Capturing In-Context Learning Dynamics with Task Operators

Guangzhi Xiong
University of Virginia
guangzhi@virginia.edu

Zhenghao He
University of Virginia
zhenghao@virginia.edu

Bohan Liu
University of Virginia
bohan@virginia.edu

Sanchit Sinha
University of Virginia
sanchit@virginia.edu

Wenqian Ye
University of Virginia
wenqian@virginia.edu

Aidong Zhang
University of Virginia
aidong@virginia.edu

Abstract

In-context learning (ICL) enables language models to perform new tasks from demonstrations without weight updates. However, every ICL inference requires processing the full set of examples, resulting in inefficient deployments, and how ICL works mechanistically is not fully understood. Prior work compresses ICL into fixed activation vectors extracted from specific layers or positions, but these input-independent interventions fail on complex tasks where the output depends on fine-grained interactions with the input. By analyzing the ICL forward pass, we show that each attention head's output is an affine transformation of its context-masked counterpart, and that the parameters of this transformation are empirically stable across samples for a given task. Building on this, we introduce Task Operator (TO), which replays this transformation as an analytically derived update to the attention output projection. Across lexical, algorithmic, and reasoning tasks, TO achieves the best overall performance among prior methods and substantially narrows the gap between zero-shot inference and ICL. We further show that the extracted knowledge concentrates in a task-specific sparse circuit across layers and positions, and that averaging operators from disjoint demonstration batches enables effective many-shot scaling without expanding the context window. Our code is available at `https://github.com/gzxiong/task_operator`.

1 Introduction

In-context learning (ICL) enables large language models (LLMs) to quickly adapt to new tasks by providing input-output demonstrations in the context [2]. This allows LLMs to improve performance on tasks where they fail in the zero-shot learning (ZSL) setting, without any additional training. However, ICL requires processing all demonstrations for the inference of each new input, which can be computationally expensive and inefficient even with KV caching [29]. Therefore, it is important to understand the mechanisms underlying ICL and to develop methods that can distill the ICL representation into a compact form, enabling efficient replay at inference time without repeatedly processing demonstrations.

Recent approaches tackle this inefficiency by developing methods that compress ICL into lightweight, fixed representations, allowing models to bypass demonstrations and efficiently replay task knowledge during inference [13, 36, 23, 30, 19]. These methods typically extract task-specific activation vectors from particular layers or token positions during ICL and inject them as fixed interventions during zero-shot inference. While effective for simple tasks with clear input-output mappings, these approaches

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

fail when output generation requires complex multi-step reasoning over the input content, indicating that the activation vectors do not faithfully capture the knowledge encoded in the ICL process.

By analyzing how demonstrations reshape the model's internal computation, we show that at every attention head, the output under ICL is precisely an affine transformation of its context-masked counterpart, with the parameters of these transformations showing high consistency across samples (Appendix D). With extracted parameters for all attention heads, we show that the effects of demonstrations can be exactly represented as an input-dependent reparameterization of the attention output projection.

Building on this analysis, we introduce the Task Operator (TO), a training-free method that captures ICL knowledge as affine transformation parameters and replays it during inference to dynamically update the attention outputs. Through experiments on four LLMs across eight tasks, we demonstrate that TO faithfully captures ICL knowledge and outperforms prior knowledge extraction methods overall when replaying the knowledge in zero-shot inference. Causal circuit analysis and ablation studies reveal that the key sites for storing ICL knowledge are sparse and task-specific, explaining why previous methods that extract knowledge from fixed locations fall short. Scaling analysis further demonstrates the effectiveness of TO under many-shot ICL, where knowledge is averaged from disjoint mini-batches of demonstrations, highlighting its scalability within a fixed context budget. Our main contributions are:

- We propose Task Operator (TO), a training-free method that captures ICL knowledge as an affine transformation of the attention output, enabling efficient and dynamic replay of ICL knowledge during zero-shot inference.
- Evaluation across four LLMs and eight tasks demonstrates that TO achieves the best overall performance, highlighting the importance of dynamic interventions for faithfully capturing and replaying ICL knowledge.
- Our analysis reveals the diverse and sparse circuits leveraged by ICL across different tasks, and shows that TO enables efficient many-shot scaling by averaging knowledge extracted from disjoint demonstration batches.

2 Related Work

In-context learning mechanisms. In-context learning enables language models to perform new tasks from input-output demonstrations without parameter updates [2]. Several theoretical accounts have been proposed to explain this behavior. Xie et al. [39] interpret ICL as implicit Bayesian inference over latent document-level concepts induced by long-range coherence in pretraining data. Dai et al. [6] connect transformer attention to gradient descent and argue that ICL implements implicit fine-tuning through meta-gradients computed from demonstrations. Mechanistic studies identify internal structures that support ICL. Olsson et al. [26] link induction heads to the emergence of ICL during training, while Yin and Steinhardt [41] show that few-shot ICL is driven primarily by function vector heads rather than induction heads, with relevant heads distributed across layers in a task-dependent manner. Recent work further studies how task representations emerge and what forms they can take. Han et al. [12] propose an encoder-decoder account in which early layers encode task information and later layers decode it, while Dong et al. [7] analyze the linear combination structure of task vectors and prove that additive task vectors are restricted to rank-one meta predictors. Our work builds on this line of inquiry by deriving an exact per-head affine form for the effect of demonstrations and using it to replay ICL behavior without reprocessing the demonstration context.

Compressing ICL into task representations. A parallel line of work seeks to compress ICL into reusable representations that can be applied without repeatedly processing demonstrations. Ilharco et al. [17] introduce task vectors in weight space, computed from the difference between fine-tuned and pre-trained model weights. Hendel et al. [13] show that ICL creates analogous task vectors in activation space, where the hidden state at a separator token after demonstrations encodes a reusable task representation. Todd et al. [36] extend this idea to function vectors by aggregating outputs from causally important attention heads, enabling transfer to zero-shot and natural text settings. Other training-free methods construct task representations from broader activation statistics. Liu et al. [23] propose in-context vectors that use principal directions from demonstration hidden states. Li et al. [19] introduce state vectors that concatenate intermediate activations and use momentum based

optimization at test time, while Li et al. [21] compress demonstrations into context vectors through mean shift in the residual stream. These methods provide efficient alternatives to full ICL, but they primarily replay task knowledge through fixed additive interventions. In contrast, task operators include both multiplicative scaling and additive context contributions at each attention head, allowing the replayed intervention to depend on the test-time hidden state.

Richer task vector and activation interventions. Recent methods address the limitations of fixed additive vectors by introducing input dependence, learned components, or richer intervention families. DyVec [3] dynamically segments multi-head attention output projections for each input, but requires REINFORCE-based training. Adaptive Task Vectors [18] train a small language model to generate a task vector conditioned on each input, and In-Context Routing [20] trains an input-conditioned router that adjusts attention logits. LIVE [28] and LTV [32] learn per-layer vectors via gradient optimization. Related work on activation steering also shows that additive interventions alone can be insufficient. Singh et al. [34] derive affine steering functions for residual stream interventions in concept erasure and bias mitigation. Marshall et al. [24] show that refusal behavior is better characterized as an affine function of internal representations, while Stoehr et al. [35] propose activation scaling via element-wise multiplicative factors. Hernandez et al. [15] show that, for a subset of relations, relation decoding in language models is well approximated by an affine map derived from a first-order approximation of the model. In contrast, our per-head affine form is an exact identity for each input. Conceptor-based methods use soft projection matrices for activation engineering [30], with subsequent work deriving compositional affine steering mechanisms [1]. These approaches show that richer interventions can improve model steering, but they either introduce additional learned or optimized components, or estimate steering transformations empirically from activation statistics rather than deriving an analytical knowledge extraction rule. In contrast, our method derives input-dependent affine parameters directly from the ICL forward pass and replays them through dynamic updates to the attention output projection.

Mechanistic interpretability and circuit localization. Mechanistic interpretability aims to explain model behaviors in terms of internal components and their interactions. Elhage et al. [8] and Olsson et al. [26] develop the conceptual framework of circuits as computational subgraphs within transformers. Wang et al. [38] present a detailed circuit analysis of indirect object identification in GPT-2, identifying attention heads organized into functional classes. Geva et al. [9] trace information flow for factual recall, showing how MLP layers enrich subject representations before attention heads extract relevant attributes. Causal intervention techniques, including activation patching [37, 25] and path patching [10], provide tools for isolating the contributions of specific model components. Our work follows this tradition by using causal analysis to localize where ICL knowledge is stored, showing that the relevant sites form sparse and task-specific circuits, and using these localized effects to construct task operators that can be replayed during zero-shot inference.

3 Task Operators in In-Context Learning

We introduce Task Operator (TO), a method for transferring in-context learning (ICL) behavior to zero-shot inference via dynamic interventions at multiple sites throughout the model. Our approach consists of three main components: First, we derive the affine structure underlying ICL's effect on attention heads and show how to extract the relevant parameters from ICL forward passes (Section 3.1). Second, we describe how the extracted operator is replayed during zero-shot inference (Section 3.2). Third, we present a sparsification procedure that identifies and retains only the most important (layer, token) sites for intervention (Section 3.3). Figure 1 illustrates the overall framework.

3.1 Knowledge Extraction

At layer ℓ, for a query token at position t, each attention head h computes attention probabilities $\alpha_{t,i}^{(\ell,h)} = \exp(s_{t,i}^{(\ell,h)})/Z_t^{(\ell,h)}$ over all preceding positions $i \leq t$, where $s_{t,i}^{(\ell,h)}$ are the pre-softmax scores and $Z_t^{(\ell,h)} = \sum_{i \leq t} \exp(s_{t,i}^{(\ell,h)})$. The per-head output $\mathbf{u}_t^{(\ell,h)} = \sum_{i \leq t} \alpha_{t,i}^{(\ell,h)} \, \mathbf{v}_i^{(\ell,h)}$ aggregates value vectors weighted by these probabilities. The heads are concatenated and projected through the output projection, $\mathrm{Attn}_t^{(\ell)} = \mathbf{W}_O^{(\ell)} \mathbf{U}_t^{(\ell)} + \mathbf{b}_O^{(\ell)}$, where $\mathbf{U}_t^{(\ell)} = [\mathbf{u}_t^{(\ell,1)}; \ldots; \mathbf{u}_t^{(\ell,H)}] \in \mathbb{R}^d$ and each head has dimension $d_h = d/H$.

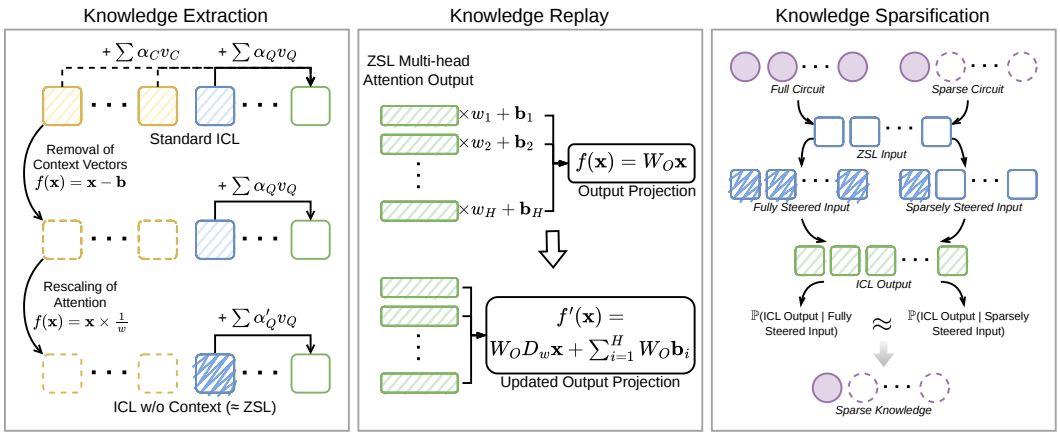

Figure 1: Overview of the Task Operator framework: extracting per-head affine parameters $(w, \mathbf{b})$ from ICL forward passes (*left*), replaying them via an updated output projection at zero-shot inference (*middle*), and sparsifying to the highest-importance sites (*right*). The yellow blocks denote the context tokens, and the blue blocks denote the tokens of the input query.

Affine decomposition. For clarity, we fix a layer ℓ, a head h, and a query position t, and omit these indices in Equations (1)–(3). Let $\mathcal{C}$ denote the context token positions (the demonstration examples) and $\overline{\mathcal{C}} = \{i : i \le t,\ i \notin \mathcal{C}\}$ the non-context positions accessible from the query position. The head output decomposes as

$$\mathbf{u} \;=\; \underbrace{\textstyle\sum_{i\in\overline{\mathcal{C}}} \alpha_i\, \mathbf{v}_i}_{\text{non-context contribution}} \;+\; \underbrace{\textstyle\sum_{i\in\mathcal{C}} \alpha_i\, \mathbf{v}_i}_{\text{context contribution} =: \mathbf{b}} \quad . \tag{1}$$

Define the context-masked head output, obtained by renormalizing over non-context positions only, as $\mathbf{u}' = \sum_{i\in\overline{\mathcal{C}}} \alpha'_i\, \mathbf{v}_i$, where $\alpha'_i = \exp(s_i)/Z'$ and $Z' = \sum_{i\in\overline{\mathcal{C}}} \exp(s_i)$. The non-context contribution in (1) can then be rewritten as

$$\sum_{i\in\overline{\mathcal{C}}} \alpha_i\, \mathbf{v}_i = \frac{Z'}{Z} \sum_{i\in\overline{\mathcal{C}}} \alpha'_i\, \mathbf{v}_i = w\, \mathbf{u}', \tag{2}$$

where $w \equiv Z'/Z \in (0, 1]$ is the fraction of softmax mass on non-context positions. Combining Equations (1) and (2) gives the per-head affine identity

$$\mathbf{u} = w\, \mathbf{u}' + \mathbf{b}. \tag{3}$$

In words, the head output under ICL equals its context-masked output, rescaled by the attention mass that remains on non-demonstration tokens, plus the value contribution retrieved from the demonstrations. This identity is exact for each input and holds for arbitrary attention patterns and context spans.

Output projection reparameterization. Restoring the indices (ℓ, h, t), let $\mathbf{D}_t^{(\ell)} = \mathrm{diag}\big(\underbrace{w_t^{(\ell,1)}, \ldots, w_t^{(\ell,1)}}_{d_h}, \ldots, \underbrace{w_t^{(\ell,H)}, \ldots, w_t^{(\ell,H)}}_{d_h}\big) \in \mathbb{R}^{d\times d}$ and $\mathbf{B}_t^{(\ell)} = [\mathbf{b}_t^{(\ell,1)}; \ldots; \mathbf{b}_t^{(\ell,H)}] \in \mathbb{R}^d$. Concatenating Equation (3) across heads and applying $\mathbf{W}_O^{(\ell)}$ gives

$$\mathrm{Attn}_t^{(\ell)} = \mathbf{W}_O^{(\ell)}\big(\mathbf{D}_t^{(\ell)}\, \mathbf{U}_t'^{(\ell)} + \mathbf{B}_t^{(\ell)}\big) + \mathbf{b}_O^{(\ell)} = \underbrace{\mathbf{W}_O^{(\ell)}\mathbf{D}_t^{(\ell)}}_{\widetilde{\mathbf{W}}_{O,t}^{(\ell)}}\, \mathbf{U}_t'^{(\ell)} \;+\; \underbrace{\mathbf{W}_O^{(\ell)}\mathbf{B}_t^{(\ell)} + \mathbf{b}_O^{(\ell)}}_{\widetilde{\mathbf{b}}_{O,t}^{(\ell)}}. \tag{4}$$

The modified weight $\widetilde{\mathbf{W}}_{O,t}^{(\ell)} = \mathbf{W}_O^{(\ell)}\mathbf{D}_t^{(\ell)}$ is a diagonal scaling of $\mathbf{W}_O$ on its input side, where each of the H head channels can be independently scaled by its corresponding $w_t^{(\ell,h)}$. Since $\mathbf{D}_t^{(\ell)}$ and $\mathbf{B}_t^{(\ell)}$ are computed from each input, Equation (4) expresses the ICL effect as an input-dependent reparameterization of the output projection.

Extraction procedure. Since the distribution of ICL-critical positions may vary across tasks [33], we extract from all token positions across all layers rather than selecting a fixed subset. When the ICL effect is negligible at a site, the parameters are close to identity ($w \approx 1$, $\mathbf{b} \approx \mathbf{0}$), thus having minimal impact when replayed at test time. Given an ICL prompt, we run the LLM and record pre-softmax attention scores at each layer. For each site (ℓ, t), we compute the per-head scaling factor via the numerically stable form

$$w_t^{(\ell,h)} = 1 - \exp\Big(\mathrm{logsumexp}_{i\in\mathcal{C}}\big(s_{t,i}^{(\ell,h)}\big) - \mathrm{logsumexp}_{i\le t}\big(s_{t,i}^{(\ell,h)}\big)\Big), \tag{5}$$

and the context bias $\mathbf{b}_t^{(\ell,h)} = \sum_{i\in\mathcal{C}} \alpha_{t,i}^{(\ell,h)}\, \mathbf{v}_i^{(\ell,h)}$.

For the fixed template tokens in the query block, we extract a set of operators across layers for each template position. For the query content tokens that vary across examples, we extract operators across content tokens and average them to obtain a single operator representing content positions in general. We also extract operators for the effect of ICL on the output tokens, which are computed on the tokens generated with the ICL prompt and averaged across output positions to form a unified generation operator. Our experiments demonstrate the effectiveness and importance of including the averaged operators for both content and generation positions to faithfully extract ICL knowledge.

To obtain a stable and reliable representation of the ICL effect induced by a fixed set of demonstrations, we compute the operator parameters on a set of m validation examples without ground truth labels, and average these parameters across examples. While Equation (3) holds exactly for each example, the averaged parameters provide a task-level approximation, which is supported by the cross-sample stability of the extracted parameters (Appendix D.1). When multiple sets of demonstrations are available, we can further average the parameters across demonstration sets following [16] to obtain a more robust and task-level representation of the ICL knowledge.

3.2 Knowledge Replay

At test time, given a new query x_q, we construct a zero-shot prompt P_{ZSL} with the same query template as the ICL prompt P_{ICL} but without demonstrations. We then run the LLM on P_{ZSL} and inject the extracted operator parameters at the mapped positions in the query template and content tokens. At each mapped position t', the original attention output is replaced with the values computed using the updated output projection layer:

$$\mathrm{Attn}_{t'}^{(\ell)} \;\leftarrow\; \widetilde{\mathbf{W}}_{O,t}^{(\ell)}\, \mathbf{U}_{t'}^{\mathrm{ZSL}} + \widetilde{\mathbf{b}}_{O,t}^{(\ell)}, \tag{6}$$

After the model steering during the prompt encoding, the LLM will then generate autoregressively with the task operator for generation positions installed at each decoding step, until the end of generation or a predefined maximum decoding length is reached. Although the operator parameters are fixed at test time, the intervention remains input-dependent, as the scaling factors multiply the current zero-shot head outputs. Since the operator only rescales and shifts the attention outputs, replay has essentially the same per-query cost as zero-shot inference (Appendix H.2).

3.3 Knowledge Sparsification

The full operator spans all layers and positions for model steering, but many sites may contribute negligibly to the model output. To understand how ICL knowledge is distributed across sites and to obtain an efficient, interpretable operator, we sparsify the full operator by intervening at each site and measuring its causal effect on the output distribution.

For each (layer, token) site i and each of m validation samples v, we compute the next-token negative log likelihood (NLL) on the teacher-forced ICL-generated tokens under three conditions: (1) with the full task operator, $\mathrm{NLL}_{\mathrm{full}}[v]$; (2) with all interventions disabled, $\mathrm{NLL}_{\mathrm{ZSL}}[v]$; and (3) with the operator applied everywhere except site i, which is patched to identity ($w \leftarrow 1$, $\mathbf{b} \leftarrow \mathbf{0}$ at site i only), giving $\mathrm{NLL}_{\mathrm{perturb}}[v, i]$. The fraction of the operator's NLL gain that remains when site i is perturbed defines the per-sample recovery:

$$\mathrm{recovery}[v, i] = \frac{\mathrm{NLL}_{\mathrm{ZSL}}[v] - \mathrm{NLL}_{\mathrm{perturb}}[v, i]}{\mathrm{NLL}_{\mathrm{ZSL}}[v] - \mathrm{NLL}_{\mathrm{full}}[v]}. \tag{7}$$

The per-sample importance is given by the loss in recovery, $\text{importance}[v, i] = 1 - \text{recovery}[v, i]$, and can be averaged across validation samples to provide a holistic view of how importance is distributed across sites. For robust site ranking, we use reciprocal rank fusion [5] to aggregate per-sample rankings, reducing sensitivity to NLL-scale variance across samples. We use knowledge sparsification mainly for understanding the ICL mechanism, with additional performance results available in Appendix E, where we show that the full operator remains a robust choice across tasks.

4 Experiments

4.1 Experimental Setup

We evaluate four LLMs from different model families and sizes, including Qwen3-4B, Qwen3-8B [40], Llama3.2-3B, and Llama3.1-8B [11]. Our evaluation covers eight tasks of varying difficulty, with two lexical tasks (Translation and Linguistic), three algorithmic tasks (Uppercase, Reverse, Deduplicate), and three reasoning tasks (GSM8K, MATH500, GPQA-Diamond). For lexical and algorithmic tasks, we use 128 held-out test samples for evaluation. For reasoning tasks, we rely on the standard test sets (GSM8K with 1319 samples, MATH500 with 500 samples, and GPQA-Diamond with 198 samples) and measure performance by exact match between the model's final answer and the reference answer. More details about the datasets are provided in Appendix B.

We compare our proposed Task Operator (TO) with four ICL knowledge extraction baselines: Task Vectors (TV) [13], Function Vectors (FV) [36], In-Context Vectors (ICV) [23], and Conceptors [30]. For all methods, we use the same $K = 8$ demonstrations for knowledge extraction and 32 answer-excluded validation samples for hyperparameter selection. We further compare with two trained input-dependent methods, ATV [18] and ICR [20], in Section 4.2. We also report the performance of zero-shot learning (ZSL) and standard 8-shot ICL as references to contextualize the results. All examples use the same prompt template, and for reasoning tasks, we append a "Let's think step by step" instruction to elicit chain-of-thought reasoning. Further implementation details for TO and the baselines are provided in Appendix C. Additional experiments on the stability of extracted parameters, TO performance with sparsified circuits, robustness to prompt template choice, and cross-task transfer of extracted knowledge are available in Appendices D - G.

4.2 Performance Comparison

Table 1 presents results across all models and tasks. Compared to other ICL knowledge extraction and replay methods, TO consistently achieves the highest accuracy on nearly every task-model combination, except in cases where ICL itself provides minimal or negative improvement over ZSL.

On lexical tasks and simple algorithmic tasks like Uppercase, where prior methods such as TV already recover much of the ICL gain, TO matches or exceeds the best baseline, achieving near-ICL performance. ICV is an exception that stays close to ZSL, even when its steering strength is tuned on the validation set for the lexical tasks (Appendix H.1), likely because it applies the same intervention at all positions, whereas Sections 4.3 and 4.5 show that different sites play distinct roles. For more complex algorithmic tasks such as Reverse and Deduplicate, where output generation depends intricately on the input sequence, the baselines generally fail to capture and replay the ICL knowledge. This highlights their inherent limitations in representing the complex ICL knowledge required by these tasks. In contrast, TO effectively recovers a substantial portion of the ICL gain, showing its ability to capture the complex input-dependent knowledge induced by ICL.

On reasoning tasks, models already show strong zero-shot performance due to clear instructions and question formats, and ICL further improves accuracy in most cases. While the baselines improve over zero-shot, they remain far from ICL performance. In contrast, when ICL brings clear gains, as on the Qwen models, TO achieves a significant improvement over the baselines and recovers much of the ICL gain, demonstrating its ability to capture the complex reasoning patterns provided by the demonstrations.

Notably, TO does not simply outperform the baselines but also closely mimics the model's behavior under ICL. In task-model combinations where ICL provides only marginal improvement over zero-shot or even degrades performance, TO also fails to improve over ZSL. For example, on GPQA and MATH500 for both Llama models, where ICL gains are small or negative, TO shows correspondingly limited or no improvement, and in a few cases falls slightly below ZSL. This indicates that TO

Table 1: Test accuracy (%) across models and tasks. ICL denotes performance with $K = 8$ examples provided. The scores without gray shading denote the performance with no in-context demonstrations.

Model	Method	Lexical		Algorithmic			Reasoning		
		Translation	Linguistic	Uppercase	Reverse	Deduplicate	GSM8K	MATH500	GPQA
Qwen3 (4B)	ICL	69.53	73.44	100.00	43.75	24.22	89.39	55.80	37.88
	ZSL	0.00	0.00	0.00	0.00	0.00	74.53	23.60	17.68
	TV	43.75	49.22	92.19	1.56	0.00	81.05	28.00	21.21
	FV	0.00	17.97	94.53	0.00	16.41	73.16	26.00	15.15
	ICV	0.00	0.00	0.00	0.00	0.00	74.83	23.80	20.20
	Conceptor	38.28	50.78	96.09	0.78	0.78	79.83	26.20	16.16
	TO (Ours)	**67.19**	**71.09**	**100.00**	**39.06**	**17.19**	**86.20**	**51.20**	**33.33**
Qwen3 (8B)	ICL	68.75	82.81	100.00	67.19	76.56	90.75	61.20	34.85
	ZSL	0.00	0.00	0.00	0.00	0.00	58.76	19.00	21.21
	TV	49.22	64.84	94.53	9.38	9.38	73.69	26.00	22.73
	FV	36.72	53.91	98.44	21.09	57.03	63.61	22.40	19.70
	ICV	0.00	1.56	0.00	0.00	0.00	59.74	20.00	21.21
	Conceptor	49.22	60.94	91.41	4.69	4.69	70.74	25.40	18.18
	TO (Ours)	**67.19**	**75.00**	**100.00**	**60.16**	**65.62**	**78.62**	**50.60**	**33.33**
Llama3.2 (3B)	ICL	66.41	75.78	100.00	21.88	33.59	69.98	27.20	24.24
	ZSL	0.00	3.91	0.00	0.00	0.00	58.15	24.80	26.26
	TV	30.47	71.09	86.72	0.78	2.34	59.06	**27.60**	25.76
	FV	39.84	63.28	93.75	1.56	2.34	62.62	26.00	24.75
	ICV	0.00	4.69	0.00	0.00	0.00	58.00	24.80	**30.81**
	Conceptor	28.12	1.56	87.50	0.00	3.12	57.32	26.80	23.74
	TO (Ours)	**67.97**	**75.78**	**100.00**	**31.25**	**24.22**	**67.78**	25.20	22.73
Llama3.1 (8B)	ICL	71.09	82.81	100.00	48.44	37.50	80.14	28.80	26.26
	ZSL	1.56	10.94	0.00	0.00	0.00	37.91	27.60	23.74
	TV	53.12	75.00	88.28	6.25	2.34	41.32	26.00	20.71
	FV	64.06	63.28	50.78	6.25	2.34	41.70	28.20	23.23
	ICV	1.56	10.16	0.00	0.00	0.00	39.58	26.60	**26.26**
	Conceptor	47.66	41.41	82.03	7.03	2.34	38.67	**31.80**	25.25
	TO (Ours)	**71.88**	**81.25**	**100.00**	**51.56**	**28.12**	**75.82**	25.40	23.74

faithfully captures the internal mechanisms of ICL rather than providing a generic performance boost, reinforcing the validity of TO as a tool for understanding and replicating ICL behavior.

To assess the variability from sampling demonstrations, we repeat the comparison on Qwen3-4B and Qwen3-8B with multiple random seeds and a fixed test set (Table 2). Although accuracy varies with the sampled demonstrations, as is typical for ICL [42, 27], the advantage of TO is consistent. On Deduplicate, a paired comparison over the same seeds shows that TO outperforms FV ($p \approx 0.01$) and TV ($p < 0.01$) on both models, and on GPQA, TO exceeds all baselines by a clear margin.

Table 2: Accuracy (%) with randomly sampled demonstrations, reported as mean $\pm$ standard deviation over 10 seeds for Deduplicate and 5 for GPQA.

Method	Qwen3 (4B)		Qwen3 (8B)	
	Deduplicate	GPQA	Deduplicate	GPQA
ZSL	0.0	17.7	0.0	21.2
TV	1.0±0.5	18.5±1.6	6.9±2.7	23.0±2.3
FV	16.6±5.1	17.5±1.8	55.8±5.8	20.3±1.2
TO (Ours)	**26.9±9.8**	**29.1±3.5**	**63.9±8.7**	**29.5±2.7**

Beyond Conceptor, a training-free input-dependent baseline, we compare TO with two trained input-dependent methods, ATV [18] and ICR [20], each trained with $K \in \{8, 32, 128\}$ examples. As shown in Table 3, ATV benefits from more training data, whereas ICR remains weak in this low-data regime. In contrast, TO, which uses only eight demonstrations and no training, outperforms both methods in every setting except Qwen3-4B Deduplicate, where only ATV with $16\times$ more training examples surpasses it, suggesting that its analytically derived operator captures input-dependent knowledge that these trained methods need substantially more data to learn.

Table 3: Accuracy (%) of trained input-dependent methods with K training examples, compared with training-free TO using K=8 demonstrations.

Model	Task	ATV (K)			ICR (K)			TO (K)
		8	32	128	8	32	128	8
Qwen3 (4B)	Deduplicate	3.1	15.6	**26.6**	10.2	6.2	13.3	17.2
	Translation	10.2	25.0	48.4	3.9	6.2	4.7	**67.2**
Qwen3 (8B)	Deduplicate	56.2	42.2	60.9	32.0	42.2	42.2	**65.6**
	Translation	50.8	60.2	64.8	2.3	4.7	5.5	**67.2**

4.3 Sparse Circuit Analysis

We use the sparsification procedure from Section 3.3 to identify which sites are most important for the ICL behavior captured by TO. Figure 2 presents per-site importance maps for the algorithmic tasks on Llama3.1-8B, revealing both shared and task-specific patterns.

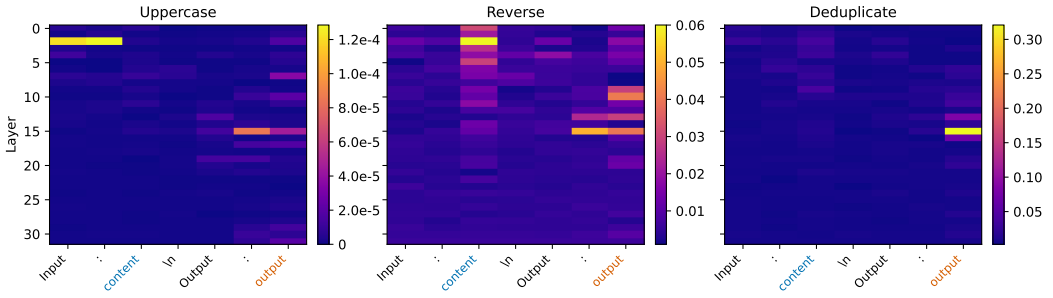

Figure 2: Per-site importance for three tasks on Llama3.1-8B. Brighter cells indicate higher importance. Token labels are color-coded by position type.

The importance maps show that the ICL knowledge captured by TO is often concentrated in a small subset of sites, with many sites having near-zero importance. However, the specific patterns vary by task, reflecting the different sparse circuits required for each. For the simpler Uppercase task, importance is highly localized, with most concentrated in template tokens within certain layers. In contrast, the more complex Reverse task displays a more distributed pattern, with importance spread across both template and content tokens, and some remaining at output positions. The Deduplicate task shows a focused pattern on output positions, especially in the middle layers.

These patterns help explain why baselines work reasonably well on Uppercase but fail on Reverse and Deduplicate. They mainly capture template-based ICL signals and miss the critical knowledge distributed across content and output positions in more complex tasks. Our ablation studies in Section 4.5 further validate these task-specific importance patterns, showing that knowledge in the identified areas contributes most to the improved performance over ZSL.

4.4 Scaling Analysis

The extraction of ICL knowledge with TO involves two key hyperparameters: the number of demonstrations K used in the prompt and the number of validation prompts m averaged to estimate the per-site parameters. We analyze how varying each of these parameters affects the performance of TO on Llama3.1-8B across the three algorithmic tasks, comparing to the strong baseline, TV.

Table 4 reports accuracy as m increases with K fixed at 8. Overall, the performance of both TV and TO is largely recovered at m=8 and tends to converge as m increases, while TO consistently outperforms TV at every m on every task. Interestingly, increasing m does not always increase the accuracy of TO, but it consistently pushes it toward the performance of standard ICL. For example, on Deduplicate, where TO performance at m=8 is lower than ICL, increasing m to 32 significantly improves TO and narrows the gap to ICL. However, on the Reverse task, where TO performance at m=8 is better than ICL, increasing m to 32 actually leads to a slight decrease in performance,

making it closer to the ICL result. These results confirm the faithfulness of TO in replicating ICL behavior, and suggest that while a small number of validation prompts is sufficient to capture the ICL signal, increasing m better aligns the replayed behavior with the original ICL performance.

Table 4: Effect of the number of validation prompts m on test accuracy (Llama3.1-8B, K=8 fixed). ICL (K=8) is shown as a reference baseline.

Task	**Method**	$m = 8$	$m = 16$	$m = 24$	$m = 32$	ICL
Uppercase	TV	87.50	88.28	87.50	88.28	100.00
	TO	100.00	100.00	100.00	100.00	
Reverse	TV	5.47	6.25	6.25	6.25	48.44
	TO	56.25	54.69	51.56	51.56	
Deduplicate	TV	2.34	2.34	2.34	2.34	37.50
	TO	19.53	26.56	28.91	28.12	

Figure 3 shows the accuracy of TV, TO, and standard ICL as K increases with m fixed at 32. As expected, ICL performance generally improves with more demonstrations, though gains diminish and eventually plateau. TV shows only slight improvements as K increases on Uppercase and Reverse, and remains far behind ICL. For TO, we observe an increase-then-decrease pattern on Reverse and Deduplicate as K grows. We attribute this to $w_t^{(\ell,h)}$ in Equation (2) approaching zero for large K, which may lead to more sensitive and noisier replay behavior.

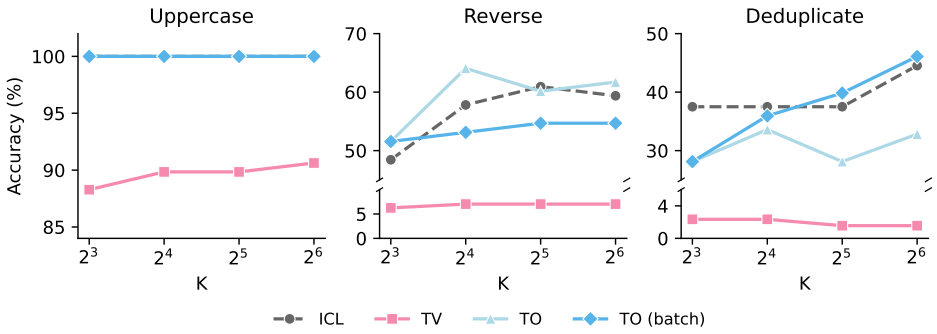

Figure 3: Effect of the number of demonstrations K on test accuracy (Llama3.1-8B, m=32). TO (batch) averages operators from disjoint 8-shot batches.

To address this and enable task operator extraction for many-shot learning, we break the large demonstration set into smaller batches (each with eight examples) and average the extracted operators across batches. This approach allows us to scale K effectively without running into long-context bottlenecks, and produces more stable parameter estimates as the demonstration size is consistent across batches. We denote this batch-averaged version as TO (batch). As shown in Figure 3, TO (batch) shows clear non-decreasing performance with larger K on all tasks, and may even outperform standard ICL in some settings (e.g., K=64 on Deduplicate). An alternative remedy that normalizes the context attention mass is discussed in Appendix H.3.

4.5 Ablation Studies

To understand the source of TO's superior performance over baselines, we conduct two ablation studies. Specifically, we ask whether TO's advantage comes from its ability to capture the ICL signal at specific positions missed by prior methods, or from its affine intervention structure that encodes dynamic, input-dependent knowledge beyond what fixed vectors can represent.

First, we assess the importance of different position types (template, content, and output) for capturing the ICL signal. We compare the template-only setting (used by baselines) to versions that also include content and output positions. This allows us to determine whether the ICL signal is primarily carried by the fixed instruction text, or also depends on sample-specific input and output tokens.

Second, we isolate the contribution of the multiplicative scaling component in the per-head operator by comparing the full affine operator to versions that use only the scaling or only the bias component. For a fair comparison, we re-optimize the parameters of the reduced operators in each condition. In the bias-only condition, we set $w_t^{(\ell,h)} = 1$ at every site and refit $\mathbf{b}_t^{(\ell,h)}$ as the mean difference between the full and context-masked head outputs, $\mathbf{u}_t^{(\ell,h)} - \mathbf{u}_t'^{(\ell,h)}$, across extraction samples. In the scaling-only condition, we set $\mathbf{b}_t^{(\ell,h)} = \mathbf{0}$ and refit $w_t^{(\ell,h)}$ via least-squares regression of $\mathbf{u}_t^{(\ell,h)}$ onto $\mathbf{u}_t'^{(\ell,h)}$. We use the template-only setting for analyzing the affine components, since template positions contain the fixed instruction text (identical across all test queries), making it easier to isolate the effect of the multiplicative component. In contrast, content and output positions carry sample-specific information, which naturally favors the affine operator for dynamic model steering.

Table 5 presents the results of both ablations on Qwen3-4B, which most closely matches ICL performance among the tested models. The upper portion of the table shows the position-type ablation results, while the lower portion focuses on the affine component ablation at template positions. As shown, the full all-positions operator achieves the best overall performance across tasks. However, the importance of different position types varies substantially by task. For example, the simple Translation, Linguistic, and Uppercase tasks rely almost entirely on template positions, while Reverse, Deduplicate, and the reasoning tasks require significant contributions from content and output positions. Among the tasks where content and output positions matter, their relative importance also differs: Reverse relies more on content positions, while Deduplicate suffers a larger performance drop when output positions are removed. This suggests diverse ICL mechanisms across tasks and aligns with the importance patterns observed in Figure 2.

Table 5: Position-type and affine component ablations on Qwen3-4B. Top rows vary position scope. Bottom rows isolate the scaling (w) and bias ($\mathbf{b}$) components at template positions.

Setting	Lexical		Algorithmic			Reasoning		
	Translation	Linguistic	Uppercase	Reverse	Deduplicate	GSM8K	MATH500	GPQA
ICL	69.53	73.44	100.00	43.75	24.22	89.39	55.80	37.88
Template + Content + Output	**67.19**	71.09	**100.00**	**39.06**	**17.19**	**86.20**	51.20	**33.33**
Template + Output	**67.19**	71.09	**100.00**	8.59	10.94	85.44	**53.60**	29.29
Template + Content	66.41	70.31	**100.00**	17.97	2.34	77.26	50.00	23.74
Template (w + $\mathbf{b}$)	**67.19**	**71.88**	**100.00**	7.81	9.38	79.61	47.20	22.22
Template (w only)	0.00	0.00	0.00	0.00	0.00	79.83	31.40	17.68
Template ($\mathbf{b}$ only)	0.00	0.00	0.00	0.00	0.00	34.87	10.20	19.19

From the lower portion of Table 5, we see that the multiplicative scaling component is essential for capturing the ICL signal at template positions, while the additive bias alone is insufficient. This pattern holds across all tasks, and is especially pronounced on non-reasoning tasks, where both scaling-only and bias-only versions collapse to zero accuracy. These results indicate that the two components are individually insufficient but jointly effective, consistent with the affine structure analytically observed in the model's attention computations.

5 Conclusion

We presented Task Operator (TO), a training-free method that captures in-context learning (ICL) knowledge as affine transformations of attention head outputs and replays them during zero-shot inference through dynamic updates to the output projection. Experiments across four language models and eight tasks demonstrate that TO achieves the best overall performance among prior methods, substantially closing the gap between zero-shot and few-shot performance. Our causal analysis reveals that the sites critical for ICL are sparse and task-specific, explaining why prior methods fall short on tasks requiring complex input-output interactions. Scaling experiments further show that averaging operators from disjoint demonstration batches enables effective many-shot learning without expanding the context window. These results connect in-context learning to an input-dependent reparameterization of the attention output projection, offering both a practical tool for efficient deployment and a lens for understanding how demonstrations reshape transformer computation.

Acknowledgments and Disclosure of Funding

This work is supported in part by the US National Science Foundation (NSF) and the National Institute of Health (NIH) under grants 2616033, 2538206, 2529378, 2217071, 2213700, R01LM014012-01A1, and 1R01HL184168-01. Any recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NIH or NSF.

References

[1] Steven Abreu and Joris Postmus. From steering vectors to conceptors and beyond: Compositional affine steering mechanisms for LLMs, 2024. URL https://openreview.net/forum?id=9wjGUN65tY.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[3] Wang Cai, Hsiu-Yuan Huang, Zhixiang Wang, and Yunfang Wu. Beyond demonstrations: Dynamic vector construction from latent representations. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5842–5857, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.297. URL https://aclanthology.org/2025.emnlp-main.297/.

[4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

[5] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584836. doi: 10.1145/1571941.1572114. URL https://doi.org/10.1145/1571941.1572114.

[6] Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.247. URL https://aclanthology.org/2023.findings-acl.247/.

[7] 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.

[8] 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*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

[9] Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.751. URL https://aclanthology.org/2023.emnlp-main.751/.

[10] Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching, 2023. URL https://arxiv.org/abs/2304.05969.

[11] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester

Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

[12] Seungwook Han, Jinyeop Song, Jeff Gore, and Pulkit Agrawal. Emergence and effectiveness of task vectors in in-context learning: An encoder decoder perspective. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 21871–21897. PMLR, 13–19 Jul 2025. URL `https://proceedings.mlr.press/v267/han25h.html`.

[13] Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.624. URL `https://aclanthology.org/2023.findings-emnlp.624/`.

[14] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

[15] Evan Hernandez, Arnab Sen Sharma, Tal Haklay, Kevin Meng, Martin Wattenberg, Jacob Andreas, Yonatan Belinkov, and David Bau. Linearity of relation decoding in transformer language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=w7LU2s14kE.

[16] Brandon Huang, Chancharik Mitra, Assaf Arbelle, Leonid Karlinsky, Trevor Darrell, and Roei Herzig. Multimodal task vectors enable many-shot multimodal in-context learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 22124–22153. Curran Associates, Inc., 2024. doi: 10.52202/079017-0697. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/27571b74d6cd650b8eb6cf1837953ae8-Paper-Conference.pdf.

[17] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

[18] Joonseong Kang, Soojeong Lee, Subeen Park, Sumin Park, Taero Kim, Jihee Kim, Ryunyi LEE, and Kyungwoo Song. Adaptive task vectors for large language models. In *Mechanistic Interpretability Workshop at NeurIPS 2025*, 2025. URL https://openreview.net/forum?id=HaAW356Htr.

[19] Dongfang Li, Zhenyu Liu, Xinshuo Hu, Zetian Sun, Baotian Hu, and Min Zhang. In-context learning state vector with inner and momentum optimization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 7797–7820. Curran Associates, Inc., 2024. doi: 10.52202/079017-0250. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/0ed52d7f6f641f228405d48a611e0684-Paper-Conference.pdf.

[20] Jiaqian Li, Yanshu Li, Ligong Han, Ruixiang Tang, and Wenya Wang. Train once, reuse everywhere: Generalizable implicit in-context learning by routing attention. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Gb2mqvRRa0.

[21] 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*, 2025. URL https://openreview.net/forum?id=G7u4ue6ncT.

[22] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

[23] 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 Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32287–32307. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24bx.html.

[24] Thomas Marshall, Adam Scherlis, and Nora Belrose. Refusal in llms is an affine function, 2025. URL https://arxiv.org/abs/2411.09003.

[25] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/6f1d43d5a82a37e89b0665b33bf3a182-Paper-Conference.pdf.

[26] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[27] Keqin Peng, Liang Ding, Yancheng Yuan, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. Revisiting demonstration selection strategies in in-context learning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9090–9101, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.492. URL https://aclanthology.org/2024.acl-long.492/.

[28] Yingzhe Peng, Chenduo Hao, Xinting Hu, Jiawei Peng, Xin Geng, and Xu Yang. Live: Learnable in-context vector for visual question answering. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 9773–9800. Curran Associates, Inc., 2024. doi: 10.52202/079017-0312. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/12d3e63be5574088f7c1bbc9162060bf-Paper-Conference.pdf.

[29] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In D. Song, M. Carbin, and T. Chen, editors, *Proceedings of Machine Learning and Systems*, volume 5, pages 606–624. Curran, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/file/c4be71ab8d24cdfb45e3d06dbfca2780-Paper-mlsys2023.pdf.

[30] Joris Postmus and Steven Abreu. Steering large language models using conceptors: Improving addition-based activation engineering. In *MINT: Foundation Model Interventions*, 2024. URL https://openreview.net/forum?id=gyAnAq16HC.

[31] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

[32] Baturay Saglam, Xinyang Hu, Zhuoran Yang, Dionysis Kalogerias, and Amin Karbasi. Learning task representations from in-context learning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6634–6663, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.345. URL https://aclanthology.org/2025.findings-acl.345/.

[33] Suzanna Sia, David Mueller, and Kevin Duh. Where does in-context learning happen in large language models? In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 32761–32786. Curran Associates, Inc., 2024. doi: 10.52202/079017-1030. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3979818cdc7bc8dbeec87170c11ee340-Paper-Conference.pdf.

[34] Shashwat Singh, Shauli Ravfogel, Jonathan Herzig, Roee Aharoni, Ryan Cotterell, and Ponnurangam Kumaraguru. Representation surgery: Theory and practice of affine steering. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 45663–45680. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/singh24d.html.

[35] Niklas Stoehr, Kevin Du, Vésteinn Snæbjarnarson, Robert West, Ryan Cotterell, and Aaron Schein. Activation scaling for steering and interpreting language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational*

Linguistics: EMNLP 2024, pages 8189–8200, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.479. URL https://aclanthology.org/2024.findings-emnlp.479/.

[36] 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.

[37] Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.

[38] Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=NpsVSN6o4ul.

[39] 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.

[40] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[41] Kayo Yin and Jacob Steinhardt. Which attention heads matter for in-context learning? In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 72428–72461. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/yin25e.html.

[42] Yiming Zhang, Shi Feng, and Chenhao Tan. Active example selection for in-context learning. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.622. URL https://aclanthology.org/2022.emnlp-main.622/.

A Limitations and Broader Impacts

Limitations. Task Operator (TO) provides a training-free way to extract and replay the effect of in-context demonstrations, while also opening several directions for further study. First, as shown in our scaling experiments, the performance of TO can become less stable when the number of demonstrations K is very large. This may occur because the attention mass assigned to non-context positions becomes small, which can increase the variance of the extracted parameters during replay. In practice, we find that averaging operators extracted from multiple independently sampled demonstration batches mitigates this issue and enables effective many-shot scaling. Second, our empirical evaluation focuses on text-only decoder language models across lexical, algorithmic, and reasoning tasks. These tasks cover a diverse set of input-output structures, but they do not exhaust the range of settings in which in-context learning is used. Extending TO to settings such as multimodal reasoning is a natural and promising direction for future work.

Broader Impacts. This work contributes to the mechanistic understanding of in-context learning by showing how the effect of demonstrations can be extracted from forward passes and replayed through TO. This perspective may help researchers analyze how transformer models encode task knowledge and why different tasks rely on different sparse circuits. TO may also make in-context learning more efficient in deployment by allowing demonstration-induced behavior to be reused without repeatedly processing the full demonstration context. This can reduce context-window usage and inference cost in settings where the same task is queried many times. As with other methods for adapting model behavior, extracted operators should be applied with appropriate validation and care, especially when demonstrations may contain sensitive, biased, or otherwise undesirable patterns.

B Dataset Details

We evaluate on eight tasks spanning three categories. All non-reasoning tasks use the template pair "`Input: {input}\nOutput: {output}`" for demonstrations and "`Input: {input}\nOutput:`" for queries. Reasoning tasks append "`Let's think step by step.`" to the output prefix, eliciting chain-of-thought (CoT) reasoning. For lexical and algorithmic tasks, 128 records form the held-out test set. Reasoning tasks use the original benchmark test splits.

B.1 Lexical Tasks

Both tasks use word-pair data released by Dong et al. [7].

Translation. English-to-French single-word translation. The pool of 739 word pairs is allocated into 579 demonstration, 32 validation, and 128 test records.

```
Input: subject
Output: matière

Input: look
Output: voir

Input: wait
Output:
```

Expected output: `attendre`

Linguistic. English singular-to-plural inflection, covering regular and irregular forms. The pool of 173 word pairs is split into 13 demonstration, 32 validation, and 128 test records.

```
Input: bacterium
Output: bacteria

Input: seminoma
Output: seminomata

Input: foot
```

```
Output:
```

Expected output: `feet`

B.2 Algorithmic Tasks

All three tasks are generated synthetically over an eight-letter alphabet $\{a, \ldots, h\}$. Each input is a random sequence of 5–12 space-separated tokens generated with a fixed seed. Each split contains 1024 demonstration, 256 validation, and 128 test records.

Uppercase. Map each token in the sequence to its uppercase form.

```
Input: e d g a g e d b
Output: E D G A G E D B

Input: b h a b a a g a c a g d
Output: B H A B A A G A C A G D

Input: d d f b b c b b
Output:
```

Expected output: `D D F B B C B B`

Reverse. Output the token sequence in reverse order.

```
Input: d b f g d e
Output: e d g f b d

Input: g c h b d a
Output: a d b h c g

Input: b d a h f g d b h
Output:
```

Expected output: `h b d g f h a d b`

Deduplicate. Remove duplicate tokens, retaining only the first occurrence of each in order.

```
Input: b h d e h
Output: b h d e

Input: f b f b d e
Output: f b d e

Input: g e h d h h
Output:
```

Expected output: `g e h d`

B.3 Reasoning Tasks

All three tasks use chain-of-thought (CoT) formatting. The test split is the original benchmark's held-out set. Demonstrations and validation records are sampled from the corresponding training split.

GSM8K [4]. Grade-school arithmetic word problems with multi-step reasoning. Intermediate computations are annotated as `«expr=result»`. The 1024 demonstration and 256 validation records are sampled from the GSM8K training split. The 1319 test records form the full official test set. Targets end with `The answer is N`.

```
Input: Elroy decides to enter a walk-a-thon [...] Last year, walkers
    earned
```

```
$4 a mile; this year $2.75. If last year's winner collected $44, how
    many
more miles will Elroy walk to collect the same amount?
Output: Let's think step by step. Last year's winner walked 11 miles
    because
44/4=<<44/4=11>>11. Elroy walks 16 miles because
   44/2.75=<<44/2.75=16>>16.
Elroy walks 5 more miles because 16-11=<<16-11=5>>5. The answer is 5

Input: Sandy's monthly phone bill equals ten times her age. In two
   years she
will be three times Kim's age; Kim is 10. What is Sandy's phone bill
   ?
Output: Let's think step by step. [...] The answer is 340

Input: Janet's ducks lay 16 eggs/day. She eats 3 for breakfast,
   bakes with 4,
and sells the rest at $2 each. How much does she make daily?
Output: Let's think step by step.
```

Expected output: `Janet sells 16-3-4=«16-3-4=9»9 eggs. She makes 9*2=$«9*2=18»18/day. The answer is 18`

MATH-500 [22]. Competition-level mathematics problems (AMC/AIME style) from the Hendrycks MATH benchmark [14]. The 1024 demonstration and 256 validation records span seven subject areas from the Hendrycks MATH training split. The 500 test records form the MATH-500 evaluation set. Targets end with `The answer is X` where X is the extracted boxed answer.

```
Input: What is the smallest positive integer $x$ such that $400x$ is
    a
multiple of $576$?
Output: Let's think step by step. $400=2^4*5^2$, $576=2^6*3^2$. x
   must
contain $2^2*3^2=36$ as a factor. [...] The answer is 36

Input: Convert $3206_7$ to base 10.
Output: Let's think step by step. $3*7^3+2*7^2+0+6=1133$.
The answer is 1133

Input: Convert $(0,3)$ from rectangular to polar coordinates
(with $r>0$, $0\le\theta<2\pi$).
Output: Let's think step by step.
```

Expected output: $r = 3$, $\theta = \pi/2$. `[...] The answer is` $(3, \pi/2)$

GPQA-Diamond [31]. Graduate-level multiple-choice questions in physics, chemistry, and biology. The 198 test records are the GPQA-Diamond subset. The 218 demonstration and 32 validation records are taken from GPQA-Main after removing Diamond questions. Answer choices are shuffled per record with a fixed seed. Targets end with `The answer is (LETTER)`.

```
Input: methyllithium is added to dichlorotitanocene [...] What are
   the
multiplicities of the two most deshielded H nuclei in product 3?
(A) doublet, multiplet
(B) singlet, multiplet
(C) doublet, doublet
(D) doublet of doublets, doublet of doublets
Output: Let's think step by step. [...] The answer is (C)

Input: A transiting planet candidate is detected by TESS [...] Which
method CANNOT confirm its presence?
(A) Periodic FWHM changes in spectral lines
(B) Detection of a Rossiter-McLaughlin effect
(C) A signal in reflectance spectroscopy
```

```
(D) Periodic wavelength changes of spectral lines
Output: Let's think step by step. [...] The answer is (A)

Input: Two quantum states with energies E1 and E2 have a lifetime of
10^-9 sec and 10^-8 sec. Which energy difference allows clearly
resolving their spectral lines?
(A) 10^-4 eV
(B) 10^-11 eV
(C) 10^-9 eV
(D) 10^-8 eV
Output: Let's think step by step.
```

Expected output: `[...] The energy difference must be » 10^-7 eV. The answer is (A)`

C Implementation Details

Models. All experiments use instruction-tuned model variants. Concretely, we use `Qwen/Qwen3-4B` and `Qwen/Qwen3-8B` from the Qwen3 family, and `meta-llama/Llama-3.2-3B-Instruct` and `meta-llama/Llama-3.1-8B-Instruct` from the Llama family, following their HuggingFace identifiers.

Task Operator extraction. The operator is extracted using $K{=}8$ demonstrations and $m{=}32$ answer-excluded validation prompts. In particular, we only keep validation records where ICL generation ends without hitting the token budget. Each of the T structural template token positions in the query receives an independent per-head $(w, \mathbf{b})$ pair, estimated by averaging across the validation prompts. For content and output positions, the per-head parameters are first averaged over all token positions within that category in each prompt, then averaged across prompts, resulting in a single operator per (layer, head) that is applied uniformly to every content token and every generated token during zero-shot replay. The all-positions, all-layers variant patches every decoder layer with no site pruning.

Baseline hyperparameters. All baselines use the same $K{=}8$ demonstrations and prompt templates as TO. For TV [13], FV [36], and Conceptor [30], the injection layer is selected by a full scan (step size 1) over all decoder layers, choosing the layer that minimizes NLL on the model's own ICL completions on the validation queries. FV additionally identifies the top-20 attention heads globally across all layers by average indirect effect (AIE), computed as the reduction in zero-shot NLL when each head's pre-projection input is replaced with its mean ICL activation. ICV [23] injects the context vector at every layer simultaneously via a norm-preserving residual addition with scaling coefficient $\lambda{=}0.1$. Conceptor uses aperture $\alpha{=}1.0$ and output scaling $\beta{=}1.0$.

Generation settings. All methods use a temperature of 0.0 for deterministic decoding. For lexical and algorithmic tasks, generation stops at the first newline with a token budget of 64. For reasoning tasks, generation stops at the substring `"\nInput:"` (the ICL demo boundary) with a budget of 512 tokens. The final answer is extracted as the text following the first occurrence of "`The answer is`" in the model's output. The experiments are run on NVIDIA A100 GPUs with 80GB memory.

D Stability of Extracted Parameters

The methodology aggregates per-prompt extractions in two ways. First, at each fixed token position, we average across the m ICL prompts (cross-sample). Second, for tasks with variable-length content or output, the operator is applied at token positions not seen during extraction (cross-token within a category). Each aggregation step relies on a distinct empirical stability claim, which we verify directly below. We additionally include a robustness check on cross-demonstration-set stability.

D.1 Cross-sample stability

For each (layer, head, token j), let $w_{j,i}^{(\ell,h)}$ denote the per-head scaling factor extracted from the i-th of m=32 ICL prompts. Cross-sample stability is the claim that $\{w_{j,i}^{(\ell,h)}\}_{i=1}^{m}$ varies little across prompts at each fixed token position. We measure this with the coefficient of variation $\mathrm{CV}\big(w_j^{(\ell,h)}\big) = \mathrm{std}_i\big(w_{j,i}^{(\ell,h)}\big)/\big|\mathrm{mean}_i\big(w_{j,i}^{(\ell,h)}\big)\big|$, computed independently at each token. For the bias vector $\mathbf{b}_j^{(\ell,h)}$, we report the average pairwise cosine similarity across the m prompts at each token position.

We extract per-prompt operators using all position types and all decoder layers (the same all-positions, all-layers configuration as in the main experiments) and run m=32 ICL prompts per (model, task) cell. We test the four LLMs on one representative task from each category: Translation for lexical, Reverse for algorithmic, and GSM8K for reasoning. At each (layer, token, head) site, we record one $\mathrm{CV}(w)$ value across the m prompts and one mean pairwise $\cos(\mathbf{b})$ value across the $\binom{m}{2}$ off-diagonal pairs of bias vectors.

Table 6 reports, per (model, task) cell, the fraction of sites whose per-prompt $\mathrm{CV}(w)$ falls below $\{0.1, 0.2, 0.5\}$ and whose mean pairwise $\cos(\mathbf{b})$ exceeds $\{0.9, 0.8, 0.5\}$. Averaged across the twelve cells, 80.7% of sites have $\mathrm{CV}(w)$<0.1 and 92.1% have $\cos(\mathbf{b})$>0.9. The Llama models are noticeably tighter than the Qwen models on both metrics, with the fraction at $\mathrm{CV}(w)$<0.1 shifting from 70–77% on the Qwen models to 85–92% on the Llama models. Within each model, Reverse is the tightest task and GSM8K consistently exhibits the lowest fraction of sites at $\cos(\mathbf{b})$>0.9. The per-prompt scaling factors and bias directions cluster tightly at each fixed token position, supporting cross-sample averaging as the basis for the broadcast operator and the cross-sample stability claim invoked in Section 3.1.

Table 6: Cross-sample stability across ICL prompts. Each row reports the fraction of (layer, token, head) sites (in %) whose per-prompt scaling factor satisfies $\mathrm{CV}(w)$ below the threshold and whose per-prompt bias satisfies mean pairwise $\cos(\mathbf{b})$ above the threshold.

Model	Task	$\mathrm{CV}(w)<$			$\cos(\mathbf{b})>$		
		0.1	0.2	0.5	0.9	0.8	0.5
Qwen3-4B	Translation	70.4	83.9	95.4	88.1	98.0	100.0
	Reverse	77.3	90.0	97.1	97.7	99.6	100.0
	GSM8K	71.4	85.1	94.3	83.5	92.7	99.2
Qwen3-8B	Translation	71.9	84.9	96.0	86.2	97.4	100.0
	Reverse	76.6	89.5	96.3	97.8	99.7	100.0
	GSM8K	73.1	85.9	94.7	83.9	93.2	99.3
Llama3.2-3B	Translation	89.3	96.7	99.4	95.5	99.3	100.0
	Reverse	92.0	97.7	99.7	98.7	99.9	100.0
	GSM8K	87.0	95.6	99.2	93.1	97.9	100.0
Llama3.1-8B	Translation	85.3	94.7	99.2	93.1	99.0	100.0
	Reverse	88.8	96.0	99.3	97.9	99.8	100.0
	GSM8K	85.3	94.7	99.3	89.8	96.0	99.8

We further examine how stability and replay accuracy evolve with the number of extraction samples m on Llama3.1-8B Deduplicate (Table 7). The extracted parameters are already highly stable at m=8, and both stability metrics remain nearly unchanged as m increases, while accuracy improves up to m=24 and then plateaus, consistent with Table 4. This suggests that additional extraction samples mainly refine the estimates at a small subset of sites that are critical for performance.

Table 7: Cross-sample stability and TO accuracy as the number of extraction samples m increases (Llama3.1-8B, Deduplicate).

	m=8	m=16	m=24	m=32
Sites with $\mathrm{CV}(w)$<0.1 (%)	88.9	89.7	89.3	89.1
Sites with $\cos(\mathbf{b})$>0.9 (%)	98.4	99.1	99.0	99.0
TO accuracy (%)	19.5	26.6	28.9	28.1

D.2 Cross-token stability within a category

For content and output categories, our method averages the extracted $(w, \mathbf{b})$ across all token offsets within each category, producing a single broadcast operator applied uniformly regardless of position. We assess whether this averaging discards useful offset-specific structure by comparing the broadcast variant against a per-rank variant that stores independent $(w, \mathbf{b})$ for each of the first 12 token offsets in the content and output categories. We use 12 offsets because this provides complete coverage for Translation (single-word content and output) and Reverse (sequences of 5–12 tokens), making the comparison maximally informative on these tasks: any offset-specific structure that exists will be captured by the per-rank variant. For GSM8K, where content and output sequences span dozens to hundreds of tokens, exhaustive per-rank coverage is infeasible. In this case, 12 offsets covers a prefix.

Table 8 reports the accuracy of each variant on n=128 test records per (model, task) cell, using the same four models and three tasks as in Section D.1. On Translation, the two variants match exactly ($\Delta = 0$) for all models, confirming that no exploitable offset-specific structure exists when content spans are short. On Reverse, where per-rank also has complete offset coverage, broadcast still outperforms by 2–10 pp. This shows that the averaged operator is more robust than offset-specific parameters even with full positional coverage, likely because averaging pools signal across the m validation prompts at each (layer, head) site, whereas per-rank must estimate $12\times$ as many parameters from the same data, adding additional noises. On GSM8K, broadcast matches or outperforms per-rank in three of four models (gaps of -5 to -28 pp), with a single marginal exception (Llama3.1-8B, $+2.3$ pp). Here the long sequences make exhaustive per-rank coverage infeasible, but broadcast remains effective by capturing the shared signal across all offsets without positional alignment. Across all settings, the broadcast variant matches or exceeds per-rank, confirming that averaging within a category is not only sufficient but preferable to offset-specific modeling.

Table 8: Cross-token stability via downstream accuracy. The per-rank variant uses offset-aware $(w, \mathbf{b})$ for the first 12 offsets in content and output; broadcast averages across offsets to obtain one $(w, \mathbf{b})$ per category. Δ: per-rank minus broadcast accuracy.

Model	Task	Broadcast (%)	Per-rank (%)	Δ (pp)
Qwen3-4B	Translation	67.2	67.2	0.0
	Reverse	39.1	36.7	-2.3
	GSM8K	86.2	81.2	-5.0
Qwen3-8B	Translation	67.2	67.2	0.0
	Reverse	60.2	50.0	-10.2
	GSM8K	78.6	50.8	-27.8
Llama3.2-3B	Translation	68.0	68.0	0.0
	Reverse	31.2	21.1	-10.2
	GSM8K	67.8	60.9	-6.8
Llama3.1-8B	Translation	71.9	71.9	0.0
	Reverse	51.6	47.7	-3.9
	GSM8K	75.8	78.1	$+2.3$

D.3 Cross-demonstration-set stability

Beyond the two main premises, we check whether the extracted operator is genuinely task-specific rather than demonstration-set-specific. For a given task, we extract two operators $(w, \mathbf{b})^{(1)}$ and $(w, \mathbf{b})^{(2)}$ from disjoint pools of m prompts using different demonstration sets, and compare their per-slot agreement.

We instantiate the two operators using non-overlapping demonstration pools sampled from the same task. Operator $(w, \mathbf{b})^{(1)}$ uses the first eight demonstrations and the first 32 validation queries. Operator $(w, \mathbf{b})^{(2)}$ uses the next eight demonstrations (indices 8 through 15) with the same validation queries. The two pools share no demonstration prompts, so any agreement between the two operators at a fixed (layer, token, head) site reflects the task signal rather than the identity of the eight demonstrations.

We compute two per-slot agreement metrics. Pearson correlation $\rho(w^{(1)}, w^{(2)})$ is taken over the flattened vector of per-slot scaling factors across all (layer, token, head) sites in a cell, resulting in one scalar per (model, task). Cosine similarity $\cos(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})$ is computed at each site separately, and we report its mean and median across sites within each cell.

Table 9 reports the agreement statistics across the same four models and three tasks. Pearson correlations of w exceed 0.96 in every cell, with a mean of 0.984 across the twelve cells, indicating that the per-slot scaling pattern is essentially fixed by the task. Mean cosine similarity of $\mathbf{b}$ falls in the range 0.81–0.95, with Reverse showing the highest agreement and Translation the lowest. Together, these statistics support the claim that the extracted operator captures the task-specific signal rather than the identity of the demonstrations used to extract it.

Table 9: Cross-demonstration-set stability. Pearson(w) is computed once per (model, task) cell over all flattened (layer, token, head) sites. Cosine similarity of $\mathbf{b}$ is computed per site, with mean and median aggregated across sites within each cell.

Model	Task	Pearson(w)	Mean $\cos(\mathbf{b})$	Median $\cos(\mathbf{b})$
Qwen3-4B	Translation	0.981	0.844	0.896
	Reverse	0.992	0.953	0.967
	GSM8K	0.993	0.912	0.947
Qwen3-8B	Translation	0.982	0.835	0.891
	Reverse	0.992	0.954	0.968
	GSM8K	0.993	0.908	0.945
Llama3.2-3B	Translation	0.974	0.813	0.864
	Reverse	0.987	0.941	0.962
	GSM8K	0.980	0.899	0.933
Llama3.1-8B	Translation	0.964	0.839	0.892
	Reverse	0.982	0.950	0.963
	GSM8K	0.991	0.911	0.933

E Task Operator Performance with Sparsified Circuits

We next evaluate whether the sparse circuits identified by the validation-NLL criterion are sufficient for task-operator replay. Starting from the dense task operator, we rank intervention sites using the site-importance procedure in Section 3.3 and retain only the top-p fraction of sites. Figure 4 reports the resulting test accuracy as p is swept from 1% to 100% on the three algorithmic tasks across all four models.

The sweep reveals two main trends. First, retaining only the top 1% of sites is generally insufficient: performance collapses to near-zero on almost all model–task pairs. Thus, although the ICL signal is sparse, successful replay does not reduce to a single dominant site. Instead, it requires a small but coordinated set of interventions. Second, moderate sparsification preserves most of the dense operator's performance. In particular, retaining roughly 20% of sites already recovers nearly the full dense-operator accuracy on Uppercase and Reverse, indicating that much of the useful ICL signal is concentrated in a small subset of layer-token positions. On Deduplicate, sparsification can be even more beneficial. For Qwen3-8B, accuracy increases from 65.6% with the dense operator to 82.8% at $p = 20\%$, suggesting that low-importance sites can introduce interference rather than useful task information.

To choose a sparsification level without using test labels, we select $p^\star$ separately for each model–task pair by minimizing teacher-forced validation NLL over $p \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$, breaking ties in favor of smaller circuits. Site rankings are obtained by reciprocal-rank fusion over the per-sample NLL impacts. Table 10 compares the resulting sparse operator against the dense operator. The validation-selected sparse operator preserves or improves test accuracy in all 12 model–task cells while retaining, on average, 67% of sites. The largest gains appear on Reverse and Deduplicate, where pruning low-importance sites reduces interference and improves accuracy by up to 9.4 percentage points. These results support the interpretation that task operators are not only sparse in their causal structure, but can also be made more effective by removing sites that contribute little or noisy signal.

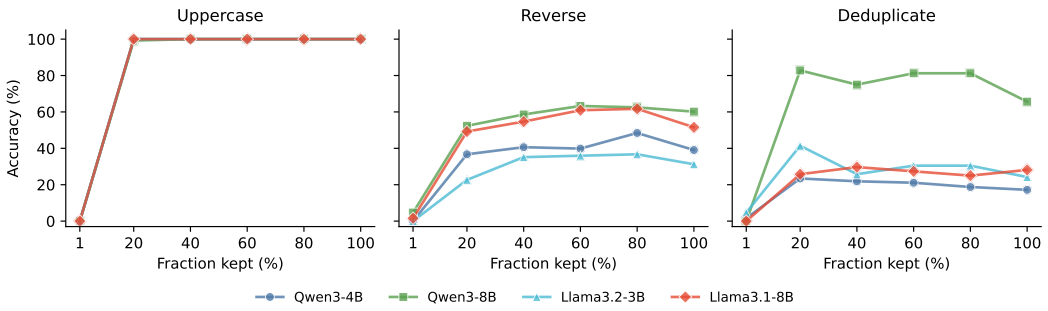

Figure 4: Test accuracy under top-p sparsification on three algorithmic tasks across four models.

However, selecting the optimal sparsification level requires sweeping multiple thresholds on the validation set, which introduces an additional hyperparameter selection step and may overfit to validation-specific noise. For example, Table 10 shows that the optimal $p^\star$ is selected as 100% for some models on the Uppercase task. However, Figure 4 reflects the sparse nature of the Uppercase task, indicating that the $p^\star$ might overfit the validation samples. For this reason, we use the full operator, corresponding to $p = 100\%$, in the main experiments. This choice avoids threshold tuning while still providing strong and stable performance, as Figure 4 shows that the dense operator remains competitive across tasks and models. We therefore view sparsification primarily as an analysis tool for identifying compact task circuits, rather than as a necessary component of the main TO method.

Table 10: Test accuracy (%) of the dense and val-NLL-sparsified task operator on the three algorithmic tasks. $p^\star$ is selected by minimizing validation NLL over $p \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. *Sites kept* is the fraction of sites retained at $p^\star$.

Model	Method	Uppercase	Reverse	Deduplicate
Qwen3 (4B)	ICL	100.00	43.75	24.22
	TO (Full)	100.00	39.06	17.19
	TO (Sparse @ $p^\star$)	**100.00**	**39.84**	**21.09**
	Sites kept (%)	100	60	60
Qwen3 (8B)	ICL	100.00	67.19	76.56
	TO (Full)	100.00	60.16	65.62
	TO (Sparse @ $p^\star$)	**100.00**	**63.28**	**75.00**
	Sites kept (%)	100	60	40
Llama3.2 (3B)	ICL	100.00	21.88	33.59
	TO (Full)	100.00	31.25	24.22
	TO (Sparse @ $p^\star$)	**100.00**	**36.72**	**30.47**
	Sites kept (%)	60	80	60
Llama3.1 (8B)	ICL	100.00	48.44	37.50
	TO (Full)	100.00	51.56	28.12
	TO (Sparse @ $p^\star$)	**100.00**	**60.94**	**29.69**
	Sites kept (%)	80	60	40

F Robustness to Prompt Template Choice

The main experiments use the canonical `Input/Output` prompt format. Since ICL and activation-based replay methods can be sensitive to prompt wording, we further evaluate whether TO remains effective when the same task is expressed with different surface templates. We consider three templates. Template `t0` is the canonical format with `Input` and `Output` field names. Template `t1` uses the corresponding `Q` and `A` field names: "`Q: {input}\nA: {output}`". Template `t2` uses a compact arrow format: "`{input} -> {output}`". For each model, task, and template, TO and TV are extracted and evaluated under the same template.

Table 11 reports results on one representative task from each category, with Translation for lexical tasks, Deduplicate for algorithmic tasks, and GPQA for reasoning. Across these settings, TO remains the stronger intervention method in most cases and is substantially more stable than TV on tasks where the prompt format strongly affects the baseline. On Translation, TO stays close to ICL across all three templates, while TV is more sensitive to template choice. On Deduplicate, performance varies with the prompt format for both ICL-style prompting and replay methods, but TO consistently remains well above TV, indicating that the operator continues to capture useful task structure under matched template changes. On GPQA, where zero-shot performance is already competitive and the margins between non-ICL methods are smaller, TO remains broadly comparable across templates.

Overall, these results suggest that TO is not tied to a single prompt wording when extraction and evaluation use the same template. TO preserves its advantage over the task vector baseline on the lexical and algorithmic tasks and remains competitive on the reasoning task. This behavior is consistent with TO's design. Instead of relying on a single fixed residual vector, TO applies affine operators across layers, attention heads, and position types, making the replayed ICL effect less dependent on any one prompt token or extraction site.

Table 11: Test accuracy (%) under three matched prompt templates. `t0` denotes the canonical `Input/Output` format, `t1` the `Q/A` format, and `t2` the compact arrow format. TO and TV are extracted and evaluated under the same template. Bold marks the best score among ZSL, TV, and TO.

Model	Method	Translation			Deduplicate			GPQA		
		t_0	t_1	t_2	t_0	t_1	t_2	t_0	t_1	t_2
Qwen3 (4B)	ICL	69.53	67.97	67.19	24.22	14.84	18.75	37.88	29.80	29.29
	ZSL	0.00	0.00	0.00	0.00	0.00	0.00	17.68	16.16	17.68
	TV	43.75	41.41	11.72	0.00	0.78	0.78	21.21	**30.30**	23.23
	TO (Ours)	**67.19**	**64.84**	**62.50**	**17.19**	**15.62**	**9.38**	**33.33**	29.29	**28.79**
Qwen3 (8B)	ICL	68.75	67.97	65.62	76.56	47.66	35.16	34.85	32.83	34.85
	ZSL	0.00	0.00	0.00	0.00	0.00	0.00	21.21	23.23	18.69
	TV	49.22	32.81	4.69	9.38	1.56	0.00	22.73	23.23	28.79
	TO (Ours)	**67.19**	**67.19**	**62.50**	**65.62**	**36.72**	**7.81**	**33.33**	**32.83**	**29.29**
Llama3.2 (3B)	ICL	66.41	66.41	61.72	33.59	21.09	20.31	24.24	25.25	22.73
	ZSL	0.00	0.78	3.91	0.00	0.00	1.56	**26.26**	28.28	22.73
	TV	30.47	48.44	53.91	2.34	0.00	5.47	25.76	24.75	**28.28**
	TO (Ours)	**67.97**	**67.19**	**66.41**	**24.22**	**6.25**	**8.59**	22.73	**28.79**	26.77
Llama3.1 (8B)	ICL	71.09	67.97	65.62	37.50	35.94	28.91	26.26	24.75	23.74
	ZSL	1.56	3.12	3.91	0.00	0.00	2.34	**23.74**	**31.82**	19.19
	TV	53.12	66.41	64.06	2.34	1.56	6.25	20.71	22.73	**24.75**
	TO (Ours)	**71.88**	**69.53**	**67.97**	**28.12**	**28.91**	**18.75**	**23.74**	27.27	23.74

G Cross-Task Transfer of Extracted Knowledge

The stability analyses in Section D show that task operators are not artifacts of individual prompts or demonstration sets. We next ask whether the extracted knowledge can generalize across related reasoning tasks. We extract an operator from GSM8K using the same setting as the main experiments, with $K = 8$ demonstrations and $m = 32$ validation prompts, and replay it on MATH-500 and GPQA-Diamond. During transfer, the input prompt remains target specific, while only the source of the operator changes. We compare this setting with zero-shot inference, standard in-domain task operators, and full in-context learning on the target task.

Table 12 shows that the GSM8K operator transfers strongly to MATH-500. Across model families, its performance remains close to the in-domain operator, and in some cases it is competitive with or better than full in-context learning. This result suggests that TO captures a reusable chain-of-thought reasoning component rather than merely memorizing task-specific demonstrations.

Transfer to GPQA-Diamond is more sensitive to the target format, which is expected because GPQA differs from GSM8K and MATH-500 in both domain and answer style. Even in this harder setting,

the transferred operator remains competitive with zero-shot inference for most model and task combinations, and it improves over the in-domain operator for one Llama model. These results suggest that TO extracts a partially task-general reasoning signal, while also preserving the ability to reflect task-specific structure when the source and target tasks are more closely aligned.

Table 12: Cross-task transfer of task operators extracted from GSM8K to MATH-500 and GPQA-Diamond. The results compare transferred operators with zero-shot inference, in-domain task operators, and standard in-context learning. Underlined entries indicate methods that improve over the zero-shot baseline.

Model	Method	MATH-500	GPQA-Diamond
Qwen3 (4B)	ICL	55.80	37.88
	ZSL	23.60	17.68
	TO (in-domain)	<u>51.20</u>	<u>33.33</u>
	TO (transfer)	<u>51.20</u>	<u>21.72</u>
Qwen3 (8B)	ICL	61.20	34.85
	ZSL	19.00	21.21
	TO (in-domain)	<u>50.60</u>	<u>33.33</u>
	TO (transfer)	<u>41.80</u>	20.71
Llama3.2 (3B)	ICL	27.20	24.24
	ZSL	24.80	26.26
	TO (in-domain)	<u>25.20</u>	22.73
	TO (transfer)	24.20	<u>28.79</u>
Llama3.1 (8B)	ICL	28.80	26.26
	ZSL	27.60	23.74
	TO (in-domain)	25.40	23.74
	TO (transfer)	<u>31.60</u>	23.74

H Additional Results

We provide additional results on ICV with a tuned steering strength, the inference efficiency of TO, and a normalized variant of TO for large numbers of demonstrations.

H.1 ICV with Tuned Steering Strength

In the main experiments, ICV follows its official implementation, which applies steering at all layers with λ=0.1. To examine whether a task-specific steering strength improves ICV, we tune a single shared $\lambda \in \{0, 0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$ for each task by minimizing the validation NLL, the same criterion used to select hyperparameters for the other baselines. As shown in Table 13, the tuned ICV performs similarly to the default setting and remains close to ZSL.

Table 13: Accuracy (%) of ICV with the default steering strength (λ=0.1) and with λ tuned on the validation set.

Model	Task	ZSL	ICV (λ=0.1)	ICV (tuned λ)	TO (Ours)
Qwen3-4B	Translation	0.0	0.0	0.0	67.2
	Linguistic	0.0	0.0	0.0	71.1
Qwen3-8B	Translation	0.0	0.0	0.0	67.2
	Linguistic	0.0	1.6	0.0	75.0
Llama3.2-3B	Translation	0.0	0.0	0.8	68.0
	Linguistic	3.9	4.7	4.7	75.8
Llama3.1-8B	Translation	1.6	1.6	0.8	71.9
	Linguistic	10.9	10.2	7.8	81.2

H.2 Efficiency Analysis

We quantify the per-query inference cost on 64 GSM8K questions with Llama3.1-8B, whose demonstrations are long chain-of-thought exemplars, assuming 128 generated tokens per query. As shown in Table 14, TO matches the cost of zero-shot inference, since it only rescales and shifts the attention outputs. Compared with 8-shot ICL, it reduces the prompt length by more than $20\times$ and the key–value (KV) cache by more than $8\times$; caching the demonstrations avoids recomputing them for each query but still requires storing and attending to them. The one-time extraction cost with m=32 validation prompts and K=8 demonstrations is about $32 \times 29{,}350$ GFLOPs, which is recovered after about 36 test queries relative to 8-shot ICL without KV caching.

Table 14: Per-query inference cost on GSM8K with Llama3.1-8B, reported as mean $\pm$ standard deviation over 64 questions.

Setting	Prompt tokens	KV cache (MB)	GFLOPs
ZSL	69±20	25±3	3180±320
8-shot ICL	1602±20	216±3	29350±330
8-shot ICL + KV caching	1602±20	216±3	3340±320
TO (Ours)	69±20	25±3	3180±320

H.3 Normalized Task Operator

As discussed in Section 4.4, the scaling factor $w = Z'/(Z' + Z_{\mathcal{C}})$, where $Z_{\mathcal{C}} = \sum_{i \in \mathcal{C}} \exp(s_i)$ is the softmax mass on context positions, approaches zero for large K because $Z_{\mathcal{C}}$ grows with the number of demonstrations. As an alternative to batch averaging, we consider a normalized task operator (NTO) that rescales the context mass to a reference number of demonstrations K_0:

$$\tilde{w} = \frac{Z'}{Z' + (K_0/K)\, Z_{\mathcal{C}}}, \qquad \tilde{\mathbf{b}} = (1 - \tilde{w})\, \frac{\mathbf{b}}{1 - w},$$

which keeps the scaling factor comparable to that of a K_0-shot prompt while preserving the direction of the context contribution. Table 15 compares NTO (K_0=8) with TO (batch) on Llama3.1-8B. NTO further improves over TO (batch) on Deduplicate for all $K > 8$, whereas on Reverse it improves at K=16 but degrades for larger K. As NTO does not consistently outperform the simpler batch averaging, we use TO (batch) in the main experiments.

Table 15: Test accuracy (%) on Llama3.1-8B as the number of demonstrations K increases.

Task	Method	K=8	K=16	K=32	K=64
Deduplicate	TO (batch)	28.1	35.9	39.8	46.1
	NTO (K_0=8)	28.1	38.3	42.2	51.6
Reverse	TO (batch)	51.6	53.1	54.7	54.7
	NTO (K_0=8)	51.6	61.7	53.1	43.0

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix A

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: There is no theoretical result.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Appendices B and C

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The link is provided in the Abstract.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix C

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Section 4.2 reports the mean and standard deviation over random seeds for sampling the demonstrations, together with paired significance tests. Additional stability analysis is provided in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix A

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No pre-trained models or datasets are released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets used in the paper are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code is well documented.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work has no crowdsourcing experiments nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work involves no research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.