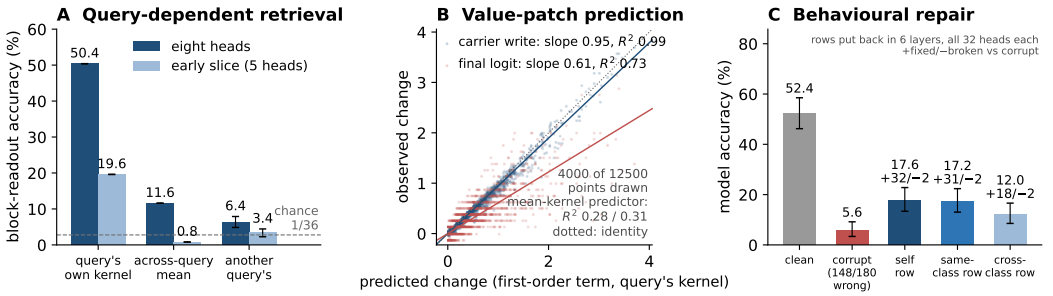


Figure 3: Mechanism tests on Llama-3.1-8B/TREC-fine ($K = 5$, seed 42, 250 queries). **(A)** Demonstration-block accuracy over 36 labels with own, mean, or shuffled query weights; dark bars: the top-eight carriers; light bars: an early carrier slice of five heads drawn from the top 128 by TL-score. Shuffle bars: mean ± s.d. over 200 permutations. **(B)** Predicted versus observed carrier and final-logit contrasts under value-only label swaps; dotted line: identity. **(C)** Final-model accuracy over 50 labels after label corruption and value repair. Protocols are in Appendix B.

final logits and prompt edits. Full fits are in Table 9; downstream corrections and output-scale comparisons are in Appendices B.5 and B.6.

Repairing values changes behavior. On Llama-3.1-8B/TREC-fine at the largest tested corruption level, self repair raises accuracy from 5.6% to 17.6%, against 52.4% clean accuracy; same-class and cross-class donors give 17.2% and 12.0% (Figure 3C). The additional settings show distinct repair profiles. On dbpedia14, self and same-class donors restore 70.8% and 72.0% from 1.2%, whereas cross-class donors give 0.8%. Both Qwen scales show partial recovery without statistically resolved donor-class advantages. Restoring clean label-position values improves final-model accuracy in all four settings (Appendix B.5). The patches affect whole value rows in carrier-containing layers, so this behavioral evidence concerns their value stream rather than the eight carriers alone.

2.4 Selective Memory

The Llama-3.1-8B/TREC-fine readout analysis motivates a memory design for reusing a demonstration bank across many queries. Selective TL Memory prepares and caches demonstration-side keys and values for a fixed causal prefix, while the receiver continues to form live queries. This separates one-time memory preparation from online retrieval and prediction, keeping model parameters fixed. We retain base demonstrations because selective access supports much better direct and final-model accuracy with them than without them (Appendix D.1).

For $K_{\text{base}} \to K_{\text{full}}$, partition demonstrations into D_{base} and D_{inc} and freeze the validation-selected carriers $\mathcal{T}$. Encode the full prefix once; retain prefix/base keys and values for all key/value heads, but increment columns only for groups used by carriers. Base states can already encode preceding increment examples, so the intervention restricts direct online access. The access controls below test how much useful evidence remains available under this restriction.

Online computation processes the query suffix at its full-prompt positions. All heads access prefix, base, and causally visible query positions; carriers additionally access the increment, with visibility enforced per query head. The upstream network remains active to form fresh queries from current residual states, and cached and live sources share one softmax.

Ours (direct) applies Equation 4 to raw writes from all visible positions, without final normalization or class centering, and stops after the deepest carrier's attention sublayer. *Ours (model)* retains these query-conditioned writes in the residual stream and continues to final logits. The two readouts compare classification from the collected writes with prediction that also uses the remaining model computation. *Ours (anchor)* keeps only increment label positions with their full-prefix keys and values, recomputes attention, and uses direct readout.

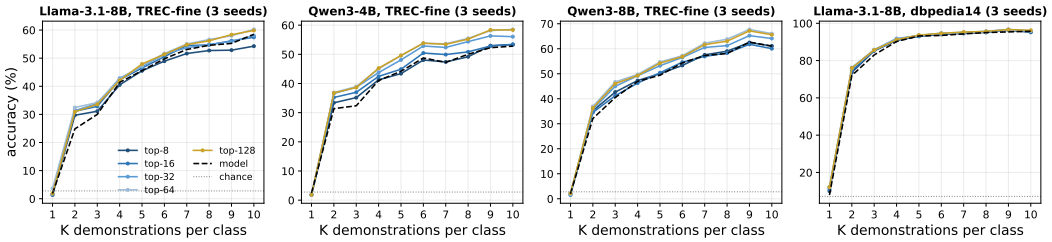


Figure 4: Accuracy across demonstration and head budgets, averaged over three seeds. From left to right: Llama-3.1-8B/TREC-fine, Qwen3-4B/TREC-fine, Qwen3-8B/TREC-fine, and Llama-3.1-8B/dbpedia14. Colored curves show readouts from the top 8, 16, 32, 64, and 128 heads; the black dashed curve shows model accuracy, and the gray dotted line marks chance. K is the number of demonstrations per class. Demonstration sets are nested across K, with their order shuffled separately at each K.

3 Experiments

3.1 Experimental Setup

Models and tasks. We evaluate six model configurations on TREC-fine (Llama-3.1-8B, Llama-2-7B with SelfExtend, Qwen2.5-7B, and Qwen3-4B/8B/14B) and Llama-3.1-8B on six natural-language classification datasets (TREC-fine, banking77, clinc150, dbpedia14, yahoo, and yelp), all with arbitrary single-token class labels; together these give 18 model–task–budget settings. Dataset construction, eligibility, templates, and model settings are in Appendices A.1 and A.3.

Protocol. Demonstrations are sampled class-balanced (Gao et al., 2021), K per class without replacement, then shuffled; a level $K_{\text{base}} \rightarrow K_{\text{full}}$ gives the base and total demonstrations per class, with the base nested in the full set. Each seed reserves validation examples per class before sampling, and carrier heads and tunable baseline settings are selected on these queries and frozen before test evaluation. We use three seeds with 250 shared test queries per seed, and evaluate eight- and thirty-head carrier sets from the same validation ranking (thirty matches TSLA's default budget; Appendix A.2).

Baselines and reference conditions. We compare against TSLA task-learning and task-recognition vectors (Yang et al., 2026), function vectors (FV) (Todd et al., 2024), task vectors (TV) (Hendel et al., 2023), in-context vectors (ICV) (Liu et al., 2024b), and I2CL (Li et al., 2025b). FV, TV, ICV, and I2CL are adapted to use the increment demonstrations and intervene on the same receiver; TSLA vectors are extracted from full-demonstration prompts (Appendix A.2). *Natural* and *All heads* give the final-model prediction with the increment hidden from, or visible to, every head; the access and readout conditions for the carrier classifier are defined in the caption of Table 2.

3.2 Direct Prediction from Mapping-Sensitive Heads

Figure 4 shows that accuracy increases strongly with the number of demonstrations per class across all four settings. At $K = 1$, both the selected-head readouts and the model remain near chance, followed by a sharp improvement at $K = 2$. Their subsequent trajectories differ across tasks: Llama-3.1-8B on dbpedia14 reaches approximately 90% accuracy by $K = 4$ and then gradually levels off in the mid-90s, whereas the three TREC-fine settings generally continue improving over larger demonstration budgets.

Readouts using only eight heads already broadly track the model's response to additional demonstrations. Expanding the head budget provides further gains on TREC-fine: the top-64 and top-128 curves closely overlap and generally exceed the smaller-head readouts. On dbpedia14, the head-budget curves remain tightly clustered, indicating a smaller benefit from additional heads. The larger head sets often match or exceed model accuracy. These patterns show that a small selected subset already supports useful prediction, with further gains available from more readers depending on the task. This sweep evaluates readout performance across budgets, complementing the query-weight and intervention tests at $K = 5$.

Table 1: Direct-output accuracy (%) under selective memory access for alternative head-selection criteria. Ours uses the registered eight-head set or the top thirty heads by TL-score; each selected set supplies both the memory readers and the direct readout. TSLA-TL is used as a head-ranking criterion here. Bold marks the highest point estimate at each budget. TREC denotes TREC-fine; Llama denotes Llama-3.1-8B. Paired no-access controls and head-set overlaps are in Appendix D.2.

Setting	Heads	Random	Attn. share	TSLA-TL	Ours
Llama/TREC, $2 \rightarrow 5$	8	5.82	22.00	21.73	**44.40**
	30	11.64	38.00	22.67	**47.47**
Qwen3-4B/TREC, $2 \rightarrow 5$	8	9.38	16.27	18.93	**41.73**
	30	6.53	23.07	20.80	**46.67**
Qwen3-8B/TREC, $5 \rightarrow 10$	8	6.09	24.13	22.93	**37.60**
	30	9.02	23.60	21.87	**42.27**
Llama/dbpedia14, $2 \rightarrow 5$	8	18.98	68.00	80.40	**83.20**
	30	14.13	85.60	84.80	**88.93**

Table 2: Test accuracy (%) on the 18 natural-language settings under different memory access and prediction readouts. *Natural*: final-model prediction with the increment hidden from every head; its base states come from the full-prefix encoding, so it is a visibility control rather than a separately encoded K_{base} prompt. *All heads*: full access, final logits. *No access* and *Full access (direct)*: the same carrier readout with the increment hidden from, or visible to, all heads; full access is an empirical reference, not an upper bound. *Ours (direct/model/anchor)* follow Section 2.4; anchor uses a lean implementation with matched controls in Appendix D.9. Base access is retained throughout. Bold compares the six Ours columns; dashes mark unavailable results.

					No access (direct)		**Full access (direct)**		**Ours (model)**		**Ours (direct)**		**Ours (anchor)**	
Model	**Task**	**Level**	Natural	All heads	8	30	8	30	8	30	8	30	8	30
Llama-3.1-8B	TREC-fine	2→5	36.93	51.60	35.07	36.93	50.27	51.60	38.40	46.67	44.40	**47.47**	44.40	47.07
Llama-3.1-8B	TREC-fine	5→10	49.33	58.27	48.93	52.67	54.53	59.07	50.00	50.40	51.60	**53.07**	51.73	52.40
Llama-2-7B + SelfExtend	TREC-fine	2→5	14.93	34.80	12.40	15.73	28.53	37.60	14.80	17.33	14.93	**21.07**	—	—
Llama-2-7B + SelfExtend	TREC-fine	5→10	34.13	39.60	23.87	30.40	29.20	38.27	34.27	**35.33**	24.67	31.87	—	—
Qwen3-14B	TREC-fine	2→5	40.27	55.20	39.73	43.20	57.07	58.13	42.80	45.73	45.07	**49.07**	46.27	**49.07**
Qwen2.5-7B	TREC-fine	2→5	33.60	48.00	28.27	34.27	49.20	54.40	34.67	36.13	41.33	43.47	41.07	**45.20**
Qwen2.5-7B	TREC-fine	5→10	38.53	59.47	35.33	41.20	56.27	68.13	38.53	29.60	45.87	52.67	44.53	**55.20**
Qwen3-8B	TREC-fine	2→5	37.33	57.60	34.13	35.47	57.20	58.40	38.13	44.80	40.80	43.47	43.33	**46.27**
Qwen3-8B	TREC-fine	5→10	33.60	62.53	34.53	33.07	63.87	65.33	34.40	40.00	37.60	42.27	37.07	**43.87**
Qwen3-4B	TREC-fine	2→5	33.60	42.27	33.47	37.07	44.27	47.20	35.73	44.27	41.73	**46.67**	40.80	45.73
Qwen3-4B	TREC-fine	5→10	34.93	54.13	34.80	37.73	54.67	56.67	31.87	33.33	40.40	41.33	38.93	**42.80**
Llama-3.1-8B	banking77	1→2	54.40	70.27	43.33	54.13	53.73	70.40	55.60	62.40	48.00	61.87	49.20	**62.53**
Llama-3.1-8B	clinc150	1→2	69.07	83.07	55.73	65.33	66.80	79.20	71.87	**77.20**	58.93	76.40	59.47	76.40
Llama-3.1-8B	dbpedia14	2→5	71.20	93.33	69.07	70.40	92.00	92.80	76.13	**91.33**	83.20	88.93	84.00	89.47
Llama-3.1-8B	dbpedia14	5→10	90.93	95.33	87.33	89.20	95.33	95.33	93.07	93.47	91.47	93.47	92.27	**94.00**
Llama-3.1-8B	yahoo	2→5	38.27	60.00	29.60	30.67	51.47	53.47	42.00	**48.80**	35.47	39.60	38.80	42.13
Llama-3.1-8B	yahoo	5→10	57.73	64.27	51.20	51.60	62.13	60.67	59.87	**61.33**	54.67	55.33	56.13	55.20
Llama-3.1-8B	yelp	2→5	21.33	28.40	20.13	20.40	22.27	27.20	20.27	22.27	22.00	24.93	22.67	**27.60**

Table 1 compares the proposed head sets with equally sized sets ranked by TSLA-TL (Yang et al., 2026), ranked by attention share, or sampled at random. Each set is evaluated through its direct output under selective memory access. This comparison tests which readers provide useful class scores at a fixed head budget. TSLA-TL supplies a ranking for this common readout; its vector-injection results are evaluated separately in Section 3.3.

Across all four settings and both head budgets, the proposed sets achieve the highest direct-output accuracy among the compared selections. On TREC-fine, their advantage over TSLA-TL ranges from 14.67 to 25.87 percentage points. They also exceed attention-share selection in every comparison, including Llama/TREC-fine with thirty heads (47.47% versus 38.00%). On dbpedia14, they outperform TSLA-TL by 2.80 and 4.13 points with eight and thirty heads, respectively, with smaller margins than on TREC-fine. These comparisons support the paired mapping criterion for identifying useful direct readers across the tested settings. Appendix D.2 separates initial readout accuracy from the gain due to memory access and reports set overlaps.

3.3 Selective Memory Reuse: Accuracy and Efficiency

Selective TL Memory separates reusable demonstration keys and values from live receiver queries and their retrieval weights (Section 2). The controls below test which predictive benefits survive this separation and how they depend on the surrounding computation.

Table 3: Comparison with vector baselines: test accuracy (%) over three seeds and 250 queries per seed. Natural and All heads provide base and full access, respectively. Bold marks the best adapted-method score in each row, excluding these two references.

					Baselines				TSLA		Ours (direct)	
Model	**Task**	**Level**	Natural	All heads	FV	TV	ICV	I2CL	TL	TR	8	30
Llama-3.1-8B	TREC-fine	2→5	36.93	51.60	22.27	36.80	32.93	41.07	32.67	12.40	44.40	**47.47**
Llama-3.1-8B	TREC-fine	5→10	49.33	58.27	30.27	48.40	47.60	46.93	39.20	19.07	51.60	**53.07**
Llama-2-7B + SelfExtend	TREC-fine	2→5	14.93	34.80	11.07	15.47	12.40	17.07	15.60	13.87	14.93	**21.07**
Llama-2-7B + SelfExtend	TREC-fine	5→10	34.13	39.60	22.40	33.87	**34.00**	24.00	25.87	28.67	24.67	31.87
Qwen3-14B	TREC-fine	2→5	40.27	55.20	30.00	41.07	38.67	40.27	17.73	10.53	45.07	**49.07**
Qwen2.5-7B	TREC-fine	2→5	33.60	48.00	30.93	32.80	24.67	20.80	27.60	0.67	41.33	**43.47**
Qwen2.5-7B	TREC-fine	5→10	38.53	59.47	39.60	40.00	36.93	32.80	33.33	0.80	45.87	**52.67**
Qwen3-8B	TREC-fine	2→5	37.33	57.60	21.60	37.47	37.87	22.13	5.20	5.07	40.80	**43.47**
Qwen3-8B	TREC-fine	5→10	33.60	62.53	24.27	33.73	34.13	21.60	5.60	4.40	37.60	**42.27**
Qwen3-4B	TREC-fine	2→5	33.60	42.27	29.73	33.33	35.87	21.33	4.53	12.13	41.73	**46.67**
Qwen3-4B	TREC-fine	5→10	34.93	54.13	34.13	36.40	40.13	36.53	16.27	23.87	40.40	**41.33**
Llama-3.1-8B	banking77	1→2	54.40	70.27	53.33	53.87	51.20	39.60	10.40	16.40	48.00	**61.87**
Llama-3.1-8B	clinc150	1→2	69.07	83.07	38.53	67.87	67.47	53.60	35.07	33.20	58.93	**76.40**
Llama-3.1-8B	dbpedia14	2→5	71.20	93.33	58.40	68.80	70.80	80.67	63.87	49.60	83.20	**88.93**
Llama-3.1-8B	dbpedia14	5→10	90.93	95.33	89.33	91.33	92.40	**93.87**	84.13	71.20	91.47	93.47
Llama-3.1-8B	yahoo	2→5	38.27	60.00	26.27	39.20	21.20	27.87	22.27	16.13	35.47	**39.60**
Llama-3.1-8B	yahoo	5→10	57.73	64.27	28.93	**57.60**	45.07	42.67	31.87	16.00	54.67	55.33
Llama-3.1-8B	yelp	2→5	21.33	28.40	20.27	22.27	**39.33**	37.60	22.80	24.80	22.00	24.93

Cached evidence remains useful through selective retrieval. With the carrier classifier fixed, selective increment access improves over No access in all 18 natural-language settings at both budgets, establishing the usefulness of additional visibility within the shared full-prefix encoding (Table 2). Thirty-head direct readout also gains 6.14–13.07 points over Natural across all six TREC-fine configurations at $2 \rightarrow 5$; this comparison changes both access and prediction rule.

Live retrieval versus fixed-vector reuse. Vector baselines inject fixed demonstration representations; our method retains live retrieval, motivated by the mean-weight control (Figure 3A). Thirty-head direct readout exceeds every adapted vector baseline in 14 of 18 settings (Table 3). For TSLA, subtracting the base-context mean largely removes full-vector injection losses but leaves accuracy near Natural (Appendix A.2): the tested averaged-write intervention does not recover the live readout's gain. On yelp, however, ICV and I2CL exceed even All heads, showing that the preferred reuse design depends on the task.

On the memory side, contextual label states retain useful evidence. In matched controls, label-only memory often preserves full-column accuracy and outperforms the non-label complement (Appendix D.9). This connects the diagnostic concentration of class votes at label positions to compact reuse. These states are constructed from full demonstrations before increment access is restricted to label columns.

On the query side, reuse depends on the receiver. Base-access and other-head controls keep cached demonstration states fixed while changing receiver computation. Retaining base demonstrations improves both selective readouts at both budgets in all four tested settings with $K_{\text{full}} = 5$ fixed (Appendix D.1). Full access also exceeds selective access in all 36 comparisons. On Qwen3-8B/TREC-fine at $5 \rightarrow 10$, the same eight-head readout reaches 60.53% when only other heads access the increment, versus 37.60% when only carriers do (Table 20). The surrounding network thus helps construct the states through which readers exploit cached evidence.

Reading out versus reading through. Direct readout uses collected carrier writes; model readout uses final logits. On Qwen2.5-7B/TREC-fine at $5 \rightarrow 10$, selective access raises thirty-head direct accuracy from 41.20% to 52.67%, while final-model accuracy falls from 38.53% to 29.60%. In some settings, downstream computation improves on direct prediction (Appendix D.8). Readable evidence and its use in final decisions can therefore diverge.

Selective reuse limits online cost. With eight carriers on Llama-3.1-8B/TREC-fine at $5 \rightarrow 10$, prefix-cached direct readout adds 0.009 GiB shared across queries (0.001 GiB for anchors), with SDPA latency and throughput close to standard $K = 5$. Resident storage is 0.377 GiB versus

0.742 GiB for standard $K = 10$. These measurements connect selective evidence reuse to modest online overhead on one 40 GB A100, excluding one-time preparation (Table 6; Appendix A.3).

4 Related Work

Mechanisms of in-context task learning. Demonstrations can identify a familiar task or supply a mapping that pretrained knowledge and other task cues do not determine (Xie et al., 2022; Min et al., 2022; Pan et al., 2023). Many-shot studies give behavioral evidence for the latter: models can follow arbitrary or flipped label mappings with additional demonstrations (Wei et al., 2023; Agarwal et al., 2024), and long-context ICL has been probed through retrieval and attention controls over which demonstrations are visible (Bertsch et al., 2025). The head-level analyses closest to ours develop their main evidence with four to sixteen demonstrations (Yu & Ananiadou, 2024; Yang et al., 2026; Sun et al., 2025); motivated by the demonstration demand in Figure 1, our diagnostics use 70–180. Kernel accounts describe ICL as a weighted read over demonstration representations (Han et al., 2025; Yu & Ananiadou, 2024), and TSLA and PC patching locate heads involved in applying input–label mappings through label-subspace outputs or patching (Yang et al., 2026; Sun et al., 2025). We select heads by a paired label contrast: TSLA's TL score normalizes the correct-versus-incorrect gap within one forward pass, whereas the paired subtraction removes class, format, and positional contributions unchanged by relabeling (Appendix B.2). Query-weight, order, and value-intervention controls then test whether the selected heads retrieve the demonstrated mapping, providing empirical evidence beyond the kernel decomposition. Label-anchor analyses show that contextual label states gather demonstration information (Wang et al., 2023; Cho et al., 2025); we examine this concentration at many-shot scale and use it to construct a compact memory interface (Section 2.4).

Reusing demonstrations: steering vectors and cached states. Task vectors, function vectors, ICV, I2CL, and TSLA reuse demonstration-derived representations through activation interventions (Hendel et al., 2023; Todd et al., 2024; Liu et al., 2024b; Li et al., 2025b; Yang et al., 2026). Our mean-weight control shows that averaging retrieval weights across queries substantially reduces carrier-readout accuracy (Figure 3A), motivating live retrieval over cached demonstration states and comparison with these methods as adapted baselines. Receiver conditioning also matters for adaptation. In our TSLA controls on Llama-3.1-8B/TREC-fine, full- and base-context mean writes overlap strongly; base subtraction recovers most of the loss from full-vector addition (Appendix A.2). A fixed vector can still act query-dependently through the remaining network, and on some tasks vector interventions are the better choice (Table 3). Prefix caching reuses precomputed contextual states (Gim et al., 2024), while KV-cache reduction and head-dependent cache budgets reduce retained context using criteria such as attention mass or retrieval-versus-streaming behavior (Zhang et al., 2023; Li et al., 2024; Xiao et al., 2024; 2025b). Selective TL Memory focuses on which cached demonstration evidence mapping-sensitive readers use. Its head and position restrictions come from the diagnosed computation, and access and receiver controls test the usefulness of retained evidence. We measure costs within a prefix-cached benchmark that benefits every method. Work on self-repair also motivates examining component-level evidence alongside final behavior (McGrath et al., 2023; Rushing & Nanda, 2024), a distinction reflected in our direct and model readouts. Further connections are in Appendix F.

5 Conclusion

Our analysis identifies mapping-sensitive carrier heads whose writes support direct classification through query-dependent retrieval of contextual class votes. Selective TL Memory reuses this evidence by caching contextual demonstration states for live retrieval through selected readers. Across models and tasks, small reader sets provide useful predictions, while selective access and label-state retention support compact evidence reuse. The access and receiver controls show that this concentrated readout depends on the surrounding network's construction of contextual states. Gains in carrier classification need not translate into better final-model decisions. These findings connect the kernel-classifier account to repeated prediction with fixed model parameters and modest online overhead. Future work can investigate how useful reader states are constructed and why the benefits of selective reuse vary across models, tasks, and reader budgets.

AI USE STATEMENT

In this work, we used generative AI tools to assist with editing and polishing the paper, primarily using it as a tool to improve the wording, grammar, flow, and organization of the paper. We also used generative AI for developing figures. We have not used generative AI tools for developing code or ideas presented in this work. We have reviewed all AI-assisted work and revised all suggested writing improvements to ensure it accurately reflects the ideas, results and conclusions we work to present. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Rishabh Agarwal, Avi Singh, Lei M Zhang, Bernd Bohnet, Luis Rosias, Stephanie C.Y. Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, John D Co-Reyes, Eric Chu, Feryal Behbahani, Aleksandra Faust, and Hugo Larochelle. Many-shot in-context learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=AB6XpMzvqH.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0g0X4H8yN4I.

Amanda Bertsch, Maor Ivgi, Emily Xiao, Uri Alon, Jonathan Berant, Matthew R. Gormley, and Graham Neubig. In-context learning with long-context models: An in-depth exploration. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 12119–12149, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.605. URL https://aclanthology.org/2025.naacl-long.605/.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In Tsung-Hsien Wen, Asli Celikyilmaz, Zhou Yu, Alexandros Papangelis, Mihail Eric, Anuj Kumar, Iñigo Casanueva, and Rushin Shah (eds.), *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pp. 38–45, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlp4convai-1.5. URL https://aclanthology.org/2020.nlp4convai-1.5/.

Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers. *Advances in neural information processing systems*, 35:18878–18891, 2022.

Ting-Yun Chang, Jesse Thomason, and Robin Jia. When parts are greater than sums: Individual llm components can outperform full models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10280–10299, 2024.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=kp1U6wBPXq.

Hakaze Cho, Mariko Kato, Yoshihiro Sakai, and Naoya Inoue. Revisiting in-context learning inference circuit in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xizpnYNvQq.

Joy Crosbie and Ekaterina Shutova. Induction heads as an essential mechanism for pattern matching in in-context learning. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5049–5111, Albuquerque,

New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.283. URL https://aclanthology.org/2025.findings-naacl.283/.

Yuxin Dong, Jiachen Jiang, Zhihui Zhu, and Xia Ning. Understanding task vectors in in-context learning: Emergence, functionality, and limitations. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=CLBVilFk7N.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, December 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3816–3830, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.295. URL https://aclanthology.org/2021.acl-long.295/.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in neural information processing systems*, 35:30583–30598, 2022.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 2591–2607, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/0b276510ec2d3f6613a8b60c41ff0438-Paper-Conference.pdf.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=F1G7y94K02.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt cache: Modular attention reuse for low-latency inference. *Proceedings of Machine Learning and Systems*, 6:325–338, 2024.

Zhiren Gong, Zihao Zeng, Chau Yuen, and Wei Yang Bryan Lim. Conditional co-ablation: Recovering self-repair backups in transformer circuits. *arXiv preprint arXiv:2607.01940*, 2026.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In *International Conference on Learning Representations*, volume 2025, pp. 97114–97144, 2025.

Chi Han, Ziqi Wang, Han Zhao, and Heng Ji. Understanding emergent in-context learning from a kernel regression perspective. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=6rD50Q6yYz.

Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9318–9333, 2023.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. Llm maybe longlm: Selfextend llm context window without tuning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. An evaluation dataset for intent classification and out-of-scope prediction. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1311–1316, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1131. URL https://aclanthology.org/D19-1131/.

Xin Li and Dan Roth. Learning question classifiers. In *International Conference on Computational Linguistics*, 2002.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970, 2024.

Yuxuan Li, Declan Iain Campbell, Stephanie C.Y. Chan, and Andrew Kyle Lampinen. Just-in-time and distributed task representations in language models. In *Mechanistic Interpretability Workshop at NeurIPS 2025*, 2025a. URL https://openreview.net/forum?id=KC13tBPbQg.

Zhuowei Li, Zihao Xu, Ligong Han, Yunhe Gao, Song Wen, Di Liu, Hao Wang, and Dimitris N. Metaxas. Implicit in-context learning. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=G7u4ue6ncT.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

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 *Forty-first International Conference on Machine Learning*, 2024b. URL https://openreview.net/forum?id=dJTChKgv3a.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8086–8098, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.556. URL https://aclanthology.org/2022.acl-long.556/.

Callum Stuart McDougall, Arthur Conmy, Cody Rushing, Thomas McGrath, and Neel Nanda. Copy suppression: Comprehensively understanding a motif in language model attention heads. In Yonatan Belinkov, Najoung Kim, Jaap Jumelet, Hosein Mohebbi, Aaron Mueller, and Hanjie Chen (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 337–363, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.22. URL https://aclanthology.org/2024.blackboxnlp-1.22/.

Thomas McGrath, Matthew Rahtz, Janos Kramar, Vladimir Mikulik, and Shane Legg. The hydra effect: Emergent self-repair in language model computations. *arXiv preprint arXiv:2307.15771*, 2023.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 11048–11064, 2022.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=2DtxPCL3T5.

Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1): 141–142, 1964.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8298–8319, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.527. URL https://aclanthology.org/2023.findings-acl.527/.

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. Parallel context windows for large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6383–6402, 2023.

Jie Ren, Qipeng Guo, Hang Yan, Dongrui Liu, Quanshi Zhang, Xipeng Qiu, and Dahua Lin. Identifying semantic induction heads to understand in-context learning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6916–6932, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.412. URL https://aclanthology.org/2024.findings-acl.412/.

Cody Rushing and Neel Nanda. Explorations of self-repair in language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=5ZwEifshyo.

Aaditya K Singh, Ted Moskovitz, Felix Hill, Stephanie C.Y. Chan, and Andrew M Saxe. What needs to go right for an induction head? A mechanistic study of in-context learning circuits and their formation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 45637–45662. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/singh24c.html.

Chenghao Sun, Zhen Huang, Yonggang Zhang, Le Lu, Houqiang Li, Xinmei Tian, Xu Shen, and Jieping Ye. Interpret and improve in-context learning via the lens of input-label mappings. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3873–3895, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.196. URL https://aclanthology.org/2025.acl-long.196/.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=F7aAhfitX6.

Sebastian Thrun, Jerzy W. Bala, Eric Bloedorn, Ivan Bratko, Bojan Cestnik, John Cheng, Kenneth A. De Jong, Saso Dzeroski, Douglas H. Fisher, Scott E. Fahlman, Rainer Hamann, Kenneth A. Kaufman, Stefanie Keller, Igor Kononenko, Juergen S. Kreuziger, Ryszard S. Michalski, T. A. Mitchell, Peter W. Pachowicz, Haleh Vafaie, Walter Van de Welde, Walter Wenzel, Janusz Wnek, and Jian-Ping Zhang. The monk"s problems-a performance comparison of different learning algorithms, cmu-cs-91-197, sch. 1991.

Eric Todd, Millicent 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.

William Lingxiao Tong and Cengiz Pehlevan. MLPs learn in-context on regression and classification tasks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=MbX0t1rUlp.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9840–9855, 2023.

Geoffrey S Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 359–372, 1964.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EytBpUGB1Z.

Emily Xiao, Chin-Jou Li, Yilin Zhang, Graham Neubig, and Amanda Bertsch. Efficient many-shot in-context learning with dynamic block-sparse attention. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31946–31958, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1542. URL https://aclanthology.org/2025.acl-long.1542/.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. In *International Conference on Learning Representations*, volume 2025, pp. 37228–37253, 2025b.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

Haolin Yang, Hakaze Cho, and Naoya Inoue. Localizing task recognition and task learning in in-context learning via attention head analysis. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=gdvOF1OMa7.

Kayo Yin and Jacob Steinhardt. Which attention heads matter for in-context learning? In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=C7XmEByCFv.

Zeping Yu and Sophia Ananiadou. How do large language models learn in-context? query and key matrices of in-context heads are two towers for metric learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3281–3292, 2024.

Ruiqi Zhang, Jingfeng Wu, and Peter Bartlett. In-context learning of a linear transformer block: benefits of the mlp component and one-step gd initialization. *Advances in Neural Information Processing Systems*, 37:18310–18361, 2024.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in neural information processing systems*, 36:34661–34710, 2023.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pp. 12697–12706. Pmlr, 2021.

A Experimental Details

A.1 Datasets and Splits

Task families and mapping demands. Consider a fixed pretrained model M_θ and a task family $\mathcal{F} = \{f_z : \mathcal{X} \to \mathcal{Y}\}_{z \in \mathcal{Z}}$, where z specifies a decision rule or label assignment. The model receives demonstrations $D = \{(x_i, y_i)\}_{i=1}^N$, with $y_i = f_z(x_i)$ in the noise-free case, and predicts the output for an unseen input. Let B denote the remaining task cues, including instructions, the input distribution, the label vocabulary, and the rendering template.

Following the distinction between task recognition and task learning (Pan et al., 2023), this definition concerns the information supplied by demonstrations. In arbitrary-label classification, the target is $f_\pi(x) = \pi(g(x))$: the semantic classification function g may be familiar, while demonstrations identify the correspondence π between classes and label tokens (Wei et al., 2023). This setting retains pretrained semantic representations while requiring the class–token correspondence to be inferred from demonstrations. Appendix E extends the evaluation to rules over categorical attributes and gives their construction.

Natural-language tasks. Table 4 gives the six classification datasets and their rendering templates. Method evaluation retains classes that appear in both training and test data and have at least fifteen distinct training rows. TREC-fine therefore uses 36 of its 50 fine classes; all classes of the other five tasks are eligible. Demonstrations and method-evaluation queries are drawn from eligible classes.

Table 4: The natural-language tasks. Sizes are the native training / test splits; the block template shows one demonstration, with the label slot carrying the abstract token.

Task	Source	Train	Test	Classes (eligible)	Block template
TREC-fine	Li & Roth (2002) (fine labels)	5,452	500	50 (36)	`Question: {text}\nType: {label}`
banking77	Casanueva et al. (2020)	10,003	3,080	77 (77)	`Query: {text}\nIntent: {label}`
clinc150	Larson et al. (2019), in-scope intents	15,000	4,500	150 (150)	`Query: {text}\nIntent: {label}`
dbpedia14	Zhang et al. (2015)	560,000	70,000	14 (14)	`Title: {title}\nDescription: {content}\nType: {label}`
yahoo	Zhang et al. (2015)	1,400,000	60,000	10 (10)	`Text: {title + content + answer}\nTopic: {label}`
yelp	Zhang et al. (2015), five stars	650,000	50,000	5 (5)	`Review: {text}\nRating: {label}`

Labels and prompts. Natural class names are not displayed. Calibration selects a short alphabetic string for each class whose first appended token after the label cue is distinct and is a single token. The candidate pool includes upper- and lower-case letters and additional short strings for tasks with more classes. Strings matching a task's own label surface are excluded. The class-to-token assignment is fixed for a model–task cell and reused across seeds and demonstration budgets. A prompt consists of a BOS token, demonstration blocks separated by blank lines, and the query block ending at its label cue.

Sampling and selection. Prefix seeds are 42, 43, and 44. Before sampling demonstrations, each seed reserves four training examples per eligible class for validation, except clinc150 (two); the supplementary MONK tasks reserve twelve (Appendix E.1). Demonstrations are sampled without replacement, with K per class, then shuffled. A base set is nested within its full demonstration set. Test evaluation draws 250 eligible queries per seed using a content-hash ordering and uses the same draw for every method. The 750 query–seed evaluations can contain repeated queries across seeds. The 250-query test draw follows the evaluation size used by PCW (Ratner et al., 2023). Head sets and tunable baseline settings are selected using validation data and frozen before test evaluation.

A.2 BASELINE: TSLA

All adapted vector baselines act on the matched base-access receiver and predict through the model's final logits. Unless otherwise stated, we follow their default settings. For TSLA, we retain the top $\lfloor 0.03 \cdot LH \rfloor$ heads according to its default setting, which corresponds to 30 heads for 32-layer, 32-head models. This motivates us to use a budget of 30 heads for a fair comparison.

Full-vector addition and an increment control. TSLA constructs a task vector by averaging the final-position outputs of selected heads over demonstration prompts, then adds this vector to a zero-shot query's residual stream at the midpoint layer (Yang et al., 2026). Zero-base controls across models and tasks motivate retaining base demonstrations to condition the receiver (Appendix D.1). Applying the same operator to this receiver raises a different question: does the vector supply incremental task evidence, or largely reintroduce a component already produced by the base context? This can validate the way we construct Ours (model) variant.

Let $\mathcal{H}$ be the TSLA-selected head set and let $\bar{v}_{\text{full}}$ and $\bar{v}_{\text{base}}$ be its summed residual-space outputs averaged over the same 50 validation inputs per seed, with full and matched base-only access, respectively. For the receiver's answer-row state $h_m(q)$ at the injection layer, we compare

$$h_m^{\text{full}}(q) = h_m(q) + \bar{v}_{\text{full}}, \tag{6}$$

$$h_m^{\text{inc}}(q) = h_m(q) + (\bar{v}_{\text{full}} - \bar{v}_{\text{base}}). \tag{7}$$

The second arm removes the base-context mean from the injected vector. It retains the native receiver computation and does not delete or replace a whole layer. Both injected vectors are fixed across test queries. For the TL vectors in the Llama-3.1/TREC validation measurements, the full and base vectors have cosine similarity 0.978–0.994 across the two expansion levels and three seeds; the difference has 11.9–21.1% of the full vector's norm. These quantities show substantial overlap, not equality of the full and base representations.

Removing the shared component recovers much of the loss. Table 5 compares full-vector and increment-vector injection. At $2 \to 5$, the TL vector's accuracy changes from 32.67% to 38.53% after base subtraction, against Natural's 36.93%; at $5 \to 10$, it changes from 39.20% to 49.47%, against 49.33%. The TR vector's much larger losses are also largely removed. For the TL increment arm, the paired net changes against Natural are $+12$ predictions ($p = 0.088$) and $+1$ ($p = 1.0$). Thus the correction can preserve or modestly improve the base decision, although these two test comparisons do not establish a reliable accuracy gain. Its $2 \to 5$ point estimate is also comparable to eight-head Ours (model), at 38.40%.

The vector geometry and subtraction control are consistent with full-vector addition reintroducing a large shared component. They do not isolate final normalization or any particular downstream module as the cause of the loss. Additive accounting of head writes within one forward pass does not make adding an average write from another context equivalent to the effect of adding demonstrations. The supported conclusion is that unmodified full-vector addition is unreliable for isolating incremental evidence in this TL expansion setting.

A.3 MODELS, AND COST MEASUREMENTS

Models and precision. We use Llama-3.1-8B, Qwen2.5-7B, and Qwen3-4B-, 8B-, and 14B-Base, together with Llama-2-7B plus SelfExtend (Jin et al., 2024) (group size four, neighbor window 1024). Method evaluations use bf16.

Configuration	$2 \to 5$	$5 \to 10$
Natural	36.93	49.33
TSLA-TL, full vector	32.67	39.20
TSLA-TL, increment vector	38.53	49.47
TSLA-TR, full vector	12.40	19.07
TSLA-TR, increment vector	36.67	48.27
Random heads, full vector	35.73	48.53
Random heads, increment vector	37.20	48.67

Table 5: TSLA injection controls on Llama-3.1-8B and TREC-fine: final-model test accuracy (%) over 750 query–seed evaluations. The head set, extraction inputs, receiver, and injection location are held fixed within each full/increment pair. The increment arm subtracts the same heads' base-access mean output.

Carrier discovery and access. Each validation query is evaluated once with the intact prefix and five times with randomized demonstration labels. The raw TL-score is averaged across validation queries; heads are ranked with ties resolved by layer and head index. The masks are applied at every receiver position, and grouped-query attention shares storage without granting unselected query heads increment visibility. Thirty heads match TSLA's top-three-percent budget on 32-layer, 32-head models; TSLA uses the corresponding percentage on other architectures (Yang et al., 2026).

Cost measurement protocol. Table 6 reports the inference cost. In the *full-prompt* setting, each query re-encodes the $K = 5$ base prompt for vector baselines and Ours; Ours also reads a prepared $5 \to 10$ increment through the carriers. The anchor variant retains only the increment's label columns. Standard $K = 10$ re-encodes all ten demonstrations per class. In the *prefix-cached* setting, every method reuses its demonstration cache and online execution processes only the query suffix. Prefix caching thus benefits all methods; comparisons within the cached setting evaluate the method's additional storage and execution savings.

After-load memory measures loaded model allocations; resident memory counts reusable caches and intervention objects; peak memory and latency are measured at batch size one. Throughput is reported separately for batch sizes one, eight, and thirty-two. OOM indicates failure at that batch size. Timings exclude discovery, vector calibration, and the one-time full-prefix encoding used to construct memory. The cost of carrier discovery can be shared across memory levels using the same ranking, similar to other baselines.

Storage, latency, and throughput. The additional memory shared across queries is small: 0.009 GiB for direct readout and 0.001 GiB for anchors (Table 6). Online latency and throughput remain close to standard $K = 5$ prompting, while the compact readout implementations also reduce model memory. Standard $K = 10$ prompting requires a larger demonstration cache and delivers lower batched throughput. This cost gap motivates intervention methods, including ours and the vector baselines, that reuse additional demonstration evidence with modest online overhead. For demonstration prefixes approaching a million tokens, reducing retained memory and per-query attention could become substantially more valuable, especially when the same memory serves many queries.

B Mechanism Diagnostics

B.1 Primary Llama-3.1 Diagnostic Cohort and Readout Conventions

The detailed diagnostic uses Llama-3.1-8B with ordinary attention, bf16, TREC-fine, $K = 5$, prefix seed 42, and 250 queries. Its shared prefix contains 180 demonstrations from 36 eligible classes in 3144 tokens including BOS; the demonstration block excludes BOS and contains 3143 tokens. Complete prompts contain 3152–3168 tokens.

Configuration	GPU memory (GiB)			Latency (ms)	Throughput (queries/s)		
	After load	Resident	Peak, $B{=}1$	$B{=}1$	$B{=}1$	$B{=}8$	$B{=}32$
(a) Eager attention							
Full prompt							
Standard, $K{=}5$	15.0	0	18.4	821.4	1.22	OOM	OOM
Standard, $K{=}10$	15.0	0	28.7	2404.0	0.42	OOM	OOM
TSLA	15.0	0	18.4	826.6	1.21	OOM	OOM
Function vectors	15.0	0	18.4	830.1	1.21	OOM	OOM
Task vectors	15.0	0	18.4	831.7	1.20	OOM	OOM
In-context vectors	15.0	0	18.4	871.0	1.15	OOM	OOM
I2CL	15.0	0	18.4	854.8	1.17	OOM	OOM
Ours (direct)	13.7	0.009	17.7	839.6	1.19	OOM	OOM
Ours (anchor)	13.7	0.001	17.2	829.6	1.21	OOM	OOM
Prefix cached							
Zero-shot (standard)	15.0	0	15.0	31.3	31.9	239.6	408.4
Standard, $K{=}5$	15.0	0.368	15.4	35.0	28.6	131.4	148.7
Standard, $K{=}10$	15.0	0.742	15.7	34.6	28.9	75.2	81.9
TSLA	15.0	0.368	15.4	35.1	28.5	131.5	148.7
Function vectors	15.0	0.368	15.4	35.1	28.5	131.2	148.6
Task vectors	15.0	0.368	15.4	34.9	28.7	131.5	148.5
In-context vectors	15.0	0.369	15.4	44.9	22.3	121.8	142.8
I2CL	15.0	0.369	15.4	38.5	26.0	128.6	145.6
Ours (direct)	13.7	0.377	14.1	35.9	27.8	130.8	149.2
Ours (anchor)	13.7	0.369	14.1	36.2	27.6	132.7	150.5
(b) SDPA attention							
Full prompt							
Standard, $K{=}5$	15.0	0	15.4	295.5	3.38	3.58	3.59
Standard, $K{=}10$	15.0	0	15.9	614.0	1.63	1.69	OOM
TSLA	15.0	0	15.4	296.3	3.38	3.58	3.59
Function vectors	15.0	0	15.4	296.7	3.37	3.57	3.58
Task vectors	15.0	0	15.4	296.8	3.37	3.57	3.58
In-context vectors	15.0	0	15.4	331.6	3.02	3.20	3.21
I2CL	15.0	0	15.4	315.9	3.17	3.36	3.37
Ours (direct)	13.7	0.009	14.1	293.6	3.41	3.64	3.64
Ours (anchor)	13.7	0.001	14.1	292.5	3.42	3.64	3.64
Prefix cached							
Zero-shot (standard)	15.0	0	15.0	23.7	42.2	315.0	417.3
Standard, $K{=}5$	15.0	0.368	15.3	29.5	33.9	173.0	221.8
Standard, $K{=}10$	15.0	0.742	15.7	36.4	27.5	121.2	139.9
TSLA	15.0	0.368	15.3	29.6	33.8	173.1	221.9
Function vectors	15.0	0.368	15.3	29.5	33.9	173.4	221.0
Task vectors	15.0	0.368	15.3	29.5	33.9	173.1	221.2
In-context vectors	15.0	0.369	15.4	39.4	25.4	156.4	209.3
I2CL	15.0	0.369	15.4	33.0	30.3	167.1	216.9
Ours (direct)	13.7	0.377	14.1	30.4	32.9	167.3	217.8
Ours (anchor)	13.7	0.369	14.1	30.3	33.0	171.3	219.8

Table 6: Inference cost on TREC-fine with Llama-3.1-8B (bf16, one 40 GB A100). Vector baselines and Ours use $5 \to 10$, with eight carriers for Ours. *Full prompt* re-encodes the demonstration prefix per query; for Ours, this is the base prefix, while the increment remains cached. *Prefix cached* reuses the demonstration prefix. Resident memory counts reusable caches and intervention objects. Timings exclude one-time preparation; measurement details are in Appendix A.3. OOM: out of memory.

B.2 Controlling Label Preferences and Positional Effects

Mapping sensitivity is distinct from a large label-directed write. The discovery contrast uses the same abstract label vocabulary, demonstration inputs, query, and token positions on both sides. Only the displayed demonstration labels change. To see which biases this controls, write a head's score as

$$s_c^{(\ell,h)}(q; D) = b_c^{(\ell,h)}(q, p) + m_c^{(\ell,h)}(q; D), \tag{8}$$

where p denotes the fixed positional configuration and $b_c^{(\ell,h)}$ is any contribution unchanged by the label intervention. This term can include a class preference or a format- or position-dependent offset. It cancels in Equation 2 before averaging over queries. A large unchanged write therefore does not by itself increase the localization score. In contrast, TSLA's TL score normalizes the correct-versus-incorrect label gap within a forward pass (Yang et al., 2026); that statistic does not perform this paired subtraction. Static class-subspace alignment likewise identifies where a head can write, while the mapping contrast measures how its write responds to demonstrated relationships (Appendix C).

Diagnostic	Llama s42	Llama s43	Qwen3 8B	Qwen3 4B	dbpedia14
Block readout, query's own weights (%)	50.4	46.8	55.6	44.4	92.0
Block readout, across-query mean weights (%)	11.6	4.0	22.8	2.0	7.2
Block readout, permuted query weights (%; 200 draws)	6.4 ± 1.5	6.5 ± 1.4	10.3 ± 1.8	6.4 ± 1.5	7.2 ± 1.7
Variance left by a separable weight fit	0.822	0.857	0.644	0.491	0.942
Variance left after accounting for query length	0.797	0.825	0.617	0.465	0.934
Own-class retrieval lift, original order	5.60	6.47	5.22	4.87	6.24
First permuted order	4.59	7.10	4.81	3.61	6.99
Second permuted order	5.86	6.34	4.46	3.61	6.00
Cross-order correlation by demonstration identity	0.630–0.684	0.607–0.689	0.401–0.446	0.344–0.370	0.776–0.812
Cross-order correlation by source slot	0.068–0.111	0.013–0.046	0.083–0.113	0.021–0.084	−0.017–0.075
Label positions / demonstration-block tokens	180/3143 (5.73%)	180/3071 (5.86%)	180/3169 (5.68%)	180/3169 (5.68%)	70/5198 (1.35%)
Label positions' share of demonstration-block attention (%)	88.5	86.2	74.6	74.5	81.2
Positive own-class vote contrasts at labels	1252/1260 (99.4%)	1258/1260 (99.8%)	1248/1260 (99.0%)	1249/1260 (99.1%)	182/182 (100.0%)
Positive own-class vote contrasts over whole demonstrations	1219/1260 (96.7%)	1150/1260 (91.3%)	1244/1260 (98.7%)	1244/1260 (98.7%)	128/182 (70.3%)
Share of total attention: demonstration block (%)	57.1	55.5	59.5	65.1	49.3
Position 0 (%)	25.6	24.3	29.2	28.9	24.1
Query positions (%)	17.4	20.2	11.3	6.0	26.6

Table 7: Query-weight, order, and class-vote controls with eight heads selected per condition at $K = 5$. Llama columns use Llama-3.1-8B/TREC-fine at seeds 42 and 43; Qwen columns use TREC-fine; dbpedia14 uses Llama-3.1-8B. Permuted-weight entries give mean $\pm$ s.d. over 200 draws. Count ratios include percentages; vote counts cover ordered class pairs. Label attention is normalized within the demonstration block, which excludes position 0; the final three rows use all source positions (rounded shares).

This cancellation applies only to contributions invariant to the control. Shared label vocabulary does not remove every contextual token preference, and a head that reads labels at a fixed slot can still respond to label randomization. Consequently, the score supplies candidates for mechanism tests; it does not establish position-independent retrieval or exclude attention concentrations shared across queries (Appendix B.4). It also does not center or otherwise alter the raw direct readout used on test queries.

Position controls test what the selected readers retrieve. With a fixed demonstration order, content and source slot are confounded, and query length changes relative token distances. Table 7 therefore tests query-length effects and reorders the same demonstrations, comparing retrieval matched by demonstration identity with retrieval matched by source slot. Across five diagnostic conditions, these controls support content-sensitive retrieval while allowing interactions with position. Separately, the method comparisons hold receiver positions and cached base states fixed when changing increment visibility. This ensures that the selective-versus-no-access comparison tests access under a common positional configuration.

B.3 Weight and Vote Controls

Table 7 reports five conditions: Llama-3.1-8B/TREC-fine with prefix seeds 42 and 43, Qwen3-8B and Qwen3-4B on TREC-fine, and Llama-3.1-8B on dbpedia14. Each uses $K = 5$ and eight heads selected separately for that condition. To describe retrieval at the demonstration level, we sum attention over each demonstration's positions and over the selected heads. The resulting query-by-demonstration matrix is used for separability, own-class mass, and order analyses; it has 180 demonstration columns for TREC-fine and 70 for dbpedia14. Prediction uses the head-indexed token-weighted votes. A query-independent control must remove query dependence in the weights multiplying each fixed per-head vote; replacing only a head-summed weight while retaining a query-dependent mixture of votes would not perform that control.

Query and order dependence. Own-weight block readout exceeds the mean- and permuted-weight controls in every condition, including both Llama prefix seeds. A separable fit of the form $\kappa_{q,d} = b_q a_d$ leaves 49.1–94.2% of weight variance unexplained; accounting for each demonstration's linear response to query length leaves 46.5–93.4%. Qwen3-4B has the smallest residual share,

allowing a larger shared component, yet its own-weight readout reaches 44.4% against 2.0% with mean weights. The residual fraction measures weight variation, not the fraction of predictive information.

Across reordered demonstrations, correlations are consistently higher by demonstration identity (0.344–0.812 across conditions) than by source slot (-0.017–0.113), and own-class lift remains 3.61–7.10 under the permuted orders. These results support content-sensitive retrieval while allowing positional effects. In the original Llama-3.1-8B/TREC-fine seed-42 cohort, the length-adjusted residual favors the query's own class (paired $t = 16.76$ on 235 queries with demonstrated classes); that class receives 14.6% of the original block mass, compared with 2.6% under uniform allocation averaged over this cohort. The added seed also contributes intervention-predictor, value-response decomposition, and joint-ablation replications below; repair and uncertainty estimates retain their separately reported scope.

Label-position evidence. Across the five conditions, label positions occupy 1.35–5.86% of demonstration-block tokens but receive 74.5–88.5% of its attention, about 13–60 times their share under uniform attention over the block. For each ordered class pair (c, c'), we average the difference between votes for c and c' over the five demonstrations of class c, after summing the selected heads. Positive contrasts cover 99.0–100.0% of pairs at labels, compared with 70.3–98.7% over whole demonstrations. These are class-conditional mean contrasts, not per-query accuracies or claims that every demonstration or head has the same sign.

The gap is largest on dbpedia14: all 182 label-position contrasts are positive, versus 128 of 182 (70.3%) over whole demonstrations. There, label positions occupy only 1.35% of the block and receive 81.2% of its attention. The Qwen gaps are smaller, at 0.3–0.4 percentage points. Thus, the direction of the class-averaged votes is more consistent at label positions in all five conditions, with varying margins. Together with the attention concentration, this motivates retaining contextual label states as compact memory. Those states are constructed from full demonstration context; the accuracy of restricting access to them is tested separately in Appendix D.9.

Attention across source positions. The final three rows of Table 7 use all source positions as the denominator: demonstrations receive 49.3–65.1% of carrier attention, position 0 receives 24.1–29.2%, and query positions receive 6.0–26.6%. These shares describe where the heads read; they do not measure signed class contributions or causal importance. The method's raw readout includes prefix and query sources as well as demonstrations.

Per-head variation. In the original Llama seed-42 cohort, six selected heads predominantly exhibit the demonstration-retrieval pattern; the two layer-31 heads put much of their class-directed write on the query's own positions. An additional early carrier slice comprises five heads from layers 12–16 within the top 128 by TL-score. This extends the query-weight test to carriers beyond the selected top eight, with the same qualitative query dependence at lower accuracy: 19.6% for its own-weight block readout, 0.8% for mean weights, and $3.4 \pm 1.1\%$ for permuted weights.

B.4 Demonstration-Level Retrieval Patterns

Figures 5–8 visualize the query-by-demonstration weights for Llama-3.1-8B on TREC-fine and dbpedia14, and Qwen3-8B and Qwen3-4B on TREC-fine. Attention is summed over eight selected heads and over either whole demonstrations or their label tokens. Rows and columns are sorted by class for display; cyan boxes mark the query's own-class demonstrations. The panels show relative kernel mass, with separate color limits capped at the 99.5th percentile.

Class preference and unequal weights within a class. The same-class blocks show preferential retrieval, especially clearly on dbpedia14. Within these blocks, individual demonstrations receive unequal weights. This heterogeneity can combine query-specific matching with preferences for particular examples or positions; it does not by itself establish that the ranking of same-class examples changes across queries. The heatmaps visualize retrieval preference, while the query-weight controls and value interventions separately test its predictive and behavioral roles.

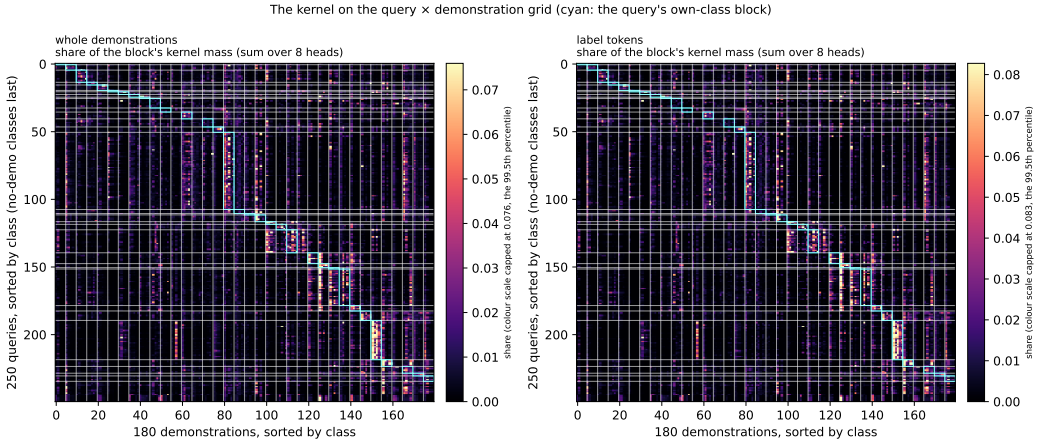


Figure 5: Demonstration-level retrieval on Llama-3.1-8B/TREC-fine. Rows are 250 queries and columns are 180 demonstrations, sorted by class; queries without a demonstrated class appear last. Cyan boxes identify same-class blocks. Left: whole-demonstration kernel mass; right: label-token kernel mass, summed over eight heads. Color limits are separately capped at the 99.5th percentile. Same-class blocks contain non-uniform weights across examples.

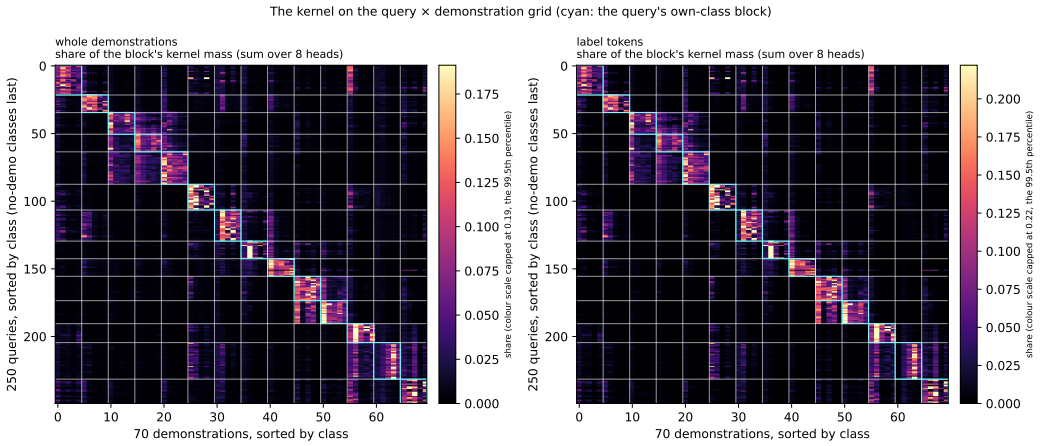


Figure 6: Demonstration-level retrieval on Llama-3.1-8B/dbpedia14, with 250 queries and seventy demonstrations. Conventions follow Figure 5. Class-selective blocks are visible in both panels, with unequal weights among the five demonstrations of each class.

Shared attention concentration in Qwen3. The Qwen3 whole-demonstration panels contain a prominent vertical stripe spanning many query classes, particularly in Qwen3-4B. This concentration coexists with class-selective retrieval. The corresponding stripe is less prominent in the label-token panels, suggesting a contribution from non-label positions. This pattern resembles the attention-sink phenomenon described by Xiao et al. (2024), in which particular tokens attract disproportionate attention across many queries. The observation is consistent with TL-score's scope: it cancels output contributions unchanged by relabeling, without requiring every attention weight of a selected head to be free of shared preferences.

B.5 Intervention Predictions and Behavioral Repair

Readout direction and intervention scope. For a demonstration whose label changes from c to c', the perturbation diagnostic measures a score contrast along $u = \gamma \odot (W_U^{c'} - W_U^{c})$, where γ is the learned weight of the final RMS normalization. The token-level identity and first-order

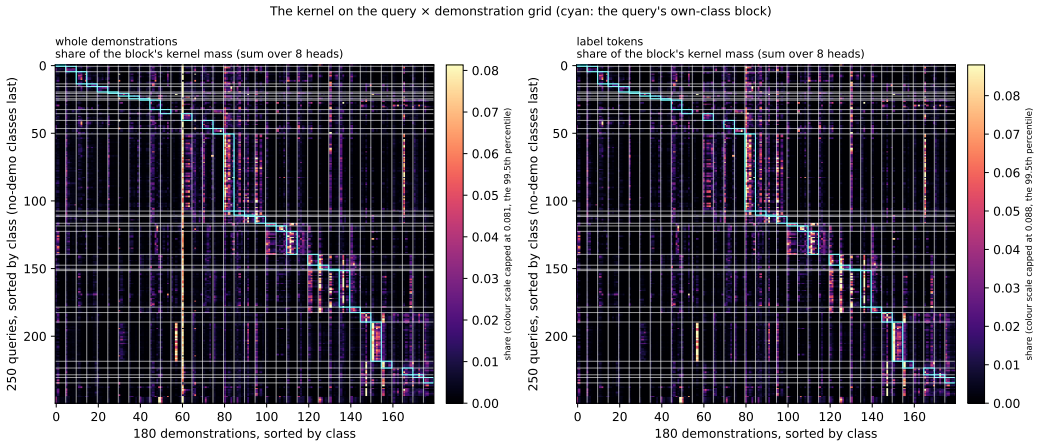


Figure 7: Demonstration-level retrieval on Qwen3-8B/TREC-fine, following Figure 5. Class-selective blocks coexist with a column that attracts attention across many query classes in the whole-demonstration panel. Its weaker appearance in the label-token panel suggests a contribution from non-label positions.

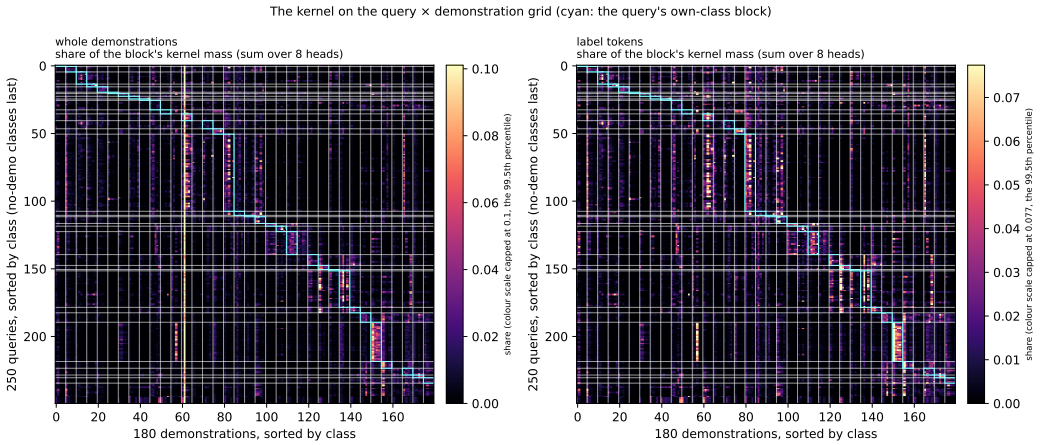


Figure 8: Demonstration-level retrieval on Qwen3-4B/TREC-fine, following Figure 5. A persistent vertical stripe is especially prominent in the whole-demonstration panel. Same-class retrieval and unequal weights within classes remain visible alongside this shared concentration.

Intervention	Readout	Predictor	Slope (95% interval)	R^2
Llama-3.1-8B / TREC-fine / seed 42				
Value patch	Carrier contrast	Query-dependent	$0.947\ [0.910, 0.981]$	0.987
		Query-independent	$0.942\ [0.901, 0.983]$	0.277
	Final logits	Query-dependent	$0.611\ [0.530, 0.688]$	0.728
		Query-independent	$0.695\ [0.601, 0.787]$	0.314
Prompt edit	Carrier contrast	Query-dependent	$0.933\ [0.785, 1.197]$	0.432
		Query-independent	$0.667\ [0.568, 0.830]$	-0.035
	Final logits	Query-dependent	$0.907\ [0.765, 1.144]$	0.400
		Query-independent	$0.657\ [0.549, 0.835]$	-0.045
Llama-3.1-8B / TREC-fine / seed 43				
Prompt edit	Carrier contrast	Query-dependent	$1.023\ [0.808, 1.308]$	0.602
		Query-independent	$0.639\ [0.471, 0.934]$	0.004
	Final logits	Query-dependent	$1.141\ [0.890, 1.464]$	0.497
		Query-independent	$0.817\ [0.589, 1.197]$	-0.020
Qwen3-8B / TREC-fine				
Prompt edit	Carrier contrast	Query-dependent	$0.742\ [0.583, 1.076]$	0.155
		Query-independent	$0.506\ [0.367, 0.795]$	-0.169
	Final logits	Query-dependent	$0.029\ [0.022, 0.042]$	0.177
		Query-independent	$0.020\ [0.015, 0.032]$	-0.148
Qwen3-4B / TREC-fine				
Prompt edit	Carrier contrast	Query-dependent	$0.880\ [0.637, 1.253]$	0.220
		Query-independent	$0.521\ [0.312, 0.852]$	-0.140
	Final logits	Query-dependent	$0.063\ [0.050, 0.083]$	0.327
		Query-independent	$0.040\ [0.030, 0.056]$	-0.094
Llama-3.1-8B / dbpedia14				
Prompt edit	Carrier contrast	Query-dependent	$1.214\ [0.977, 1.628]$	0.452
		Query-independent	$0.944\ [0.759, 1.328]$	-0.090
	Final logits	Query-dependent	$1.059\ [0.878, 1.380]$	0.448
		Query-independent	$0.920\ [0.762, 1.210]$	-0.069

Table 8: Intervention-response fits. Value-patch rows reproduce the Llama-3.1-8B/TREC-fine seed-42 reference; prompt-edit rows cover five diagnostic conditions. Each predictor is fitted through the origin, with reported 95% slope intervals and R^2 after fitting. Prompt edits change input labels and allow retrieval and contextual states to respond.

predictor apply with W_U^c replaced by any fixed direction, including u. Its corresponding model-output measurement is the final-logit contrast for the same label pair.

In the Llama-3.1-8B/TREC-fine diagnostic, both value-row probes replace whole value-projection rows in the six layers containing the eight carriers. Each layer has eight key/value heads shared by 32 query heads, so the intervention affects the value inputs of all 32 query heads in those layers. For implementation simplicity, the intervention does not isolate the eight selected carriers. We use these experiments to assess the predicted carrier response and the behavioral role of contextual values.

Predictors and estimators. The value-patch experiment uses 25 demonstrations and two replacement labels per demonstration, evaluated on 250 queries per perturbation. Donor values come from the relabeled prefix, while the receiver retains its original input tokens. For each perturbation and query, the query-dependent predictor weights the changed votes by the unperturbed attention. Averaging this prediction over queries within each perturbation yields a query-independent diagnostic surrogate; the FV baseline in Table 3 is evaluated separately.

For predictor x_j and observed response y_j, we fit a slope through the origin, $\hat{\beta} = \sum_j x_j y_j / \sum_j x_j^2$, and report $R^2 = 1 - \sum_j (y_j - \hat{\beta} x_j)^2 / \sum_j (y_j - \bar{y})^2$, which may take negative values under this constrained fit. In the original Llama-3.1-8B/TREC-fine seed-42 probe, uncertainty is estimated at the perturbation level: the 12,500 query–perturbation observations are grouped by their underlying demonstration, yielding a slope interval of $[0.910, 0.981]$ across the 25 demonstration clusters. Regression weights are uneven, with a Kish effective count of 9.57 demonstrations. Figure 3B shows 4,000 of the 12,500 swap–query pairs for readability, while the through-origin fits and reported statistics use the full set. Blue and red points show the carrier contrast and final-logit contrast, respectively, and the dotted identity line provides a common calibration reference. Differences between the carrier and final-logit responses reflect downstream processing beyond the carrier readout.

Query-specific prediction beyond class averages. Table 9 compares four versions of the first-order predictor in Equation 5: weights averaged over all queries, weights averaged over other queries of the same gold class, weights averaged over other queries of the class predicted on the

Readout	Value patch: R^2				Prompt edit: R^2			
	Query-indep.	Class (gold)	Class (model)	Per-query	Query-indep.	Class (gold)	Class (model)	Per-query
Llama-3.1-8B / TREC-fine / seed 42								
Carrier contrast	0.277	0.561	0.558	0.987	−0.035	0.179	0.169	0.432
Final logits	0.314	0.439	0.427	0.728	−0.045	0.163	0.162	0.400
Llama-3.1-8B / TREC-fine / seed 43								
Carrier contrast	0.203	0.488	0.550	0.992	0.004	0.208	0.244	0.602
Final logits	0.231	0.394	0.459	0.779	−0.020	0.177	0.217	0.497
Qwen3-8B / TREC-fine								
Carrier contrast	0.279	0.553	0.627	0.899	−0.169	0.004	0.032	0.155
Final logits	0.294	0.472	0.507	0.702	−0.148	0.032	0.057	0.177
Qwen3-4B / TREC-fine								
Carrier contrast	0.223	0.530	0.580	0.971	−0.140	0.031	0.046	0.220
Final logits	0.239	0.458	0.483	0.782	−0.094	0.109	0.132	0.327
Llama-3.1-8B / dbpedia14								
Carrier contrast	0.145	0.737	0.739	0.966	−0.090	0.324	0.324	0.452
Final logits	0.190	0.657	0.654	0.828	−0.069	0.338	0.338	0.448

Table 9: Predictor ladder for value patches replacing label-position value rows from a relabeled demonstration and prompt edits changing the displayed label token. Entries give R^2 of the observed response after fitting a through-origin gain to the first-order predictor. Query-independent averages retrieval weights over all queries; class controls average over other queries sharing the gold class or the class predicted on the clean prompt (leave-one-out); per-query retains the query's own weights. Each condition reads the eight heads selected by its rediscovery run. Carrier contrasts use the normalization-weighted diagnostic direction.

clean prompt, and each query's own weights. The two class-conditional controls use leave-one-out averages. Gold-class grouping provides an oracle class-conditioned control, while predicted-class grouping uses only the clean model output. The global-average control and the implemented FV baseline in Table 3 serve separate diagnostic and inference roles.

In all twenty condition–intervention–readout comparisons, both class means improve on the global average, while per-query weights outperform both class means. Relative to the stronger class-mean control, per-query R^2 gains are 0.227–0.442 for value-patch carrier contrasts and 0.171–0.320 for final logits; prompt-edit gains are 0.123–0.358 and 0.110–0.280, respectively. These results separate a class-shared component of retrieval from additional predictive information carried by query-specific weights. The latter can arise from within-class changes in attention mass, head mixtures, or demonstration weighting, so the fit comparison alone does not distinguish among these mechanisms. Prompt-edit fits remain partial, particularly on Qwen, despite the consistent predictor ordering.

Local value changes and downstream responses. A value-only patch leaves the retrieval weights of the first patched layer unchanged, while subsequent selected layers can adapt through the modified residual stream. Table 10 decomposes the carrier-response slope into the local value predictor, retrieval-weight changes on the perturbed demonstration, and spillover through other positions. The local term contributes 1 by construction because it defines the predictor. Total slopes remain near unity (0.8849–1.0370), with small positive retrieval contributions (0.0109–0.0322). Spillover provides the larger correction in four of five settings and is negative in those cases; seed 43 instead shows two positive corrections. Thus, the local value predictor remains well calibrated even as later layers adjust their retrieval through the evolving residual stream.

The reported maximum attention changes summarize the largest observed shifts, while the slope decomposition characterizes carrier-level responses in aggregate. Pointwise predictive variation and the mapping from carrier responses to final logits are evaluated separately. The spillover term collects downstream effects outside the perturbed demonstration and is left mechanistically unresolved here.

Table 10: Value-only intervention response decomposition. Llama denotes Llama-3.1-8B and TREC denotes TREC-fine. Layer indices are zero-based. The two attention columns give maximum absolute weight changes at the first patched carrier layer and later selected layers. Slope attribution is local value + retrieval response + spillover ≈ total; separately rounded entries may differ by 0.0001.

Setting	First layer	First $\max \lvert\Delta\alpha\rvert$	Later $\max \lvert\Delta\alpha\rvert$	Slope attribution
Llama/TREC, s42	L22	0	0.0977	$1 + 0.0236 - 0.0765 \approx 0.9471$
Llama/TREC, s43	L22	0	0.0469	$1 + 0.0322 + 0.0048 \approx 1.0370$
Qwen3-8B/TREC	L29	0	0.1802	$1 + 0.0237 - 0.1388 \approx 0.8849$
Qwen3-4B/TREC	L29	0	0.0605	$1 + 0.0133 - 0.0851 \approx 0.9283$
Llama/dbpedia14	L18	0	0.1631	$1 + 0.0109 - 0.0531 \approx 0.9578$

Direct prompt-label edits. The prompt-edit arm changes demonstration labels in the input and reruns the model. The original-kernel predictor in Equation 5 weights the changed votes at the edited demonstration using the original retrieval kernel, while the rerun permits keys, retrieval weights, and other contextual states to adapt. Table 8 extends this test across the five diagnostic conditions. After fitting a gain, query-dependent predictions achieve $R^2 = 0.155$–0.602 for carrier contrasts and 0.177–0.497 for final logits, exceeding the query-independent predictor in every condition for both readouts. This ordering recurs across the two Llama/TREC-fine seeds, both Qwen scales, and dbpedia14.

Prompt edits induce substantially broader contextual recomputation than value patches, altering the encoding on which the original-kernel predictor was defined. The persistence of predictive signal under this stronger intervention therefore provides evidence that the identified pathway remains operative beyond the narrowly controlled value-patch setting. On Qwen3-8B, for example, the carrier R^2 is 0.155 under prompt editing, compared with 0.90 in the value-patch diagnostic; although attenuated, the signal remains systematically stronger for query-dependent than query-independent prediction. Qwen3-8B and Qwen3-4B have final-logit slopes of 0.029 and 0.063, respectively, reflecting the rescaling introduced by final normalization and downstream computation (Equation 9).

Label repair. The Llama-3.1-8B/TREC-fine seed-42 repair experiment draws one set of wrong labels and corrupts nested subsets of 10, 30, 74, and 148 demonstrations. Repair leaves the corrupted label tokens in the receiver and replaces only their value rows, using donors encoded in the clean prefix and the layer scope specified above. Self repair restores each demonstration's own clean row. At 148 corruptions, same-class donors use another demonstration of the same true class, while cross-class donors use a demonstration from a different class. Keys remain unchanged. Final-model argmax is evaluated over all fifty candidate labels on the 250 diagnostic queries.

Table 11 extends the strength scan to Qwen3-8B, Qwen3-4B, and Llama-3.1-8B/dbpedia14, with 250 queries per setting. Repairs operate at the value-row level across carrier-containing layers, providing a behavioral test of the broader contextual value pathway. At the largest corruption level, all four settings improve over corruption ($p < 10^{-4}$). On TREC-fine, the repair effect becomes reliable only at the largest level, with Qwen3-4B showing a small decline at the weakest intervention; dbpedia14 instead improves across all three tested levels. The largest corruption levels correspond to different fractions of the demonstration set: 148/180 labels on TREC-fine and 70/70 on dbpedia14.

At maximum corruption, self repair recovers approximately 12–73% of the clean–corrupt accuracy gap (Table 12). In both Llama settings, self and same-class donors each outperform cross-class donors (Table 13), supporting a role for class-consistent contextual values in repair. The corresponding donor contrasts are less resolved in Qwen, where Qwen3-4B has a same-class point estimate below the cross-class estimate. Across models, the most consistent result is therefore the recovery produced by contextual-value repair, with donor-class specificity emerging more clearly in the Llama settings.

The margin diagnostic is the true-class logit minus the strongest competing candidate logit, and the reported shift compares self repair with corruption. Qwen gains accuracy even as its mean margin decreases, whereas dbpedia14 improves in both accuracy and margin and the Llama/TREC margin interval spans zero. Thus, repair can recover decision outcomes without requiring the full logit geometry to return toward the clean state. Corruption labels and their nested order are fixed within

Table 11: Label corruption and self repair, accuracy in % ($n = 250$ queries). Clean accuracies are 52.4, 56.0, 52.4, and 96.8 in row-group order. Paired p values are exact McNemar tests against the corrupt arm for the fixed corruption draw. TREC-fine has 180 demonstration labels; dbpedia14 has 70.

Setting	Corrupted labels	Corrupt accuracy	Self repair	Paired p
Llama/TREC, s42	10	49.6	49.6	1.000
	30	49.2	50.8	0.289
	74	34.0	38.4	0.071
	148	5.6	17.6	$< 10^{-4}$
Qwen3-8B/TREC	10	54.4	54.8	1.000
	30	49.2	52.0	0.143
	74	42.0	45.6	0.188
	148	7.2	16.0	$< 10^{-4}$
Qwen3-4B/TREC	10	52.4	52.0	1.000
	30	49.2	49.2	1.000
	74	36.8	40.8	0.064
	148	1.6	7.6	$< 10^{-4}$
Llama/dbpedia14	10	81.2	96.4	$< 10^{-4}$
	30	61.2	93.6	$< 10^{-4}$
	70	1.2	70.8	$< 10^{-4}$

Table 12: Repair magnitude and donor outcomes at maximum corruption. Recovery is $100(A_{\text{self}} - A_{\text{corrupt}})/(A_{\text{clean}} - A_{\text{corrupt}})$, with the reported query-bootstrap 95% interval. Donor accuracies are percentages; correct counts are out of 250. Margin shifts and their reported 95% intervals are in logit units. S/M/X denote self, same-class, and cross-class donors.

Setting	Recovery (%) [CI]	Accuracy S/M/X	Correct S/M/X	Margin shift [CI]
Llama/TREC, s42	25.6 [17.1, 34.7]	17.6/17.2/12.0	44/43/30	$+0.011$ $[-0.049, +0.072]$
Qwen3-8B/TREC	18 [11, 26]	16.0/16.4/14.0	40/41/35	-0.226 $[-0.344, -0.103]$
Qwen3-4B/TREC	12 [6, 18]	7.6/4.8/5.6	19/12/14	-0.322 $[-0.377, -0.265]$
Llama/dbpedia14	73 [67, 79]	70.8/72.0/0.8	177/180/2	$+1.558$ $[+1.464, +1.651]$

each setting, so the reported query-level intervals characterize variation over the diagnostic queries for those fixed perturbations.

In the original Llama/TREC seed-42 experiment, Figure 3C shows Wilson intervals for the individual accuracies. Paired comparisons against corruption are significant for self repair ($p < 10^{-4}$), same-class repair ($p < 10^{-4}$), and cross-class repair ($p = 0.0004$). The sham control leaves values, predictions, and the reported attention weights unchanged, providing a negative control for the targeted value-row operation.

B.6 From Carrier Contrasts to Final Logits

The diagnostic direction $u = \gamma \odot (W_U^{c'} - W_U^c)$ incorporates the learned final-normalization weights, while the input-dependent RMS denominator remains part of the final output transformation. For final residual h and logits z,

$$z^{c'} - z^c = \frac{\langle h, u \rangle}{\rho(h)}, \qquad \rho(h) = \sqrt{|h|_2^2/d + \epsilon}. \tag{9}$$

Thus, even when the carrier contrast is transmitted without additional residual changes, its final-logit expression is rescaled by $1/\rho(h)$. Changes in normalization and downstream residual computation further contribute to this gain. The fitted slope therefore characterizes the mapping from carrier scale to output scale. Notably, a small gain can coexist with substantial predictive alignment: in the reported Qwen3-8B fit, a final-logit slope near 0.02 accompanies $R^2 = 0.70$.

Downstream counterbalancing provides one plausible contributor to this rescaling. In our MLP analysis, the mapping-specific write reduces the true-class margin while increasing the nearest competing class, resembling the late-MLP counterbalancing motif reported by McGrath et al. (2023), where later MLP layers down-regulate the currently favored token. Related suppressive computations have also been identified in attention heads through copy suppression (McDougall et al., 2024),

Table 13: Paired repair diagnostics at maximum corruption. Fixed/broken counts compare each donor arm with the corrupt arm. The final three columns are exact McNemar p values between donor arms. S/M/X denote self, same-class, and cross-class donors.

Setting	S fixed/broken	M fixed/broken	X fixed/broken	M vs. S	S vs. X	M vs. X
Llama/TREC, s42	32/2	31/2	18/2	1.0	0.0013	0.0023
Qwen3-8B/TREC	23/1	24/1	19/2	1.0	0.18	0.11
Qwen3-4B/TREC	15/0	9/1	11/1	0.092	0.27	0.73
Llama/dbpedia14	176/2	179/2	2/3	0.63	4×10^{-53}	5×10^{-54}

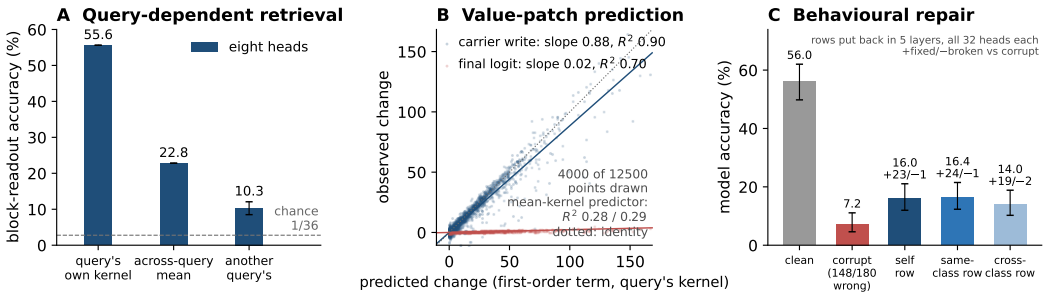


Figure 9: Mechanism checks on Qwen3-8B/TREC-fine. **(A)** Eight-head demonstration-block accuracy with own, mean, or shuffled query weights. **(B)** Predicted versus observed carrier and final-logit changes; 4,000 of 12,500 pairs are shown, lines fit all pairs through the origin, and the dotted line is the identity. No early carriers in top 128. **(C)** Final-model accuracy after corrupting 148 of 180 labels and repairing values with self, same-class, or cross-class donors; counts are corrected/broken predictions relative to corruption. Whole value rows are patched in five carrier-containing layers, affecting all 32 query heads per layer.

and broader studies find self-repair across model families through multiple mechanisms, including changes in final-normalization scale (Rushing & Nanda, 2024). These parallels suggest that the carrier signal can remain predictive while downstream computation recalibrates its expression at the output.

The value-patch panels illustrate how this output gain varies across settings. Qwen3-8B has carrier/final-logit slopes of 0.88/0.02, whereas Llama/dbpedia14 has slopes of 0.96/0.62. Their corresponding R^2 values are reported at higher precision in Table 9. Across settings, the carrier predictor can therefore remain aligned with final-logit variation even when normalization and downstream counterbalancing place that response on substantially different numerical scales.

B.7 Additional Mechanism Visualizations

Qwen3-8B/TREC-fine. Figure 9 brings together query-weight controls, value-patch responses, and repair. The numerical comparisons are consolidated in Tables 7, 9, and 12. Response fits use all 12,500 perturbation–query pairs; the panel displays 4,000. Whole-row repair affects five carrier-containing layers and all 32 query heads per layer.

Llama-3.1-8B/dbpedia14. Figure 10 provides the corresponding three-panel view on another task. Its eleven-head early carrier slice, drawn from the top 128 by TL-score, reaches 80.0% block-readout accuracy with its own kernel, versus 5.2% with mean and 7.3% with shuffled weights; this extends the evidence for query-dependent retrieval to additional carriers beyond the selected eight. The donor comparisons and query-bootstrap uncertainty are reported together with the other settings in Tables 12 and 13.

Scope of the mechanism evidence. The core mechanism evidence replicates across seeds, model scales, and tasks. Weight, order, and vote controls, the intervention-predictor ladder, value-response decomposition, and component ablations cover Llama/TREC seeds 42 and 43, both Qwen scales, and Llama/dbpedia14. Table 9 extends the value-patch and prompt-edit R^2 comparisons to all five

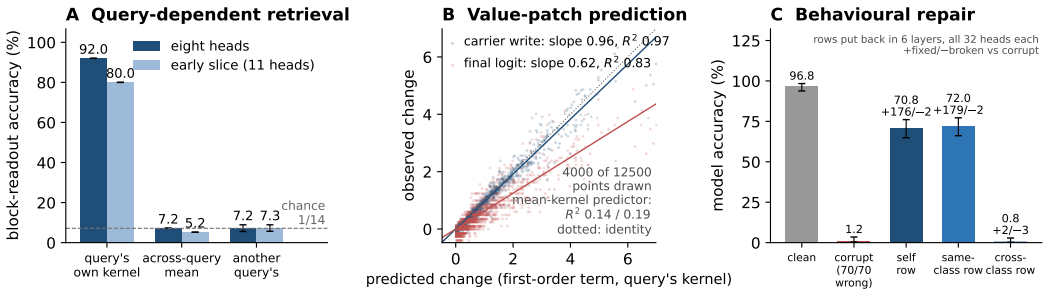


Figure 10: Mechanism checks on Llama-3.1-8B/dbpedia14. **(A)** Demonstration-block accuracy under own, mean, or shuffled query weights for the top-eight carriers and an early carrier slice of eleven heads drawn from the top 128 by TL-score; chance is $1/14$. **(B)** Predicted versus observed carrier and final-logit changes, showing 4,000 of 12,500 value-patch–query pairs. **(C)** Final-model accuracy after corrupting all seventy labels and repairing their values with self, same-class, or cross-class donors. Counts show corrected/broken predictions. The repair affects whole value rows in six layers, not only the eight carriers.

conditions. The broader ladder establishes replication of the predictor ordering and fit pattern, while detailed slope and clustered-uncertainty estimates are reported for their designated subsets.

Repair strength scans and donor comparisons cover four model–task settings, with Llama/TREC evaluated at seed 42. The reported repair intervals quantify query-level variation for the fixed corruption draws and prefixes used in each setting; robustness across fresh draws or prefixes remains a separate replication axis. Because tasks and corruption fractions differ across settings, repair magnitude is most directly interpreted within setting rather than as a cross-setting measure of mechanism strength. Static head analyses and the broader method matrix provide complementary coverage beyond these causal diagnostics.

B.8 Component Attribution and Ablation

The additive attribution below concerns Llama-3.1-8B/TREC-fine seed 42; the ablations extend to five diagnostic settings. For a fixed prompt, residual writes are additive:

$$h_{t_\star}^{(L)} = h_{t_\star}^{(0)} + \sum_{\ell=1}^{L} \left(\sum_{h} a_{t_\star}^{(\ell,h)} + m_{t_\star}^{(\ell)} \right). \tag{10}$$

Folding the learned final-normalization weights into the readout and freezing its scale at the observed prompt value permits an additive logit budget. Such an attribution is different from removing a component and allowing the model to respond. In this diagnostic, the top-eight heads contribute 125.8% of the net model margin; opposing writes elsewhere reduce the final margin below the selected heads' positive contribution. The percentage therefore measures their signed share of the observed margin, while the ablations below test the behavioral consequences of removing those components.

Carrier-layer MLPs provide a localized comparison group because they operate on residual states in the same layers as the selected heads. The joint ablations reveal strong non-additive interactions between these components. In the three Llama settings in Table 14, joint accuracy drops exceed the sums of the corresponding individual drops: 27.2 versus 3.6 percentage points for TREC seed 42, 39.6 versus 11.6 for seed 43, and 26.8 versus 16.4 for dbpedia14. This pattern is consistent with self-repair effects such as the Hydra effect (McGrath et al., 2023), where downstream components adapt after an ablation and thereby mask part of the removed component's behavioral contribution. Under this interpretation, the relatively small individual drops can reflect compensation by components that remain active, while joint ablation removes more of that compensatory capacity.

Qwen3-8B shows a smaller joint drop of 8.8 points, while Qwen3-4B exhibits a different interaction pattern: MLP-only ablation reduces accuracy by 14.4 points, compared with 8.4 points under joint ablation. The form of component interaction is therefore model-dependent, with the Llama results

Table 14: Individual and joint ablations ($n = 250$ queries per setting). Accuracy and drop are fractions; drop is clean minus ablated accuracy. Demo. gain lost divides this drop by clean minus diagnostic zero-shot accuracy, a behavioral normalization rather than a fraction of the TL mechanism. Carrier MLPs occupy the layers containing each setting's selected top-eight heads. In row-group order, layer counts are 6, 5, 5, 4, 6; clean (zero-shot) accuracies are 0.524 (0.004), 0.532 (0.004), 0.560 (0.004), 0.524 (0.020), and 0.968 (0.048).

Setting	Ablation	Accuracy	Acc. drop	Demo. gain lost
Llama/TREC, s42	Top-8 heads	0.4760	0.0480	9.2%
	Carrier MLPs	0.5360	−0.0120	−2.3%
	Both	0.2520	0.2720	52.3%
Llama/TREC, s43	Top-8 heads	0.5120	0.0200	3.8%
	Carrier MLPs	0.4360	0.0960	18.2%
	Both	0.1360	0.3960	75.0%
Qwen3-8B/TREC	Top-8 heads	0.5000	0.0600	10.8%
	Carrier MLPs	0.5760	−0.0160	−2.9%
	Both	0.4720	0.0880	15.8%
Qwen3-4B/TREC	Top-8 heads	0.5080	0.0160	3.2%
	Carrier MLPs	0.3800	0.1440	28.6%
	Both	0.4400	0.0840	16.7%
Llama/dbpedia14	Top-8 heads	0.8880	0.0800	8.7%
	Carrier MLPs	0.8840	0.0840	9.1%
	Both	0.7000	0.2680	29.1%

showing the clearest compensation-like signature. The additive residual budget complements these interventions by describing signed contributions in the intact forward pass.

Relation to self-repair. The superadditive accuracy drops in the Llama settings are consistent with compensatory computation of the kind studied by McGrath et al. (2023), where downstream components adjust after an ablation and partially offset the removed contribution. Rushing & Nanda (2024) further show that such self-repair is prompt-dependent and typically incomplete, with mechanisms including final-normalization rescaling and reduced erasure by downstream neurons. This perspective provides a natural explanation for why single-component ablations can substantially understate a component's role in the intact computation: surviving components can absorb part of the perturbation, whereas joint ablation removes more of that compensatory capacity.

Our measurements establish this interaction at the accuracy level rather than directly tracing the compensatory writes themselves, so we treat self-repair as a mechanistically plausible account of the Llama pattern. Qwen3-4B exhibits a different interaction regime, indicating that the form and strength of compensation are model-dependent. This endogenous adaptation is conceptually distinct from our externally imposed clean-value repair of corrupted labels.

C Static Alignment and Spectra of Carrier Writes

C.1 Cohort and Measured Writes

The three-seed cohort uses Llama-3.1-8B on TREC-fine with 36 candidate labels, $K = 5$, and prefix seeds 42, 43, and 44. Each seed uses 144 validation queries, with four reserved training examples per class. We capture all 1,024 heads' class-directed writes under intact demonstrations and five randomized-label controls, using bf16 weights and SDPA attention. Recomputing Equation 2 exactly recovers each seed's previously selected top eight. We additionally compare seed-42 plots for Llama-3.1-8B/dbpedia14 (56 queries, 14 classes) and Qwen3-8B and Qwen3-4B/TREC-fine (144 queries, 36 classes). Cross-seed AUROCs and rank summaries refer only to the original Llama-3.1-8B/TREC-fine cohort. These analyses characterize validation-selected outputs and are separate from the intervention diagnostics and held-out method evaluation.

For visualization and spectra, we remove each query's mean write across the C candidate classes:

$$\bar{s}_c^{(\ell,h)}(q) = s_c^{(\ell,h)}(q; D) - \frac{1}{C} \sum_{c' \in Y} s_{c'}^{(\ell,h)}(q; D). \tag{11}$$

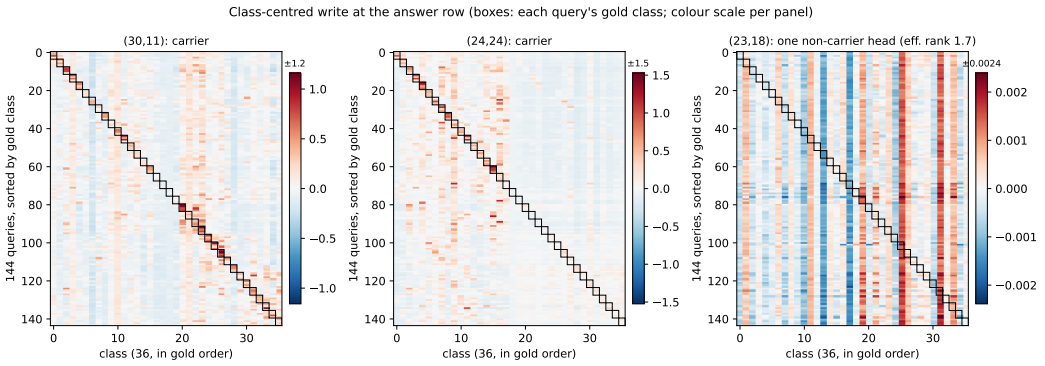


Figure 11: Class-directed writes on 144 validation queries: two selected carriers and an illustrative comparison head outside the selected top eight (Llama-3.1-8B, TREC-fine, $K = 5$, seed 42). Rows are sorted by correct class; columns are 36 candidate labels, with correct-class boxes. Each row is centered across classes. Color scales differ: the comparison head's writes are much smaller. These are head writes, not final logits.

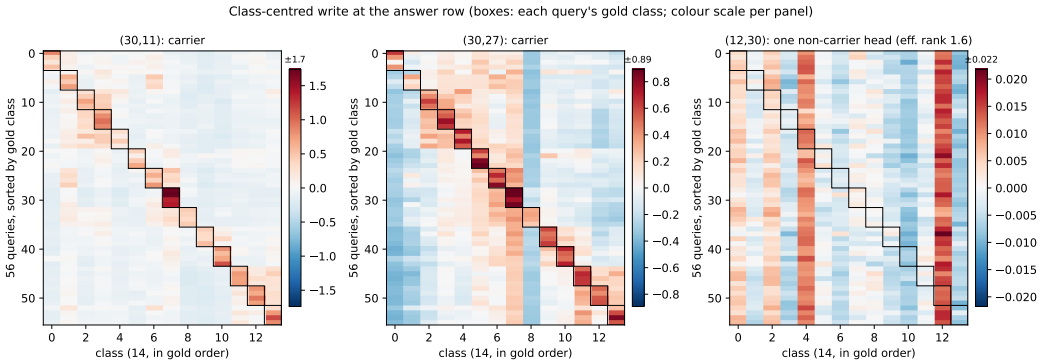


Figure 12: Class-directed writes for Llama-3.1-8B on dbpedia14, using the visualization conventions of Figure 11. Rows are sorted by correct class and centered across candidate labels; each panel has its own color scale.

Figures 11–14 display these values under intact demonstrations. Class centering removes a common offset within each row and preserves class-score differences; it does not subtract the mean across queries. Thus, class-specific preferences shared across queries remain visible. The method itself continues to use the raw readout in Equation 4.

Across these heatmaps, the selected carriers' writes track the query's correct class, while the illustrative comparison heads outside the selected set mainly exhibit class preferences shared across queries. The alignment and spectral comparisons below examine how consistently this functional distinction corresponds to weight geometry and the breadth of observed writes.

C.2 Static Alignment with Class Directions

Let $U_Y = [W_U^{c_1}, \ldots, W_U^{c_C}]$ collect the candidate-label directions, and let

$$H_C = I_C - \frac{1}{C}\mathbf{1}\mathbf{1}^\top, \qquad \widetilde{W}_U = H_C U_Y^\top. \tag{12}$$

The rows of $\widetilde{W}_U$ span a class-contrast subspace of dimension at most $C - 1$ (35 for TREC-fine and 13 for dbpedia14). With Q_Y an orthonormal basis for this subspace, define the static class-energy

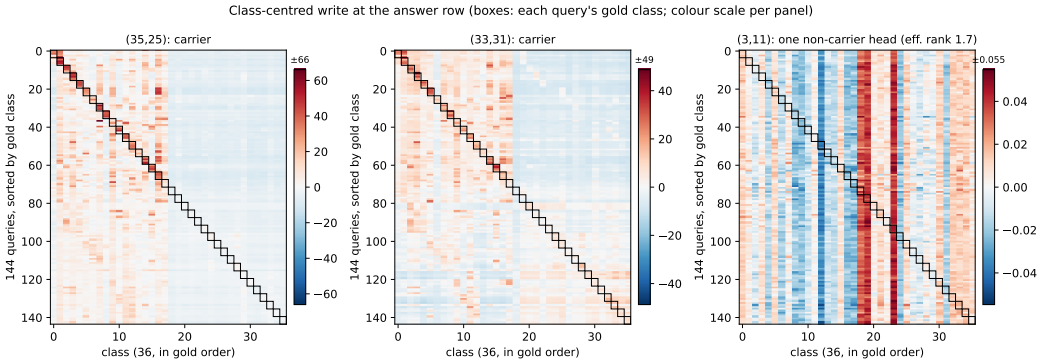


Figure 13: Class-directed writes for Qwen3-8B on TREC-fine, using the visualization conventions of Figure 11. Rows are sorted by correct class and centered across candidate labels; each panel has its own color scale.

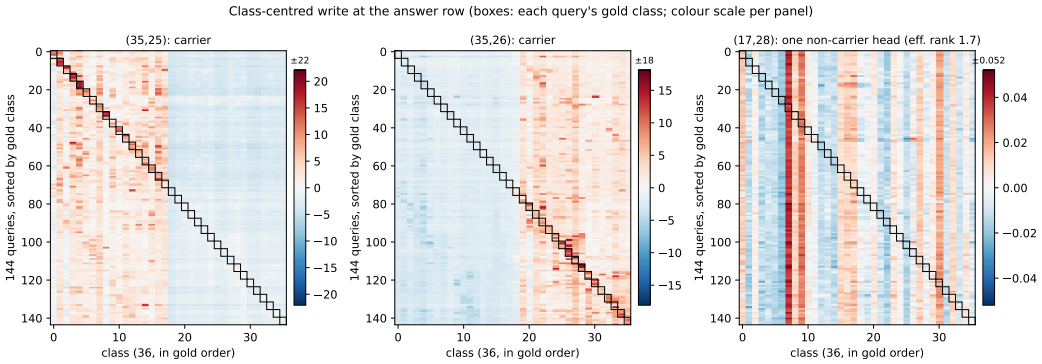


Figure 14: Class-directed writes for Qwen3-4B on TREC-fine, using the visualization conventions of Figure 11. Rows are sorted by correct class and centered across candidate labels; each panel has its own color scale.

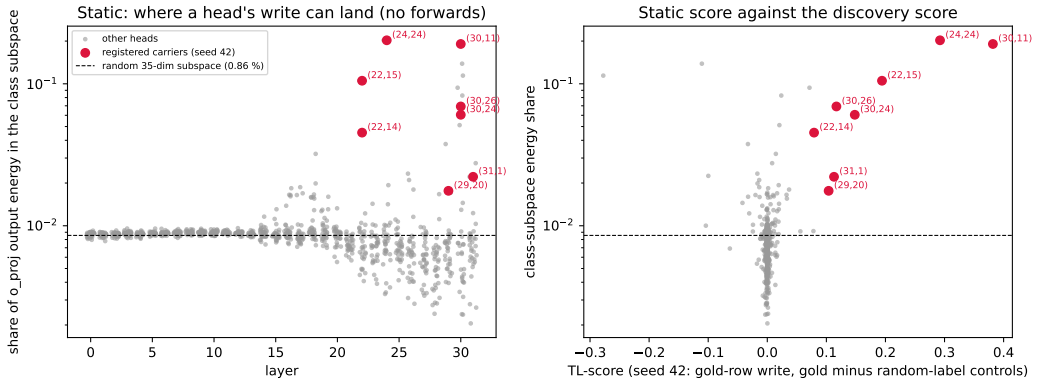


Figure 15: Static class-subspace alignment of head outputs. Each point is one of 1,024 heads in Llama-3.1-8B; red points mark the eight method carriers selected for TREC-fine at $K = 5$, seed 42. Left: the class-energy fraction in Equation 13 against layer. Right: the same fraction against TL-score. The vertical axis is logarithmic, and the dashed line gives the random 35-dimensional subspace reference. High static alignment occurs in some heads with little mapping-sensitive correct-class write, motivating the functional selection criterion.

fraction of a head's output projection as

$$E_{\ell,h} = \frac{\|Q_Y^\top W_O^{(\ell,h)}\|_F^2}{\|W_O^{(\ell,h)}\|_F^2}. \tag{13}$$

This quantity requires only the model weights and candidate-label directions. It measures the orientation of the output map, not the fraction of activation energy observed during a forward pass.

For Llama-3.1-8B/TREC-fine, Figure 15 shows that the selected carriers are strongly enriched for static alignment with the class-contrast subspace. At seed 42, their median class-energy fraction is 6.49%, compared with 0.87% across all heads and approximately 0.86% for a random 35-dimensional subspace. Static class energy separates the eight selected carriers from the remaining heads with AUROC 0.995–0.997 across the three prefix seeds, and all eight carriers appear within the static top thirty for every seed. Restricting the seed-42 comparison to layers 22–31 retains an AUROC of 0.986, showing that the enrichment persists within the later-layer regime.

Static alignment and TL sensitivity nevertheless capture distinct properties. Their Spearman correlation is only 0.087–0.122, and the static top thirty overlap with only fourteen to sixteen of the TL-score top thirty. The additional model–task settings show the same qualitative separation: selected carriers lie above their random-subspace references, while some strongly aligned non-carriers have near-zero or negative TL-scores. Static class energy therefore identifies heads whose output maps are well positioned to write class contrasts, whereas TL-score further selects heads whose class-directed writes respond favorably to the demonstrated mapping.

C.3 Mapping Spectra and Observed Write Spectra

For a matrix A with nonzero singular-value energy, we summarize its spectrum using

$$p_j = \frac{\sigma_j(A)^2}{\sum_k \sigma_k(A)^2}, \qquad r_{\text{eff}}(A) = \exp\left(-\sum_j p_j \log p_j\right), \tag{14}$$

with $0 \log 0 = 0$. The spectral curves show the cumulative energy $\sum_{j=1}^{m} p_j$ in the first m singular directions. Effective rank summarizes how broadly this energy is distributed across directions: it equals the number of directions under a uniform spectrum and decreases as the spectrum becomes more concentrated.

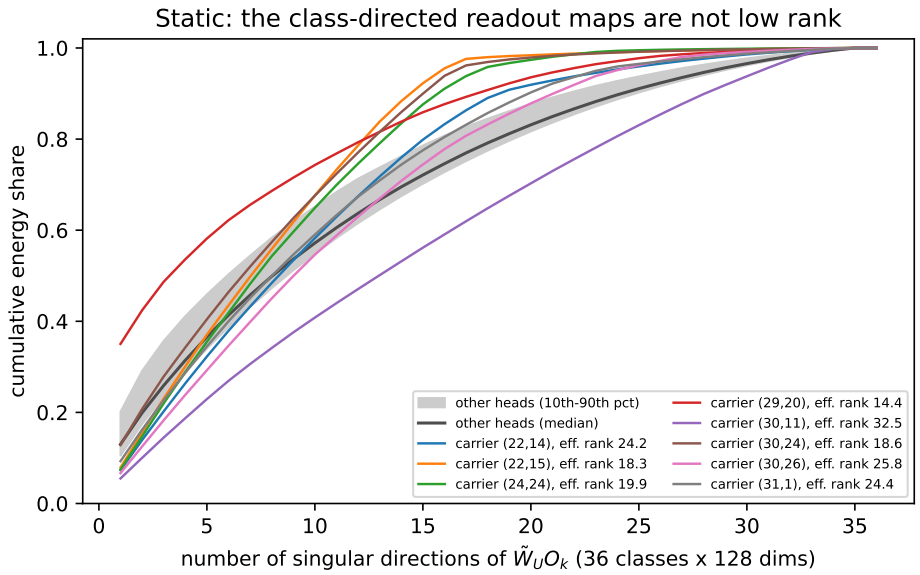


Figure 16: Static class-directed map spectra for Llama-3.1-8B/TREC-fine, seed 42. Curves show cumulative squared singular-value energy of $\widetilde{W}_U W_O^{(\ell,h)}$ for the eight carriers. The gray line and band show the median and 10th–90th percentiles over other heads. Carrier spectra vary and largely overlap this reference in this setting.

Static alignment does not imply concentration in a few modes. We first decompose the class-directed output map

$$A^{(\ell,h)} = \widetilde{W}_U W_O^{(\ell,h)} \in \mathbb{R}^{C \times d_{\text{head}}}. \tag{15}$$

This map sends a head's pre-output-projection vector to class-centered scores. For Llama-3.1-8B/TREC-fine, the selected carrier spectra largely overlap those of the remaining heads (Figure 16); at seed 42, their effective ranks range from 14.4 to 32.5, with a median of 22.0 compared with 26.4 across all heads. The corresponding carrier ranges are 10.9–12.9 for Llama-3.1-8B/dbpedia14, 18.5–29.1 for Qwen3-8B, and 19.2–28.7 for Qwen3-4B (Figures 18–20, middle). These spectra are generally multi-mode rather than concentrated in one or two dominant directions, although several Qwen carriers are more concentrated than the typical head. Static class alignment therefore reflects how strongly a head can access the class-contrast subspace, while its spectrum describes how broadly that access is distributed within the subspace. Effective ranks are interpreted relative to the class-centering ceilings of 13 for dbpedia14 and 35 for TREC-fine.

Observed write spectra vary across models and tasks. We form a query-by-class matrix $X^{(\ell,h)}$ from the intact-prefix writes $\bar{s}^{(\ell,h)}(q)$. On Llama-3.1-8B/TREC-fine, these matrices have size 144×36 (Figure 17). Across seeds 42, 43, and 44, carrier median effective ranks are 10.6, 10.7, and 12.2, compared with 1.71, 1.69, and 1.96 across all heads. Their median first-mode energy shares are 0.347, 0.289, and 0.257, versus 0.903, 0.906, and 0.872. Thus, carriers in this cohort typically distribute their observed class-directed writes across substantially more modes, although individual carriers such as seed-42 heads $(30, 24)$ and $(29, 20)$ remain concentrated, with effective ranks 2.0 and 3.3.

The additional settings reveal substantial variation in this spectral breadth (Figures 18–20, bottom). Carrier write ranks range from 2.3 to 10.0 on Llama-3.1-8B/dbpedia14, 3.7 to 11.6 on Qwen3-8B, and 2.9 to 9.6 on Qwen3-4B; five of the eight Qwen3-4B carriers lie between 2.9 and 3.3. Static capacity and observed use can also differ sharply: Qwen3-4B head $(33, 30)$ has static effective rank 21.4 but observed write rank 2.9. These results show that carriers can implement query-sensitive class writing with either narrow or broad observed spectra, depending on model, task, and head.

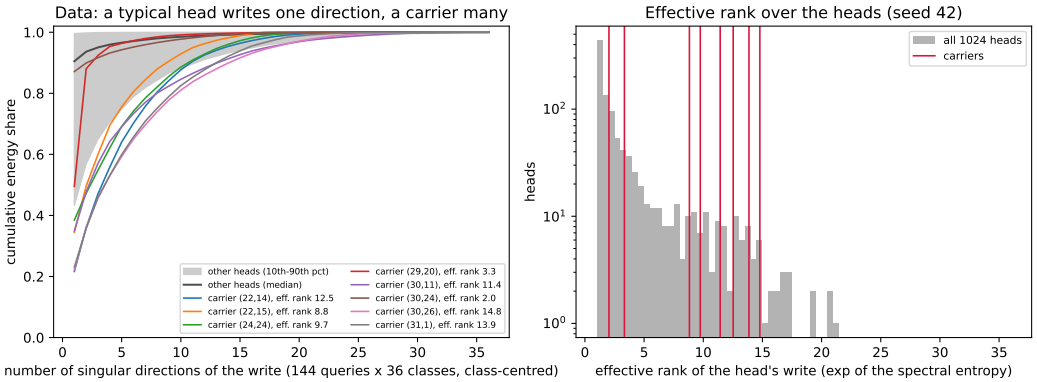


Figure 17: Observed write spectra for Llama-3.1-8B/TREC-fine, seed 42. Left: cumulative squared singular-value energy of each carrier's 144×36 class-centered write matrix, with the median and 10th–90th percentiles over other heads. Right: effective ranks over all 1,024 heads, with carriers marked. Most carriers have broader spectra in this setting, with two low-rank exceptions. No mean is removed across queries.

Because the query-by-class matrices retain shared class preferences, spectral breadth captures the diversity of observed write directions rather than query dependence itself. A fixed direction can still vary strongly in magnitude across queries, while a broad spectrum need not align with the correct class. We therefore use TL-score and query-weight controls to identify mapping sensitivity and query-dependent retrieval, with the write spectra characterizing how those responses are distributed across class directions.

C.4 Corresponding Carrier Locations Across Qwen3 Scales

In the seed-42 TREC-fine comparison, Qwen3-4B and Qwen3-8B select exactly the same eight layer–head coordinates:

$$\mathcal{T}_{4B} = \mathcal{T}_{8B} = \{(29,5),(29,7),(33,30),(33,31),\\ (34,28),(34,30),(35,25),(35,26)\}. \tag{16}$$

The two scales therefore localize the tested TL carrier role to the same head positions, despite substantial differences in write magnitude and spectral breadth (Figures 19 and 20). The correspondence is positional: the selected heads occupy the same coordinates while retaining model-specific response magnitudes and spectral profiles.

This cross-scale alignment motivates the hypothesis that related training histories can preserve the location of TL-relevant computation across model sizes. Distillation or other shared-lineage training procedures provide natural settings in which to test this possibility. We leave the emergence conditions of this cross-scale correspondence as an open question for future work on pretraining dynamics and model lineage.

D Additional Access and Readout Controls

D.1 Why Retain Base Demonstrations?

Table 15 tests how base demonstrations condition the receiver's use of selective memory, using both direct and final-model readouts on Llama-3.1-8B with TREC-fine and dbpedia14, and Qwen3-8B and Qwen3-4B with TREC-fine. Comparing $0 \to 5$ with $2 \to 5$ fixes the full memory at $K_{\text{full}} = 5$ while changing base access; $5 \to 10$ provides a larger-budget reference.

At $0 \to 5$, thirty-head selective access with model readout reaches 1.60–10.53% across the four settings, indicating that carrier access alone provides only a weak substitute for the demonstration-conditioned receiver. Adding two base examples per class sharply improves model readout at both

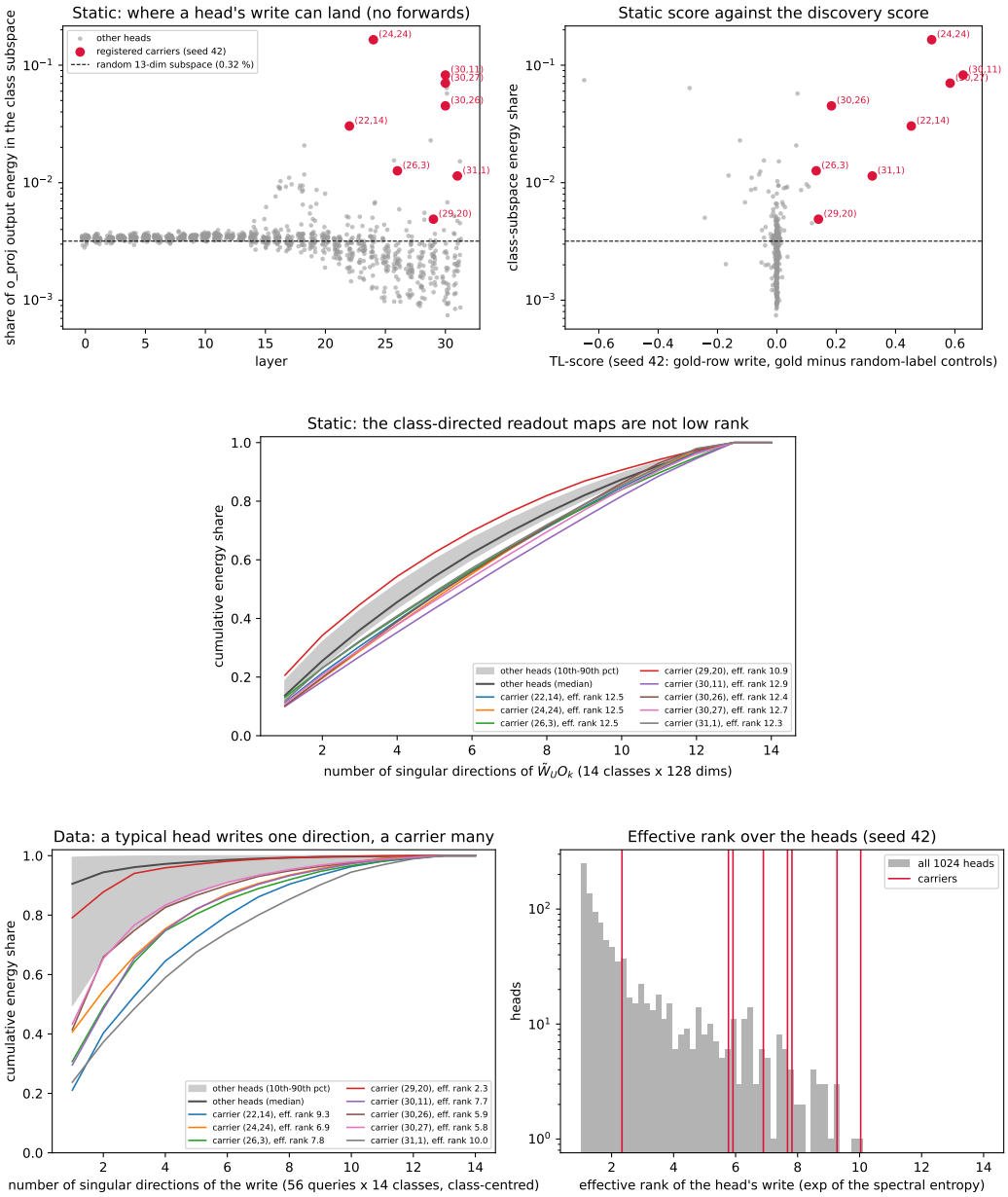


Figure 18: Alignment and spectra for Llama-3.1-8B/dbpedia14, seed 42. Top: static class energy against layer and TL-score. Middle: static class-map spectra. Bottom: observed write spectra on 56 queries over 14 classes and the distribution of effective ranks. The class-centering rank ceiling is 13. Colors and references follow Figures 15, 16, and 17.

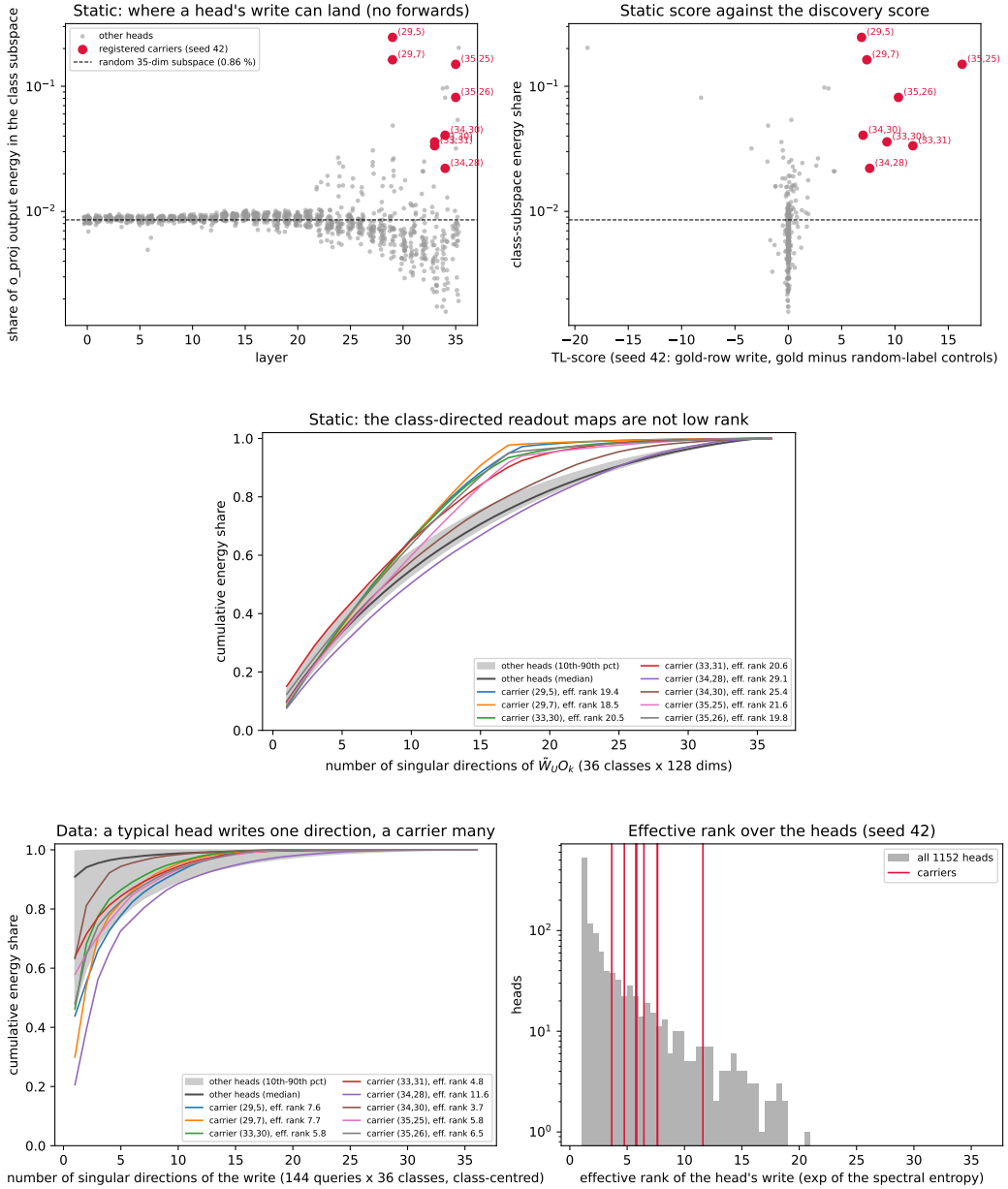


Figure 19: Alignment and spectra for Qwen3-8B/TREC-fine, seed 42, with the panel order of Figure 18. Observed writes use 144 queries and 36 classes. Some strongly aligned heads outside the selected set have negative TL-scores, and high effective rank does not uniquely identify the selected heads.

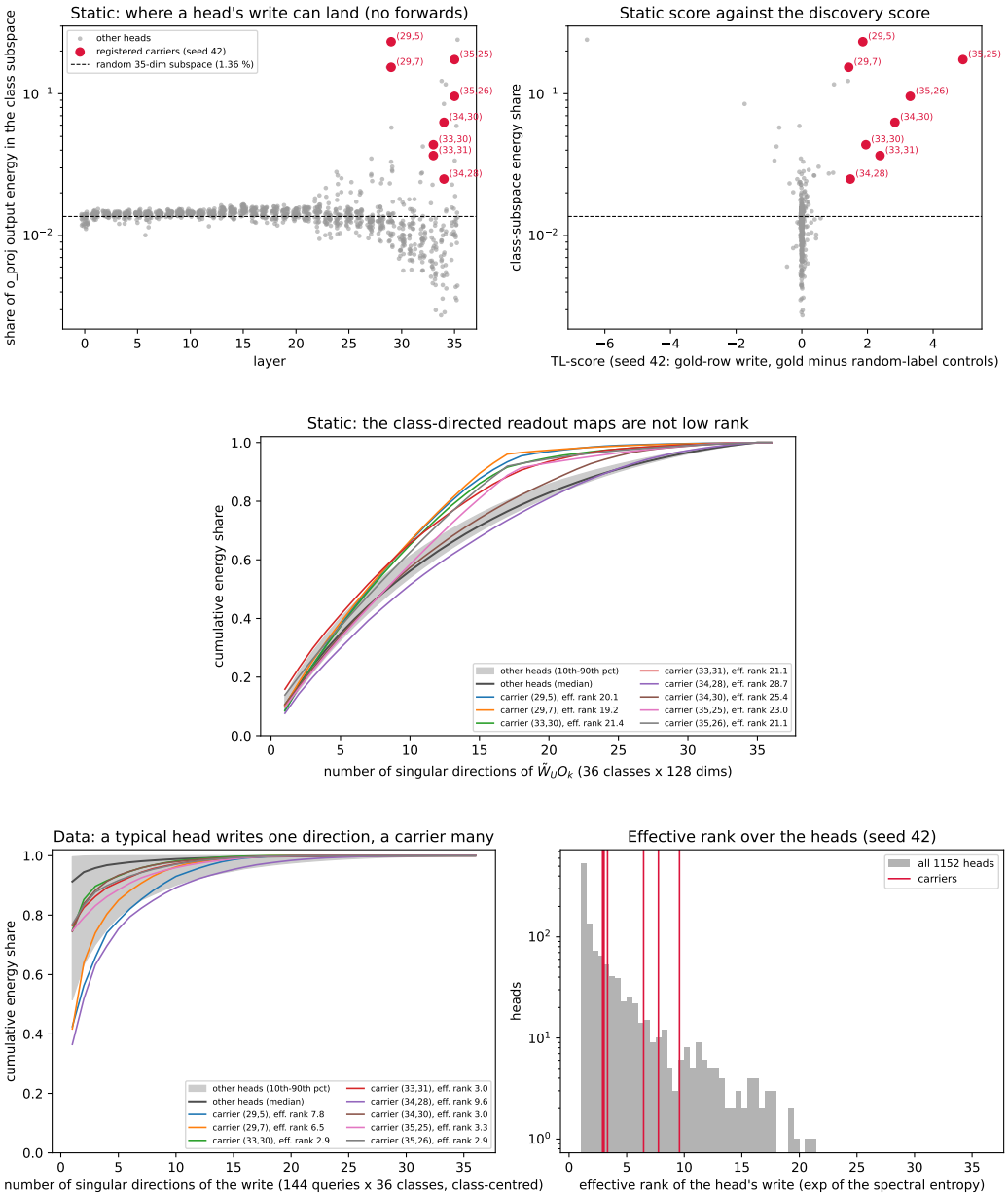


Figure 20: Alignment and spectra for Qwen3-4B/TREC-fine, seed 42, with the panel order of Figure 18. Observed writes use 144 queries and 36 classes. Five carriers have write effective ranks of 2.9–3.3 despite broad static class maps. The selected head coordinates match Qwen3-8B, while their write spectra differ.

Configuration	$0 \to 5$	$2 \to 5$	$5 \to 10$
Llama-3.1-8B / TREC-fine			
Natural	0.40	36.93	49.33
All heads	51.60	51.60	58.27
Ours (model), eight heads	0.13	38.40	50.00
Ours (model), thirty heads	3.73	46.67	50.40
Ours (direct), eight heads	10.13	44.40	51.60
Ours (direct), thirty heads	11.07	47.47	53.07
TSLA, TL heads	0.67	32.67	39.20
TSLA, TR heads	1.33	12.40	19.07
Llama-3.1-8B / dbpedia14			
Natural	5.73	71.20	90.93
All heads	93.33	93.33	95.33
Ours (model), eight heads	7.87	76.13	93.07
Ours (model), thirty heads	10.53	91.33	93.47
Ours (direct), eight heads	8.93	83.20	91.47
Ours (direct), thirty heads	8.40	88.93	93.47
TSLA, TL heads	3.87	63.87	84.13
TSLA, TR heads	6.40	49.60	71.20
Qwen3-8B / TREC-fine			
Natural	0.67	37.33	33.60
All heads	57.60	57.60	62.53
Ours (model), eight heads	0.53	38.13	34.40
Ours (model), thirty heads	1.60	44.80	40.00
Ours (direct), eight heads	10.67	40.80	37.60
Ours (direct), thirty heads	10.40	43.47	42.27
TSLA, TL heads	0.80	5.20	5.60
TSLA, TR heads	1.87	5.07	4.40
Qwen3-4B / TREC-fine			
Natural	1.07	33.60	34.93
All heads	42.27	42.27	54.13
Ours (model), eight heads	1.47	35.73	31.87
Ours (model), thirty heads	1.60	44.27	33.33
Ours (direct), eight heads	11.60	41.73	40.40
Ours (direct), thirty heads	9.33	46.67	41.33
TSLA, TL heads	2.00	4.53	16.27
TSLA, TR heads	8.53	12.13	23.87

Table 15: Dependence on base demonstrations: test accuracy (%) over three seeds and 250 queries per seed. Natural hides the increment at every head; All heads uses full access. Both Ours variants grant selective access: model readout continues to final logits, while direct readout uses raw class scores from the selected heads. All other rows use final-model predictions. TSLA uses full-vector addition at $\alpha = 1$. Uniform guessing gives 2.78% over 36 TREC-fine candidates and 7.14% over 14 dbpedia14 candidates. The nonzero-base columns reproduce the corresponding main-evaluation controls.

carrier budgets in every setting: thirty-head accuracy rises from 10.53% to 91.33% on Llama-3.1-8B/dbpedia14 and from 1.60% to 44.80% on Qwen3-8B/TREC-fine.

Direct readout shows the same dependence on base conditioning while revealing that some class information is already present at zero base. At $0 \to 5$, TREC-fine direct accuracies range from 9.33% to 11.60%, above the 2.78% uniform reference, whereas dbpedia14 remains near its 7.14% reference. With two base examples per class, thirty-head direct accuracy rises from 11.07% to 47.47% on Llama/TREC-fine and from 8.40% to 88.93% on Llama/dbpedia14. These results suggest that the carrier pathway is most effective relative to a demonstration-conditioned receiver: limited class information is readable without base examples, but base conditioning strongly amplifies both the model's use of the increments and their directly readable class structure. This motivates expansion from a demonstration-conditioned receiver, consistent with the demonstration demand in Figure 1.

TSLA's full-vector steering also remains far below full access at $0 \to 5$, although the TR vector raises Qwen3-4B accuracy from 1.07% to 8.53%. Zero-shot injection can therefore help without

recovering the performance obtained from full demonstrations. Base context changes the state being steered, motivating the base-subtraction in Appendix A.2.

The zero-base comparisons reveal that predictive information can be present in the selected writes before it is fully expressed in final-model decisions. Direct readout exceeds the corresponding model readout in all six TREC-fine setting–budget comparisons. On Llama-3.1-8B/TREC-fine, eight-head model readout at $0 \rightarrow 5$ also improves candidate negative log-likelihood by 0.349 nats even when final-model accuracy does not exceed Natural. Together, these results show a subthreshold memory response: selected writes can carry and improve task-relevant evidence that downstream processing has not yet converted into an argmax change.

D.2 Head-Selection Controls

The head-selection comparison evaluates registered eight-head carriers and the top thirty heads by the paired TL-score alongside equally sized TSLA-TL, attention-share, and random sets. The prediction rule reads the selected heads' outputs directly. For the score-based sets, Table 16 reports no-access and selective-access accuracies; the proposed sets additionally have full-access references. These reference values come from the separate access-control evaluation in Table 20, and all head-selection comparisons use the values in this block.

Readout accuracy and gains from memory access. The proposed sets have higher no-access direct accuracy than both alternative scoring rules in all eight setting–budget comparisons, indicating that their outputs are already more class-readable before selective memory access. Selective access improves all three score-based readouts at both budgets in every setting, but the size of this gain need not track final accuracy. On Qwen3-8B/TREC-fine with eight heads, our readout rises from 34.53% to 37.60% (+3.07 points), while attention-share selection rises from 17.73% to 24.13% (+6.40 points); the proposed set therefore retains the stronger classifier despite the smaller incremental gain.

The same distinction appears on dbpedia14. TSLA-TL gains 23.20 and 26.00 percentage points with eight and thirty heads, compared with 14.13 and 18.53 points for the proposed sets, yet the latter reach higher selective-access accuracies of 83.20% and 88.93%. Random-set direct accuracies range from 5.8% to 19.0%, and the proposed sets exceed their same-budget random references in all eight comparisons. Across these settings, paired TL-score selection therefore identifies outputs that are both intrinsically more class-readable and able to benefit from selective memory access.

Overlap between selection criteria. TSLA-TL's reported Jaccard overlap with the proposed sets ranges from 0.29 to 0.68, compared with 0.00–0.21 for attention-share selection (Table 17). The criteria thus select partially overlapping reader sets. The largest TSLA-TL overlap occurs for thirty heads on dbpedia14 (0.68); the proposed set has a 4.13-point advantage in selective direct-output accuracy (88.93% versus 84.80%). The accuracy comparisons provide the functional assessment of these alternative selections.

D.3 Comparison with an Explicit Nearest-Neighbor Classifier

The direct readout in Equation 4 is a kernel classifier whose weights and votes are produced by the model itself: attention weights compare a live query with contextual demonstration states, and value–output projections supply the class votes. A natural question is whether this learned retrieval provides more than an explicit nearest-neighbor rule applied to the same model's representations of the same demonstrations. Table 18 compares the thirty-head direct readout with a k-nearest-neighbor (kNN) classifier built from hidden states of the same frozen model.

Protocol. The kNN classifier uses the same K_{full} demonstrations per class, prefix seeds, and 250 test queries per seed as the corresponding rows of Table 2. Each demonstration input and each test query is represented by a residual-stream state from one layer, either mean-pooled over the example's tokens or taken at its final token, and the prediction is a majority vote among the k nearest demonstrations. The number of neighbors, the pooling rule, and the layer are treated as hyperparameters and selected on the same validation queries used for head selection, separately for each model–task–budget setting, then frozen before test evaluation. The explicit classifier therefore

Setting	Selection	Heads	No	Selective	Full
Llama/TREC, $2 \rightarrow 5$	Ours	8	35.07	44.40	50.27
	Ours	30	36.93	47.47	51.60
	TSLA-TL	8	19.33	21.73	—
	TSLA-TL	30	19.20	22.67	—
	Attention share	8	14.13	22.00	—
	Attention share	30	26.80	38.00	—
Qwen3-4B/TREC, $2 \rightarrow 5$	Ours	8	33.47	41.73	44.27
	Ours	30	37.07	46.67	47.20
	TSLA-TL	8	13.47	18.93	—
	TSLA-TL	30	15.07	20.80	—
	Attention share	8	13.73	16.27	—
	Attention share	30	16.27	23.07	—
Qwen3-8B/TREC, $5 \rightarrow 10$	Ours	8	34.53	37.60	63.87
	Ours	30	33.07	42.27	65.33
	TSLA-TL	8	19.60	22.93	—
	TSLA-TL	30	15.60	21.87	—
	Attention share	8	17.73	24.13	—
	Attention share	30	15.60	23.60	—
Llama/dbpedia14, $2 \rightarrow 5$	Ours	8	69.07	83.20	92.00
	Ours	30	70.40	88.93	92.80
	TSLA-TL	8	57.20	80.40	—
	TSLA-TL	30	58.80	84.80	—
	Attention share	8	45.07	68.00	—
	Attention share	30	55.87	85.60	—

Table 16: Direct-output accuracy (%) before and after selective increment access for the three score-based head-selection criteria. Within a row, the selected head set and direct prediction rule are fixed. No hides the increment from every head; Selective exposes it only to the selected heads; base access remains available. Full exposes it to all heads while retaining the same selected-head readout. Full values are available for the Ours reference sets; dashes indicate unreported measurements. These references reproduce the corresponding entries in Table 20. Random-set selective readouts are reported in Table 1.

	TSLA-TL vs. Ours		Attention share vs. Ours	
Setting	8 heads	30 heads	8 heads	30 heads
Llama/TREC, $2 \rightarrow 5$	0.39	0.38	0.09	0.20
Qwen3-4B/TREC, $2 \rightarrow 5$	0.37	0.29	0.00	0.10
Qwen3-8B/TREC, $5 \rightarrow 10$	0.30	0.33	0.00	0.13
Llama/dbpedia14, $2 \rightarrow 5$	0.43	0.68	0.09	0.21

Table 17: Reported Jaccard overlap between each alternative ranking and the corresponding proposed head set at the same budget. Ours uses the TL-score top eight head set or the top thirty. Jaccard is $|A \cap B|/|A \cup B|$ for head sets A and B; values describe selection overlap, with direct-output performance reported in Table 1.

sees every demonstration and chooses its representation by validation, matching the demonstration bank available to the full-access readout.

The model's kernel outperforms tuned neighbor voting. With full access, the thirty-head carrier readout exceeds the tuned kNN classifier in all six settings, by 2.0 to 15.7 percentage points. On Llama-3.1-8B/TREC-fine it reaches 51.60% and 59.07% against 43.60% and 51.07% for kNN; the margin is largest on Qwen3-8B/TREC-fine at $5 \rightarrow 10$ (65.33% versus 49.60%) and smallest on dbpedia14, where both classifiers exceed 90%. The readout is therefore not merely recovering a neighbor structure already present in pooled hidden states: the model's query-dependent weights over contextual demonstration states, combined with contextual class votes, form a more accurate classifier than an explicit neighbor rule given the same examples and a validation-tuned representation.

Selective access retains the advantage on Llama-3.1-8B but not on Qwen3-8B. Under selective access, the same readout exceeds kNN in three of the four Llama-3.1-8B settings, by 3.87 and 2.00

				Carrier readout, 30 heads	
Model	Task	Level	kNN	Ours (direct)	Full access (direct)
Llama-3.1-8B	TREC-fine	2→5	43.60	47.47	**51.60**
Llama-3.1-8B	TREC-fine	5→10	51.07	53.07	**59.07**
Llama-3.1-8B	dbpedia14	2→5	90.80	88.93	**92.80**
Llama-3.1-8B	dbpedia14	5→10	92.93	93.47	**95.33**
Qwen3-8B	TREC-fine	2→5	50.80	43.47	**58.40**
Qwen3-8B	TREC-fine	5→10	49.60	42.27	**65.33**

Table 18: Thirty-head carrier readout versus a k-nearest-neighbor classifier over the same model's hidden states: test accuracy (%) over three seeds and 250 queries per seed. The kNN classifier uses all K_{full} demonstrations per class; its number of neighbors, pooling rule (mean over tokens or final token), and layer are selected on the validation queries of each setting. Ours (direct) and Full access (direct) reproduce the thirty-head columns of Table 2. Bold marks the highest value in each row.

points on TREC-fine and by 0.54 points on dbpedia14 at $5 \rightarrow 10$, and falls 1.87 points short on dbpedia14 at $2 \rightarrow 5$. On Qwen3-8B/TREC-fine it is 7.33 points below kNN at both levels. This shortfall reflects the access restriction rather than the readout: the identical thirty-head classifier exceeds kNN by 7.60 and 15.73 points once the increment is visible to all heads, and the factorial in Table 20 shows that on Qwen3-8B most of the full-access gain arrives through other heads' access rather than through the carriers' own view of the increment. The comparison thus separates two statements: the kernel the model implements, read from thirty heads with the demonstrations fully visible, is a stronger classifier than tuned kNN over the model's own representations in every tested setting, whereas how much of that advantage survives selective memory access depends on the model and on the states that the surrounding network constructs for the readers.

D.4 Task Dependence and Carrier Budget

Dataset-specific gains and exceptions. Additional access produces gains beyond the TREC-fine cells. On Llama-3.1-8B/dbpedia14 at $2 \rightarrow 5$, All heads reaches 93.33%, compared with Natural at 71.20%. This comparison uses full-prefix representations and measures expanded visibility, while Figure 1 varies the demonstration budget directly.

The broader benchmark also reveals task-dependent intervention preferences. On yelp, thirty-head direct readout reaches 24.93%, while ICV and I2CL reach 39.33% and 37.60%, respectively; both vector methods also exceed All heads at 28.40%. On dbpedia14 at $5 \rightarrow 10$, I2CL reaches 93.87%, slightly above the method's 93.47%. On yahoo, direct readout gives 39.60% at $2 \rightarrow 5$ and 55.33% at $5 \rightarrow 10$, compared with Natural at 38.27% and 57.73%. These variations indicate that selective readout and vector interventions exploit different task regimes, motivating further analysis of the conditions under which each is most effective. Among vector baselines, TV most often exceeds Natural, in 9 of the 18 natural-language settings. TSLA's full-vector and base-subtracted results are examined separately in Appendix A.2.

Task-dependent gains from additional readers. Thirty-head direct readout exceeds eight-head readout in all 18 natural-language settings, with gains of 13.87 points on banking77 and 17.47 on clinc150, compared with 2.00 points on dbpedia14 at $5 \rightarrow 10$. These comparisons characterize useful reader budgets under the current ranking and raw aggregation rule. Supplementary rule-learning results, including cases favoring the smaller budget, are reported in Appendix E.2.

Budget effects arise through both the starting readout and the gain from memory access. On banking77, the 13.87-point selective-readout advantage of thirty heads is accompanied by a 10.80-point no-access difference, while the corresponding increment gains are 4.67 and 7.74 points. Expanding the reader set can therefore strengthen both baseline class readability and the use of additional memory, suggesting that the preferred budget depends on how task information is distributed across heads.

Table 19: Ours (direct) at larger carrier budgets: test accuracy (%) over three seeds and 250 queries per seed. Each budget takes the top-ranked heads of the same validation ranking, which serve as both the memory readers and the direct readout under selective increment access. Natural and All heads give final-model predictions with the increment hidden from or visible to every head. Natural, All heads, and the eight- and thirty-head values reproduce Table 2. TREC denotes TREC-fine; Llama denotes Llama-3.1-8B.

			Ours (direct)			
Setting	Natural	All heads	8	30	64	128
Llama/TREC, $2 \to 5$	36.93	51.60	44.40	47.47	48.93	49.47
Qwen3-8B/TREC, $2 \to 5$	37.33	57.60	40.80	43.47	46.67	46.93
Llama/dbpedia14, $2 \to 5$	71.20	93.33	83.20	88.93	91.07	91.47

D.5 Larger Carrier Budgets under Selective Access

Table 19 shows a trade-off in the head budget: more heads can yield higher accuracy, but with diminishing marginal returns. From 30 to 64 heads, accuracy increases by 1.46–3.20 points across the three settings, whereas doubling the budget to 128 heads adds only 0.26–0.54 points.

D.6 Holding the Carrier Readout Fixed

Table 2 provides a no-increment direct-readout control for every model, task, level, and head budget. No access retains the base demonstrations while withholding direct visibility of the increment positions. Comparing Ours (direct) with No access (direct) therefore holds the head set, raw projection, cached base representations, query draw, and receiver positions fixed while changing selective increment visibility. Subsequent states are recomputed normally, so this comparison measures the net effect of the access intervention on the fixed carrier readout.

Accuracy improves in all 36 head-budget comparisons across the 18 natural-language settings. Full-access direct readout exceeds both no access and selective access in every comparison. Thus, increment information is useful to the carrier readout in these settings, while access through the surrounding network can further improve the states on which that readout operates.

Access gains differ from advantages over Natural. On Llama-3.1/TREC-fine at $5 \to 10$, thirty-head direct readout reaches 53.07%, 3.74 points above Natural at 49.33%, but only 0.40 points above its no-access reference at 52.67%. Conversely, on Llama-2 with SelfExtend and on yahoo at $5 \to 10$, thirty-head direct readout remains below Natural while improving over no access by 1.47 and 3.73 points. These comparisons separate the quality of the standalone readout from the incremental value contributed by newly visible memory.

Full access can further improve the carrier readout. On Llama-3.1/TREC-fine at $2 \to 5$, the eight-head readout progresses from 35.07% with no increment access to 44.40% with selective access and 50.27% with full access. This progression is consistent with the additional access routes examined in Appendix D.7: other heads can improve the contextual states on which the selected readout relies. These controls characterize online visibility within a shared full-prefix computation, complementing experiments that vary the demonstration budget itself.

D.7 Carrier and Other-Head Access

Table 20 adds the missing complementary-access condition to a separate four-condition evaluation. Within each row, the same carrier set is read through the raw, uncentered class argmax. The increment is visible to no heads (No), only heads outside the selected carrier set $\mathcal{T}$ (X), only heads in $\mathcal{T}$ (Ours), or all heads (Full); base access remains available throughout. The complementary group can include additional mapping-sensitive carriers outside the selected budget. These are visibility interventions, with the remaining computation active. All effects below use the matched values in this table, which are reported separately from Table 2.

Model	Task	Level	Heads	No	X	Ours	Full	Carrier access	Other-head access
Llama-3.1-8B	TREC-fine	$2\to5$	8	35.07	43.07	44.40	50.27	$+8.3$	$+6.9$
			30	36.93	39.20	47.47	51.60	$+11.5$	$+3.2$
Qwen3-4B	TREC-fine	$2\to5$	8	33.47	41.87	41.73	44.27	$+5.3$	$+5.5$
			30	37.07	35.87	46.67	47.20	$+10.5$	-0.3
Llama-3.1-8B	dbpedia14	$2\to5$	8	69.07	86.67	83.20	92.00	$+9.7$	$+13.2$
			30	70.40	72.27	88.93	92.80	$+19.5$	$+2.9$
Qwen3-8B	TREC-fine	$5\to10$	8	34.53	60.53	37.60	63.87	$+3.2$	$+26.1$
			30	33.07	54.27	42.27	65.33	$+10.1$	$+22.1$
Qwen3-4B	TREC-fine	$5\to10$	8	34.80	50.80	40.40	54.67	$+4.7$	$+15.1$
			30	37.73	49.07	41.33	56.67	$+5.6$	$+13.3$

Table 20: Increment visibility under a fixed carrier readout: test accuracy (%) over three seeds and 250 queries per seed. No hides the increment from every head; X grants access only to heads outside the selected set $\mathcal{T}$; Ours only to heads in $\mathcal{T}$; Full to all heads. The last two columns give the average marginal access effects in Equation 17, in percentage points. All conditions use the same raw carrier argmax within a row.

Average effects of the two access switches. Using the four accuracy values, define

$$\begin{aligned} \Delta_{\text{carrier}} &= \tfrac{1}{2}\big[(\text{Ours} - \text{No}) + (\text{Full} - \text{X})\big], \\ \Delta_{\text{other}} &= \tfrac{1}{2}\big[(\text{X} - \text{No}) + (\text{Full} - \text{Ours})\big]. \end{aligned} \tag{17}$$

Each effect averages the gain from one group's access with and without access by the other group. They sum to $\text{Full} - \text{No}$ before rounding. Any interaction, $\text{Full} - \text{Ours} - \text{X} + \text{No}$, is shared equally by this convention. The columns therefore describe average marginal effects of visibility on accuracy; they do not divide the model into independent circuits. The readout and complementary head group both change between the eight- and thirty-head rows.

The useful access routes depend on budget and setting. In the three $2 \to 5$ cells, access through thirty carriers retains most of the full-access accuracy gain. The eight-head results are more mixed: on dbpedia14, X reaches 86.67%, above Ours at 83.20%. At both Qwen $5 \to 10$ settings, the other-head access effect exceeds the carrier-access effect at both budgets. For Qwen3-8B with thirty heads, the effects are 22.1 versus 10.1 points; for eight heads, X reaches 60.53%, close to Full at 63.87%, while Ours reaches 37.60%.

These results reveal two complementary routes by which increment information reaches the carrier readout. Carriers can read the increment directly, but other heads can also transform newly visible evidence into contextual states that the selected writes subsequently expose. The balance between these routes varies with model and reader budget, supporting a view in which carrier readout is embedded in the broader network that constructs its input states.

D.8 Readout Dependence and Component Interactions

The model readout can further improve on a standalone classifier that already benefits from incremental evidence. At $5 \to 10$ on Llama-2-7B with SelfExtend and on Llama-3.1-8B/yahoo, thirty-head Ours (model) reaches 35.33% and 61.33%, exceeding both Ours (direct) and the corresponding Natural accuracies of 34.13% and 57.73%. In these settings, downstream computation successfully converts readable carrier evidence into stronger final predictions.

The opposite pattern on Qwen2.5-7B/TREC-fine highlights a complementary regime. At $5 \to 10$, thirty-head selective access improves direct readout from 41.20% to 52.67%, while model accuracy changes from 38.53% under Natural to 29.60%. Thus, the selected writes become more informative even when downstream processing does not translate that gain into better final decisions. Memory usefulness under a fixed readout and the accuracy of the model's full prediction rule are therefore distinct properties.

Together with the head–MLP ablations in Appendix B.8, these cases point to a separation between evidence formation and evidence utilization. Comparing model and direct readouts changes the contribution of later residual writes and final normalization, revealing how downstream computation can amplify, preserve, or counteract information already readable from the selected heads.

			8 heads			30 heads		
Model	Task	Level	Full	Label	Non-label	Full	Label	Non-label
Llama-3.1-8B	TREC-fine	2→5	44.00	44.40	35.47	47.07	47.07	37.07
Llama-3.1-8B	TREC-fine	5→10	50.93	51.73	48.53	52.00	52.40	52.80
Qwen2.5-7B	TREC-fine	2→5	41.33	41.07	28.67	43.47	45.20	33.47
Qwen3-8B	TREC-fine	2→5	40.80	43.33	32.00	43.47	46.27	34.13
Qwen3-4B	TREC-fine	2→5	41.20	40.80	34.40	45.87	45.73	37.47
Qwen3-4B	TREC-fine	5→10	40.40	38.93	36.40	41.33	42.80	37.33
Qwen3-14B	TREC-fine	2→5	45.33	46.27	38.53	49.20	49.07	41.60
Llama-3.1-8B	banking77	1→2	48.00	49.20	42.53	61.87	62.53	53.07
Llama-3.1-8B	clinc150	1→2	59.60	59.47	54.27	76.67	76.40	64.27
Llama-3.1-8B	dbpedia14	2→5	83.47	84.00	66.80	89.07	89.47	68.27
Llama-3.1-8B	dbpedia14	5→10	91.73	92.27	86.40	93.47	94.00	88.27
Llama-3.1-8B	yahoo	2→5	35.33	38.80	28.67	39.60	42.13	28.00
Llama-3.1-8B	yahoo	5→10	54.40	56.13	49.33	55.47	55.20	49.87
Llama-3.1-8B	yelp	2→5	21.87	22.67	20.40	24.40	27.60	21.07

Table 21: Accuracy (%) of the raw carrier readout with full-column, label-only, or non-label-only increment memory, within the same lean implementation. Each row uses the main evaluation's three seeds and 250 queries per seed. Full-column values can differ slightly from the main pipeline because of numerical implementation differences; the label columns are the main table's anchor results.

D.9 Anchor Memory and Matched Implementation Controls

The anchor variant retains the label positions of increment demonstrations while keeping the base, prefix, and live query positions available. These labels use the same cached contextual representations as in the full increment, preserving the demonstration context already encoded into the label states. For each row, Table 21 compares full-column, label-only, and non-label-only increment memory within the same lean implementation and on the same test draw, isolating the effect of restricting memory to particular positions. The table covers natural-language settings with all three matched controls available; the Qwen2.5-7B and Qwen3-8B TREC-fine $5 \rightarrow 10$ anchor results are reported in Table 2. The corresponding MONK controls are in Table 24.

At $5 \rightarrow 10$ on TREC-fine, thirty-head anchors reach 55.20% on Qwen2.5-7B and 43.87% on Qwen3-8B, compared with 52.67% and 42.27% for the main direct implementation. Eight-head anchors are slightly less accurate in both cells, indicating some dependence on the reader set. These results support label positions as compact contextual memory anchors: much of the useful increment signal can be retained when access is restricted to their cached states. The comparison keeps label-position keys and values together, and removing other increment positions naturally renormalizes attention. The matched controls below further separate this positional restriction from small implementation-level differences in the attention kernel.

The label-only increment preserves substantially more accuracy than its non-label complement in many rows, including the Qwen2.5, Qwen3-8B, and banking77 configurations. This extends the diagnostic evidence by showing that contextual label states provide a particularly information-dense memory interface in the access experiments as well. These states are constructed from the full demonstrations, so the result highlights where useful information is concentrated at readout time while allowing non-label positions to contribute upstream to the formation of those representations.

E Supplementary Rule-Learning Evaluation on MONK

The main experiments examine learning arbitrary class–label correspondences over natural-language inputs. This supplementary evaluation broadens the mapping demands to rules over categorical attributes. A template T renders an attribute vector a as text, and a rule r_z assigns its label: $f_z(T(a)) = r_z(a)$. MONK-1 and MONK-3 use fixed Boolean concepts involving attribute equality and value conditions (Thrun et al., 1991). The rules are not stated in the prompt, and held-out

attribute combinations test generalization. These tasks retain the possibility of useful pretrained representations while extending the evaluation beyond label reassignment.

We report all five MONK model–task–budget settings separately from the 18 natural-language settings summarized in the main text. The same carrier readouts, access interventions, and adapted baselines test the scope of the reuse design. These behavioral comparisons do not repeat the query-weight and value-intervention diagnostics used to motivate that design.

E.1 Task Construction and Evaluation

MONK-1 and MONK-3 are fixed Boolean concepts over the same universe of 432 robots, each described by six categorical attributes (Thrun et al., 1991). We render each robot with a fixed sentence template:

```
The robot has a {a1} head, a {a2} body, {a3}, holds a {a4},
wears a {a5} jacket, and {a6}.
Category: {label}
```

Attribute values follow the UCI code order: a_1 and a_2 use `round`, `square`, or `octagon`; a_4 uses `sword`, `balloon`, or `flag`; and a_5 uses `red`, `yellow`, `green`, or `blue`. The yes/no values of a_3 become `is smiling`/`is not smiling`, and those of a_6 become `wears a tie`/`does not wear a tie`. In the shared-bank evaluation, {`label`} is the cell's single-token abstract class label. Demonstrations are separated by blank lines, and the query ends at `Category:`; prediction selects among the candidate label tokens at the next position. The decision rule is not included in the prompt.

MONK-1 uses the rule $(a_1 = a_2) \vee (a_5 = 1)$; its demonstration bank is the released 124-example training set, and the test pool contains the 308 records outside that bank. MONK-3 uses its released 122-example training set, which includes $5\,\%$ label noise, and a disjoint pool of 310 records. The class counts are 62/62 and 154/154 in the MONK-1 bank and pool, and 62/60 and 146/164 for MONK-3. Each seed reserves twelve training examples per class for validation. Tables 22 and 23 report these two tasks at the displayed budgets. We use Llama-3.1-8B with the same abstract-label calibration, three prefix seeds, 250 test queries per seed, nested demonstration sets, and validation-based selection protocol described in Appendix A.1; the larger validation reservation above is specific to these tasks.

E.2 Results and Analysis

Additional evidence and direct prediction. Table 22 reports the complete access and readout comparison. All heads outperforms Natural in all five settings. On MONK-1 at $20 \rightarrow 40$, thirty-head selective direct readout reaches 84.67%, versus 79.60% for Natural and 85.33% for All heads. With full access, thirty-head direct readout matches or exceeds final-model accuracy in two of the five settings. The results extend the practical evaluation of the design beyond the natural-language tasks, with performance depending on the rule and demonstration budget.

The larger reader budget is not uniformly better. Thirty-head selective direct readout exceeds eight-head readout in three of the five settings. MONK-3 favors eight heads: expanding the set changes accuracy from 73.20% to 72.40% at $20 \rightarrow 40$. MONK-1 at $2 \rightarrow 5$ also slightly favors eight heads (54.93% versus 54.53%), whereas its larger demonstration budgets benefit from thirty. These results delimit the positive budget trend in the main tables: useful reader budgets depend on the task under the current ranking and raw aggregation rule.

Selective access has task-dependent limits. Relative to No access (direct), selective access improves the fixed readout in four of five settings with eight heads and all five with thirty heads. Full-access direct readout exceeds no access in all ten head-budget comparisons and exceeds selective access in nine, with one tie. We also observe a trend that full-network access can recover benefits that selective access alone does not preserve. Together with the access controls in Appendix D.7, these results emphasize that carrier predictions depend on contextual states supplied by the surrounding network.

Table 22: Test accuracy (%) on the supplementary MONK tasks under different memory access and prediction readouts. All rows use Llama-3.1-8B, three seeds, and 250 queries per seed. No access hides only the increment; base access remains available. Bold compares the six Ours columns. Anchor results use the lean implementation, with matched controls in Table 24.

					No access (direct)		Full access (direct)		Ours (model)		Ours (direct)		Ours (anchor)	
Model	**Task**	**Level**	Natural	All heads	8	30	8	30	8	30	8	30	8	30
Llama-3.1-8B	Monk-1	2→5	53.47	54.40	52.80	53.73	54.93	56.00	53.87	51.60	**54.93**	54.53	54.67	54.53
Llama-3.1-8B	Monk-1	5→10	57.07	58.67	55.07	56.53	58.53	59.47	56.80	57.47	56.27	**58.67**	56.67	57.60
Llama-3.1-8B	Monk-1	10→20	64.00	70.80	63.47	63.87	68.80	69.20	63.20	**66.00**	62.40	65.33	63.87	64.27
Llama-3.1-8B	Monk-1	20→40	79.60	85.33	76.53	77.47	84.53	84.80	79.60	**85.07**	78.93	84.67	79.20	83.07
Llama-3.1-8B	Monk-3	20→40	72.40	79.47	70.00	71.20	79.47	78.40	72.27	**74.13**	73.20	72.40	73.87	73.33

Table 23: Comparison with adapted vector baselines on the supplementary MONK tasks: test accuracy (%) over three seeds and 250 queries per seed, using Llama-3.1-8B. Natural and All heads provide base and full access, respectively. Bold marks the best adapted-method score in each row, excluding these two references.

					Baselines				TSLA		Ours (direct)	
Model	**Task**	**Level**	Natural	All heads	FV	TV	ICV	I2CL	TL	TR	8	30
Llama-3.1-8B	Monk-1	2→5	53.47	54.40	52.40	55.33	50.27	**60.53**	52.67	54.27	54.93	54.53
Llama-3.1-8B	Monk-1	5→10	57.07	58.67	57.47	56.13	49.47	57.47	56.93	53.47	56.27	**58.67**
Llama-3.1-8B	Monk-1	10→20	64.00	70.80	62.13	**65.47**	64.00	57.73	61.87	57.47	62.40	65.33
Llama-3.1-8B	Monk-1	20→40	79.60	85.33	76.13	78.13	75.20	62.93	74.67	61.73	78.93	**84.67**
Llama-3.1-8B	Monk-3	20→40	72.40	79.47	53.73	72.27	68.80	66.27	68.93	49.33	**73.20**	72.40

Comparison with adapted vector baselines. Table 23 retains the full baseline comparison. Thirty-head direct readout exceeds every vector baseline in three of five settings. On MONK-1 at $2 \to 5$, I2CL reaches 60.53%, above both direct-readout budgets; at $10 \to 20$, TV reaches 65.47%, slightly above thirty-head direct readout at 65.33%. On MONK-3, eight-head direct readout gives the highest adapted-method score. These mixed rankings extend the task dependence observed in the main evaluation and do not establish a uniformly preferable intervention or reader budget.

E.3 Matched Anchor-Memory Controls

Table 24 reports the MONK full-column, label-only, and non-label-only controls within the same lean implementation described in Appendix D.9. The label-only columns are the anchor results in Table 22; full-column values can differ slightly from that table's direct implementation.

The benefit of retaining label positions varies across these tasks. With thirty heads on MONK-1 at $20 \to 40$, label-only memory reaches 83.07%, compared with 84.40% for full-column memory and 77.73% for non-label memory. On MONK-3 at $20 \to 40$, the corresponding values are 73.33%, 72.27%, and 70.80%. Contextual label states can support useful compact memory, but the MONK controls do not show a uniform label-only advantage. As in the main evaluation, all retained states are constructed from full demonstrations, and restricting visible positions renormalizes attention.

F Extended Related Work

Pretraining priors and task specification. ICL combines information supplied by demonstrations with several kinds of prior structure: representations of inputs, associations between inputs and labels, preferences over tasks, and biases toward particular output tokens. Bayesian accounts formalize inference over latent concepts acquired during pretraining (Xie et al., 2022), while demonstration and calibration studies distinguish task identification, label semantics, and output preferences (Min et al., 2022; Pan et al., 2023; Zhao et al., 2021). Training distributions also influence which source of information a model uses: burstiness, rare classes, and varying label assignments can encourage in-context learning, and in-context and in-weights learning can coexist under suitable distributions (Chan et al., 2022). Flipped-label and semantically unrelated-label experiments probe whether models follow demonstrated correspondences when familiar associations conflict with or fail to specify them (Wei et al., 2023). Our arbitrary-label tasks retain useful pretrained representations while requiring demonstrations to supply the class–label correspondence; MONK extends the evaluation to rules over attribute combinations (Appendix E). The paired selection contrast targets

			8 heads			30 heads		
Model	Task	Level	Full	Label	Non-label	Full	Label	Non-label
Llama-3.1-8B	Monk-1	$2\rightarrow5$	55.07	54.67	51.87	54.80	54.53	52.40
Llama-3.1-8B	Monk-1	$5\rightarrow10$	56.40	56.67	55.07	58.13	57.60	57.60
Llama-3.1-8B	Monk-1	$10\rightarrow20$	62.53	63.87	62.93	65.33	64.27	63.60
Llama-3.1-8B	Monk-1	$20\rightarrow40$	78.67	79.20	76.80	84.40	83.07	77.73
Llama-3.1-8B	Monk-3	$20\rightarrow40$	73.60	73.87	68.67	72.27	73.33	70.80

Table 24: Matched memory-position controls on MONK: accuracy (%) of the raw carrier readout with full-column, label-only, or non-label-only increment memory in the same lean implementation. All rows use Llama-3.1-8B, three seeds, and 250 queries per seed. Full-column values can differ slightly from the main pipeline because of numerical implementation differences; the label columns match the anchor results in Table 22.

writes that change with the displayed mapping by cancelling contributions invariant to relabeling (Appendix B.2). This defines the information dependence being measured while allowing pretrained structure to participate in its computation.

Learning algorithms and contextual retrieval. Algorithmic accounts ask what learning procedure a forward pass implements. Studies of synthetic function classes establish settings in which Transformers learn predictors resembling regression or gradient-based updates (Garg et al., 2022; Akyürek et al., 2023; Von Oswald et al., 2023), while kernel accounts describe prediction through weighted combinations of demonstration information (Han et al., 2025). At the circuit level, induction heads and learned query–key metrics provide mechanisms for matching context and retrieving continuations or labels (Olsson et al., 2022; Singh et al., 2024; Yu & Ananiadou, 2024). These descriptions concern different levels of computation and need not be mutually exclusive. Head-level taxonomies further distinguish induction heads, semantic induction heads that match relations rather than tokens, retrieval heads defined by cross-task copying, and function-vector heads that encode task identity (Crosbie & Shutova, 2025; Ren et al., 2024; Wu et al., 2025; Yin & Steinhardt, 2025); carriers, which weight same-class demonstrations by identity rather than slot, lie nearest semantic induction. A useful connection outside language-model analysis is Matching Networks, which classify by attention-weighted support labels and can condition their embeddings on the support set (Vinyals et al., 2016). Our carrier decomposition likewise connects to kernel smoothing (Nadaraya, 1964; Watson, 1964), with contextual value–output projections supplying the class votes. Both the similarities and the votes are therefore products of the network's preceding computation. Query-weight and value-intervention controls test their empirical role within the selected writes. Retrieval weights also carry content-independent structure: order and position effects (Lu et al., 2022; Liu et al., 2024a) and attention sinks linked to massive activations (Xiao et al., 2024; Sun et al., 2024; Gu et al., 2025), which our reorder controls and the invariance cancellation in TL-score accommodate rather than remove (Appendix B.4). The explicit kNN comparison in Appendix D.3 examines another prediction rule over the same demonstration bank using hidden states from the same frozen model.

Attention, MLPs, and the construction of task states. Feed-forward layers have been analyzed as parameterized key–value memories whose activations contribute vocabulary-directed information (Geva et al., 2021). Factual-recall studies further trace interactions between modules: early MLPs enrich subject representations, and later attention heads extract relevant attributes from those states (Geva et al., 2023). These findings provide a precedent for separating the construction of a readable state from the operation that retrieves it. MLPs also enter formal accounts of ICL. For linear regression with a nonzero-mean Gaussian task prior, adding an MLP component allows a linear Transformer block to approach Bayes-optimal prediction, with a correspondence to gradient descent from a learned initialization (Zhang et al., 2024). Separately, MLP and MLP-Mixer architectures can learn in context on controlled regression and classification tasks (Tong & Pehlevan, 2025). These results concern different architectures and settings, but together motivate examining how attention and feed-forward computation contribute to task adaptation. In our models, carriers read contextual keys and values using queries formed from the receiver's residual stream. Label-anchor and receiver controls test the usefulness of these contextual states, while the access factorial and head–MLP ablations probe dependencies on surrounding computation (Appendices D.9, D.7, and B.8). Local-

izing mapping-sensitive readers thus leaves a further question about how the network constructs and transforms the states they use.

Component readouts and compensatory computation. Residual-stream decomposition supports reading individual components through the model's output directions (Elhage et al., 2021). Closely related work finds attention heads and MLPs whose individual classification accuracy exceeds the full model's, alongside components with below-chance or label-biased predictions, and learns to reweight their contributions (Chang et al., 2024). Our paired mapping contrast and retrieval controls examine which demonstrated relationships support such component-level predictions. Interventions introduce a further distinction because surviving components can respond to the perturbation. In factual recall, the Hydra study identifies both increased contributions from downstream attention after an attention-layer ablation and late-MLP counterbalancing whose suppressive effect weakens when the upstream signal is removed (McGrath et al., 2023). Subsequent work finds prompt-dependent, incomplete self-repair involving final-normalization rescaling and reduced erasure by downstream neurons (Rushing & Nanda, 2024). Because a backup component is invisible to single-unit scores until the primary is removed, conditional or joint ablation is needed to expose it (Gong et al., 2026). Copy suppression provides an attention-based example in which suppressing overconfident copying improves calibration (McDougall et al., 2024). In our MLP diagnostic, mapping-specific writes oppose the true-class margin, while joint head–MLP ablations in the Llama settings produce larger accuracy drops than the sums of individual drops (Appendices B.6 and B.8). The ablation interactions are consistent with compensation, although accuracy changes alone do not identify the compensating pathways. Direct and model readouts additionally compare selected class-directed writes with the decision produced by the full residual computation and final normalization (Appendix D.8). Our clean-value repair is an externally imposed intervention, distinct from the endogenous response studied as self-repair.

Context compression and reusable memory. Context-reuse methods can be compared by when they compress information, what representation they retain, and how a later query accesses it. Gist tokens, AutoCompressors, and in-context autoencoders learn compact representations through additional training (Mu et al., 2023; Chevalier et al., 2023; Ge et al., 2024). Activation-based methods retain demonstration-derived vectors and introduce them through interventions in a receiver (Hendel et al., 2023; Todd et al., 2024; Liu et al., 2024b; Li et al., 2025b; Yang et al., 2026). Dong et al. (2026) argue that a task vector behaves as a single demonstration formed by combining the originals and cannot represent high-rank mappings; a 36-class arbitrary assignment is such a mapping. Selective TL Memory retains selected states after full-prefix encoding and recomputes their attention weights for each query. Its label anchors therefore preserve contextualized states that have already integrated surrounding demonstration information, connecting memory selection to label-anchor analyses (Wang et al., 2023; Cho et al., 2025) and to evidence that task representations condense at particular tokens rather than uniformly across the context (Li et al., 2025a). Receiver conditioning matters for both forms of reuse: cached evidence must be accessible to the receiver's queries, and an injected vector interacts with information already present in its residual stream. The zero-base and base-subtraction controls examine these dependencies (Appendices D.1 and A.2); a fixed stored vector can still have query-dependent effects through subsequent computation. Prefix caching, selective KV retention, and block-level retrieval over cached demonstration groups address the associated storage and attention costs (Gim et al., 2024; Zhang et al., 2023; Li et al., 2024; Xiao et al., 2025b;a); the last selects demonstrations per query for all heads, whereas our restriction selects heads and label positions, so the two could be composed. Our prefix-cached benchmark measures shared memory and online prediction cost, with one-time preparation excluded from the online measurements.

G Future Work

A natural next step is to let the receiver run a standard K_{base} prompt while still retrieving the cached increment states. Our current setup retains full-prompt positions in the *Natural* condition, allowing the receiver to use cached demonstration states directly without additional positional-encoding adjustments. This enables comparisons of memory visibility within a shared positional layout, although *Natural* need not match the accuracy of standard K_{base} prompting. Future work could investigate how positional structure interacts with task-dependent similarity in the retrieval kernel. Understanding when positional information supports or distorts the use of demonstration evidence

could inform the design of positional representations and attention mechanisms for retrieving external memories under more flexible context layouts.

Further work could establish how the observed mechanisms and their practical benefits generalize across training regimes and context scales. The differences between Llama-3.1 and Qwen in retrieval, intervention responses, and component interactions motivate tracing how these properties emerge during training. Comparisons of pretrained checkpoints cannot isolate their causes; controlled training studies and intermediate-checkpoint analyses could clarify the roles of data distributions and optimization histories (Chan et al., 2022). On the application side, our available computational resources did not permit systematic evaluation with contexts exceeding $100{,}000$ tokens. Experiments at this scale could test whether the sender–receiver architecture preserves retrieval quality and the benefits of selective storage as demonstration banks grow, while accounting for both sender-side memory preparation and receiver-side inference costs.