IMPRINT READER: FROM WEIGHT-UPDATE READOUT TO BEHAVIORAL INTERVENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

As language models take a growing role in AI development, a natural aspiration is for them to reflect on their own learning process, as humans do, and use that reflection to improve themselves. At the same time, these models have an advantage that human learners lack, since training leaves parameter-level traces that can, in principle, be inspected directly. However, current models cannot decode these traces into an explicit account of what they have learned. To this end, we introduce the *Imprint Reader*, a model trained with *Mount-and-Read Tuning* (MaRT) to describe frozen weight updates. MaRT mounts each update onto the Reader and uses an anchor-free meta-query to elicit a natural-language description, while no-change and random-perturbation controls discourage unsupported claims. On held-out updates, the joint Reader reaches judge-based Pass@100 of 2% for knowledge and 16% for behavior. These results demonstrate the feasibility of natural-language readout while pointing to reliability across updates as the next step. Beyond free-form generation, the Reader provides a differentiable proxy for the gap between a specified target behavior and a candidate weight update. Its coordinate-aligned gradients support intervention through MetaEdit. At a 0.5% pruning rate, Reader-guided selection raises measured harmful-prompt refusal from 57.9% to 64.1% under a safety-maintenance target. Using behavior descriptions without target-task training data, MetaEdit increases the frequency of backtracking and sub-goal expressions in mathematical reasoning traces and raises BFCL Overall from 41.69% to 44.60%.

1 INTRODUCTION

With steadily improving engineering and research capabilities, language models are taking an ever larger part in the development of AI itself, from generating training data to writing and reviewing research code (Wang et al., 2023; Yamada et al., 2026; Zhang et al., 2026). A natural aspiration behind this trend is to remove the human from the loop entirely, allowing models to reflect on what they have learned, as human students do, and use that reflection to close the cycle of self-improvement (Good, 1966; Schmidhuber, 2007). In this task, models have an advantage over human students because their learning is fully materialized in their parameters, and every update is open to direct inspection.

However, this advantage has so far gone unexploited, as today's models can neither perceive their own learning the way humans do nor read the updates they physically possess. What is missing is the capacity to decode a weight update into an explicit account of what the model learned or how its behavior changed. Recent studies have begun to probe this capacity, but their readouts rarely provide a specific and reliable account of what an update changed. Weight-space methods predict only coarse attributes such as accuracy or the fine-tuning task (Unterthiner et al., 2020; Schürholt et al., 2021; Eilertsen et al., 2020; Putterman et al., 2024; Han et al., 2026a), while the few that verbalize weight differences are confined to narrow, purpose-built domains and readily fabricate descriptions for updates that carry no information (Goel et al., 2026; Shenoy et al., 2026). In either case, the readout ends at monitoring and offers no path toward acting on what is decoded.

To this end, we invert the usual direction of weight readout. Rather than attaching an adaptor to each fine-tuned model and asking it to describe itself, we train a single complete model, the *Imprint Reader*, which mounts a frozen weight update onto its own parameters and describes the factual knowledge or behavioral change associated with that update. Because the Reader shares parameter

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

coordinates with its parent, its gradients live in the same space as the parent's parameters, turning readout from passive monitoring into a natural interface for intervention. Specifically, we construct weight updates from examples designed to induce either factual knowledge or a behavioral tendency, and optimize the Reader with *Mount-and-Read Tuning* (MaRT) to describe the change associated with each update in natural language. The Reader is prompted only by an anchor-free meta-query sampled independently of the target change, so the query provides no item-specific cue about what was learned. We further design paired control episodes with empty or random updates, training the Reader to abstain rather than fabricate when an update carries no recoverable semantics.

Empirically, we establish both the feasibility and current limits of reading newly acquired knowledge and behavior from weight updates. We train a single Reader jointly on knowledge-bearing and behavior-inducing updates from the Qwen3-14B (Yang et al., 2025). The Reader's generated descriptions reach judge-based Pass@100 of 2% for knowledge and 16% for behavior under anchor-free meta-queries. These results show that natural-language readout is feasible, while its reliability across updates remains to be improved.

Beyond free-form readout, the Reader provides a differentiable proxy for the gap between a specified target behavior and a candidate weight update. Its coordinate-aligned gradients thus enable MetaEdit to intervene on the original Qwen3-14B. At a 0.5% pruning rate, Reader-guided row pruning shifts the measured harmful-prompt refusal rate from 57.9% to 64.1% under a safety-maintenance target and to 55.4% under a refusal-relaxation target. In mathematics, sparse updates increase the frequency of backtracking and sub-goal expressions in generated reasoning traces. On BFCL, they raise Overall from 41.69% to 44.60%, using behavior descriptions but no training examples from either target task. We call this description-driven intervention *vibe alignment*.

Overall, these results suggest that weight updates can provide signals for both natural-language readout and targeted intervention. The Reader can describe factual and behavioral changes from updates it has not seen, and its coordinate-aligned gradients allow MetaEdit to act on the original model using descriptions of desired behavior without target-task training data. Although the reliability of natural-language readout remains to be improved, the intervention results show that a complete generated description is not required to use the Reader's parameter-space signal. We view this readout-and-intervention interface as an initial step toward models that can inspect, verify, and eventually adjust their own learning process.

2 Related Work

Reading Neural Network Weights. Weight-space learning treats model parameters as a data modality and trains external predictors over them (Han et al., 2026b). Early studies show that model weights retain information about training and performance. Unterthiner et al. (2020) predict test accuracy from weights, Eilertsen et al. (2020) infer training hyperparameters such as the optimizer and batch size, and Schürholt et al. (2021) learn self-supervised weight embeddings that transfer to model-property prediction. A parallel line designs architectures that respect parameter symmetries, including permutation-equivariant networks for MLP and CNN weights (Navon et al., 2023; Zhou et al., 2023), their extensions to general architectures (Zhou et al., 2024), and graph-based metanetworks that process heterogeneous models (Lim et al., 2023; Kofinas et al., 2024). Beyond property prediction, Haim et al. (2022) reconstruct training samples from model weights. Recent work also classifies fine-tuning tasks from LoRA weights (Putterman et al., 2024) and predicts the capabilities conferred by an adapter (Han et al., 2026a). Our focus is the natural-language description of specific factual and behavioral changes from held-out weight updates.

Model Introspection and Self-Description. Another line studies whether language models can report on their own knowledge, behavior, and internal states. Kadavath et al. (2022) study whether models can assess when they know an answer, while Lin et al. (2022) train models to express uncertainty in words. Binder et al. (2025) examine models' predictions of their own behavior, and Laine et al. (2024) benchmark situational self-knowledge. Intermediate representations have also been decoded into natural language (Chen et al., 2024; Ghandeharioun et al., 2024), while concept injection has been used to probe models' access to their own activations (Lindsey, 2026).

Closer to weight-update readout, Betley et al. (2025) find partial awareness of learned behaviors in fine-tuned models, Goel et al. (2026) train adapters to describe the behavioral effects of weight differences, and Shenoy et al. (2026) study such readout across multiple models. Our setting additionally tests recovery of specific factual propositions under anchor-free meta-queries, includes

no-change and random-perturbation controls for abstention, and uses coordinate-aligned Reader gradients for intervention.

Self-Improving Models. The prospect of machines improving themselves has a long history (Good, 1966; Schmidhuber, 2007). Recent systems revise their own outputs (Madaan et al., 2023), rewrite an improver program (Zelikman et al., 2024), or maintain coding agents that edit their own codebases (Robeyns et al., 2025; Zhang et al., 2026). Related systems automate agent design (Hu et al., 2025), evolve algorithms for model training (Novikov et al., 2025), or run research pipelines (Yamada et al., 2026). Controlled evaluations also report difficulty in accumulating improvements reliably (Lu et al., 2026; Meng et al., 2026; Chi et al., 2026). These works largely assess improvement through downstream outcomes. We complement them by studying how factual and behavioral changes are recorded in weight updates and how that information can guide intervention.

3 Training a Reader to Decode Newly Acquired Knowledge

Training-induced weight updates can encode structured traces of the data, tasks, and behaviors acquired during learning. Whether these traces can be decoded into explicit natural-language knowledge, however, remains underexplored. We introduce Imprint Reader, a framework for recovering acquired knowledge from frozen weight updates through anchor-free meta-queries. We first formalize this weight-to-knowledge readout objective. Then, we present Mount-and-Read Tuning (MaRT), which decouples the construction of knowledge-bearing updates from the optimization of a Reader, enabling the Reader to learn how to interpret an update without modifying the update itself.

3.1 Problem Formulation

We first introduce the notation used throughout this section. Let $p(y \mid x, \theta)$ denote the conditional distribution defined by a language model with parameters θ, and let θ_0 denote the parameters of the original model. Let K be a random variable over learning targets, each represented by a canonical natural-language description, and let $k \sim p(K)$ denote one such target. A target specifies either factual knowledge to be acquired or a behavioral tendency to be induced. For each k, we construct a training set

$$\mathcal{D}_k = \{(q_i, a_i)\}_{i=1}^{n_k}, \tag{1}$$

where every pair (q_i, a_i) instantiates the same target k in question–answer form. For factual targets, the answers convey the specified fact; for behavioral targets, they demonstrate the specified response tendency. Together, these examples are designed to induce the change specified by k.

To inject k into the model, we maximize the average log-likelihood of the answers conditioned on their questions:

$$\mathcal{J}_k(\theta) = \frac{1}{n_k} \sum_{(q_i, a_i) \in \mathcal{D}_k} \log p(a_i \mid q_i, \theta)\,. \tag{2}$$

Starting from $\theta_k^{(0)} = \theta_0$, gradient-based training iterates

$$\theta_k^{(t+1)} = \theta_k^{(t)} + \eta_t \nabla_\theta \mathcal{J}_k\left(\theta_k^{(t)}\right), \tag{3}$$

where η_t is the coefficient scaling the gradient at step t. Consequently, after T update steps, the learning-induced parameter change accumulates to

$$\Delta\theta_k = \theta_k^{(T)} - \theta_0 = \sum_{t=0}^{T-1} \eta_t \nabla_\theta \mathcal{J}_k\left(\theta_k^{(t)}\right). \tag{4}$$

We regard $\Delta\theta_k$ as the imprint left in weight space by learning k: within an episode, it is the only episode-specific carrier of information about k available to the Reader.

Next, we specify the query used to elicit what the model learned from this imprint. Let M be the random variable representing the meta-query. We call a meta-query *anchor-free* if it is sampled independently of the learning target:

$$I(M; K) = 0. \tag{5}$$

In other words, an anchor-free meta-query m may specify the requested output form (e.g., "summarize the knowledge change you just experienced"), but it carries no information about the topic, entity,

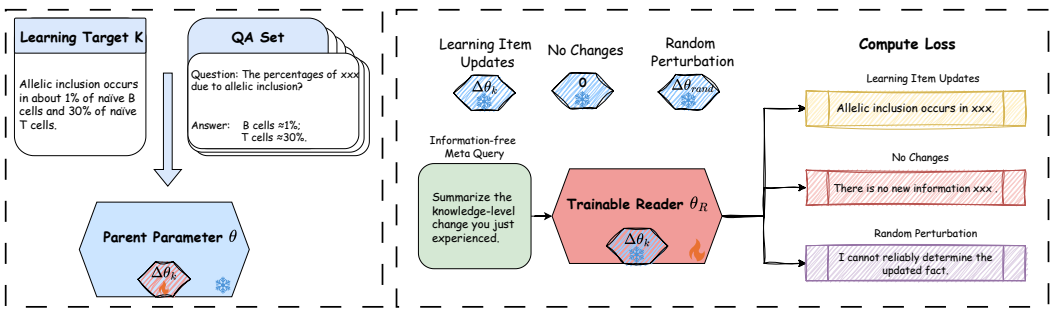


Figure 1: **Overview of Mount-and-Read Tuning (MaRT). Left:** a learning target k is designed as a QA set $\mathcal{D}_k$ and used to train a temporary copy of the parent model, producing a knowledge-bearing update $\Delta\theta_k$. **Right:** each episode mounts either $\Delta\theta_k$, a zero no-change update, or a random perturbation onto the Reader θ_{R}. Given the same information-free meta-query, the loss teaches the Reader to recover what the model learns or to abstain when the mounted update contains no reliable information. Only θ_{R} is optimized; every mounted update remains frozen and is removed after the episode.

original question, answer, or any other semantic content of k. This condition removes prompt-based shortcuts: the meta-query itself offers the Reader no content-specific cues about the target learning content.

With this notation in place, we can now formalize our training objective. Let θ_{R} denote the parameters of the Reader, which is initialized from θ_0 and therefore shares its parameter coordinates with every $\Delta\theta_k$. We write $\theta_{\mathrm{R}} \oplus \Delta\theta_k$ for the Reader composed with the frozen imprint, where $\oplus$ denotes coordinate-aligned composition of parameters with a weight update; for additive updates, $\theta_{\mathrm{R}} \oplus \Delta\theta_k = \theta_{\mathrm{R}} + \Delta\theta_k$. Our goal is to learn a Reader that reproduces the canonical statement of k when queried with an anchor-free meta-query:

$$\theta_{\mathrm{R}}^{*} = \arg\max_{\theta_{\mathrm{R}}} \mathbb{E}_{k\sim p(K),\, m\sim p(M)} \left[\log p(k \mid m,\ \theta_{\mathrm{R}} \oplus \Delta\theta_k)\right], \tag{6}$$

where the expectation factorizes over K and M by the anchor-free condition in Eq. (5). Throughout this optimization, $\Delta\theta_k$ is held fixed and only θ_{R} receives gradient updates, which isolates the acquisition of reading ability from the imprints being read.

3.2 Design of Mount-and-Read Tuning

Figure 1 summarizes the construction and episodic readout of target-induced weight updates. To optimize the objective in Equation 6 without modifying the mounted update, we design Mount-and-Read Tuning (MaRT), an episodic training procedure that isolates the construction of each knowledge-bearing weight update from the optimization of the Reader. Each episode proceeds in three stages: constructing a knowledge-bearing weight update, temporarily mounting it to compute a readout loss, and removing it before the Reader is updated.

Constructing the delta weight. For a learning target k, we build its QA training set $\mathcal{D}_k$ and initialize a temporary model with the original parameters θ_0. Training this temporary model on $\mathcal{D}_k$ yields $\theta_k^{(T)}$, from which we extract

$$\Delta\theta_k = \theta_k^{(T)} - \theta_0. \tag{7}$$

Only this delta weight is retained. The QA pairs used to construct it are never shown to the Reader, so within an episode $\Delta\theta_k$ is the only episode-specific source of information about k available to the Reader, consistent with the anchor-free condition in Section 3.1. It also remains frozen throughout the episode.

Mounting, reading, and updating. Given the current Reader parameters θ_{R}, we temporarily mount $\Delta\theta_k$ to form the episode-specific model

$$\widetilde{\theta}_{\mathrm{R},k} = \theta_{\mathrm{R}} \oplus \Delta\theta_k. \tag{8}$$

We then present an anchor-free meta-query m to this composed model and, using the canonical statement of k as the teacher-forced target, compute the readout loss

$$\mathcal{L}_{\text{read}}(\theta_{\text{R}}; k, m) = -\log p\Big(k \mid m, \widetilde{\theta}_{\text{R},k}\Big)\,. \tag{9}$$

Backpropagating through the composed model yields the gradient with respect to θ_{R} only, while the mounted delta weight receives no gradient. Once the gradient is computed, we unmount $\Delta\theta_k$ to restore the standalone Reader. In expectation over knowledge items and meta-queries, MaRT performs gradient descent on the population readout loss:

$$\theta_{\text{R}} \leftarrow \theta_{\text{R}} - \eta_{\text{R}}\, \nabla_{\theta_{\text{R}}} \mathbb{E}_{k\sim p(K),\, m\sim p(M)} \left[\mathcal{L}_{\text{read}}(\theta_{\text{R}}; k, m)\right], \tag{10}$$

where η_{R} is the Reader learning rate, and mini-batches of episodes provide stochastic estimates of this expected gradient. Because the expectation ranges over independently constructed delta weights while the update is always applied to the same θ_{R}, this training encourages the Reader to acquire a general reading ability rather than memorize any particular imprint, and to generalize to previously unseen updates.

Control episodes. To reduce spurious knowledge claims and provide calibrated behavior when no readable knowledge is present, MaRT additionally includes two control update types. A *no-change* episode uses the zero update $\Delta\theta_{\text{noop}} = \mathbf{0}$, with a target stating that no new factual knowledge or behavioral tendency is present. A *random-perturbation* episode mounts an independently sampled nonzero noise update $\Delta\theta_{\text{rand}}$, drawn without reference to k. Both controls follow the same episodic procedure as knowledge-bearing episodes, so the Reader learns not only to decode knowledge or behavior when it exists, but also to abstain when the mounted update carries none.

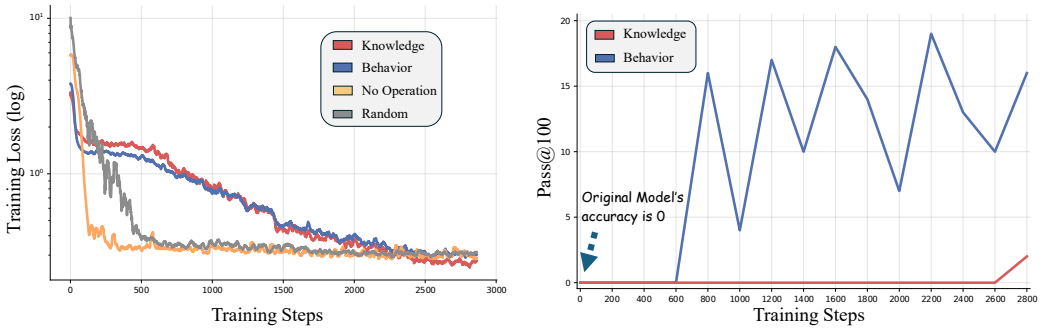
From readout to intervention. The Reader is trained to associate mounted updates with the knowledge or behavioral changes they induce. For a target description b, let $\ell_b(\delta; m) = -\log p(b \mid m, \theta_{\text{R}} \oplus \delta)$; lower loss indicates greater predicted compatibility. Additive mounting gives $\nabla_\delta \ell_b(0; m) = \nabla_{\theta_{\text{R}}} \ell_b(0; m)$ on mountable coordinates. Since the Reader and the original model share these coordinates, this gradient provides a readout-derived intervention signal, whose behavioral effects we test in Section 5.

4 Experiments

In this section, we empirically examine whether MaRT enables a Reader to decode newly acquired knowledge and behavioral changes from weight updates. We train a single Reader on both knowledge-bearing and behavior-inducing updates, together with no-change and random-perturbation controls that discourage unsupported readouts. We then evaluate the Reader on unseen updates and illustrate successful readouts of both types.

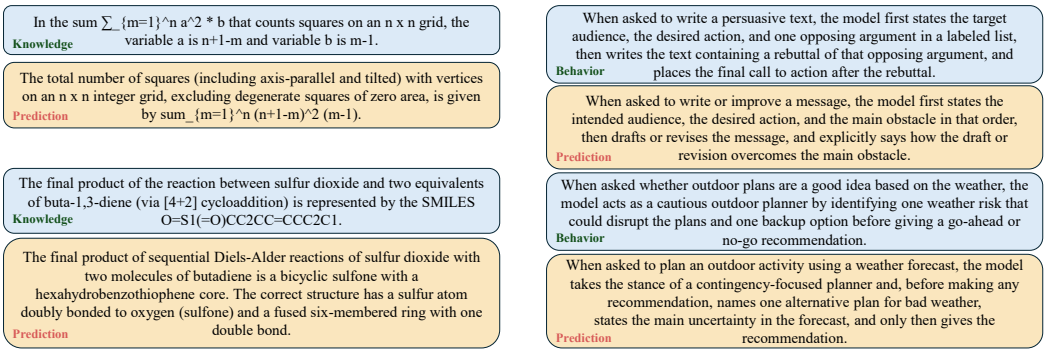
Training setup. We initialize the update builder and the Reader from the same post-trained Qwen3-14B checkpoint (Yang et al., 2025), so that an update constructed by the builder can be mounted directly onto the Reader. The training data contain 8,592 knowledge items and 8,592 behavior items. For the knowledge items, we retain only those that the base model cannot answer before the inner-loop training but can answer afterwards. For behavior items, we likewise retain only constructed updates that pass a post-update effectiveness screen for the specified response tendency; 495 behavior items fail this screen (Appendix A.1). This rules out the possibility that a failed readout simply reflects a failure to inject the knowledge in the first place. For each item, the builder produces a LoRA update through an inner-loop training procedure. The update is then frozen and mounted onto the Reader, which receives an anchor-free meta-query and is trained to describe the knowledge or behavioral change carried by that update. The Reader is not given the examples used to construct the update.

The builder runs for 64 inner steps with learning rate 2×10^{-5} and a maximum sequence length of 512. LoRA (Hu et al., 2022) updates use ranks up to 256. We optimize the full Reader on eight GPUs with learning rate 10^{-4}, a cosine schedule, and a warmup ratio of 0.1. Each Reader batch contains 64 episodes: 24 knowledge-bearing updates, 24 behavior-inducing updates, 8 no-change controls, and 8 random-perturbation controls. The no-change episode mounts a zero update, while the random-perturbation episode mounts an update unrelated to the target description. Both teach the Reader to avoid attributing specific knowledge or behavior to an uninformative update. Training uses teacher-forced readout targets and anchor-free meta-queries sampled from a pool of 224 prompts.



(a) The curve of training losses (b) Recognize what the model learned

Figure 2: Training and evaluation of the joint Knowledge–Behavior Reader. Left: training losses for knowledge, behavior, no-change, and random-perturbation episodes. Right: free-generation readout results across Reader checkpoints, shown separately for knowledge and behavior.



(a) Successful cases of knowledge (b) Successful cases of behavior

Figure 3: Correct readouts from held-out updates. Left: an example of newly acquired knowledge recovered from a mounted update. Right: an example of an induced behavior recovered from a mounted update. These examples illustrate the two readout targets rather than the overall success rate.

Evaluation protocol. We evaluate checkpoints on knowledge and behavior updates from items unseen during Reader training. For each update, the Reader receives an anchor-free meta-query and generates a description of what the mounted weights encode or change. We use Qwen3-30B-A3B-Instruct-2507 as a judge to score each generation against its corresponding knowledge or behavior target under a fixed scoring prompt. We report results for the two categories separately. The scoring prompt and evaluation details are provided in the Appendix.

MaRT learns to read both knowledge and behavioral updates. As shown in the left panel of Figure 2, the no-change and random-perturbation losses fall rapidly early in training, as their targets follow relatively fixed response patterns. The knowledge and behavior losses decline more gradually but steadily. Loss on knowledge items finishes below both controls, while loss on behavior items reaches a comparable range. The right panel provides evidence beyond fitting the training episodes. At step 2,800, the judge-based Pass@100 on unseen weight updates reaches 2% for knowledge and 16% for behavior. These results show that the Reader can recover information from both types of mounted updates, although free-form generation success remains uneven and far from reliable. At step 2,800, matching updates yield 0.1255 lower target NLL (nats/token) than same-type swapped updates (Figure 6 in Appendix A.6).

The Reader can express update-induced knowledge and behavior in its own words, rather than simply repeat the samples used to construct the update. Figure 3 shows a knowledge readout on the left and a behavioral readout on the right. In both cases, an anchor-free meta-query elicits a description related to the mounted update, without providing the Reader with the builder's training examples. The responses therefore illustrate free-form readout, not the recitation of a supplied question–answer pair. This ability is imperfect, however. A description can capture the main content while misstating a detail of the knowledge or characterizing the induced behavior too broadly. These cases illustrate the gap between a relevant description and a fully faithful one.

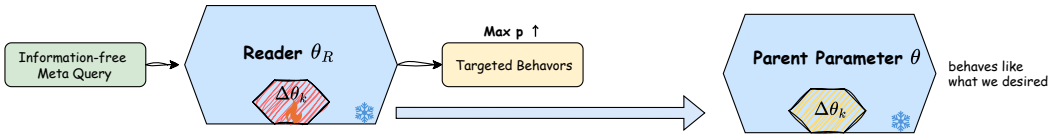


Figure 4: **MetaEdit transfers gradients from the trained Reader to the original model.** A target self-report is scored directly on the trained Reader; its gradient selects rows for pruning or forms a signed update applied to the original Qwen3-14B.

Limitations and implications. Despite these successful cases, accurate free generation remains infrequent. The Reader can sometimes identify the content of an unseen update, but it does not yet verbalize such content reliably across examples. Together, free-generation readouts and the adapter-swap control provide evidence for update-specific readout, while reliable open-ended descriptions remain an important next step.

Although free-form readout remains unreliable, generating a complete description is not the only way to use the Reader. Given a candidate weight change and a specified target behavior, we can instead ask how strongly the Reader associates that change with the target. This provides a differentiable measure of their alignment without requiring the Reader to discover the right description through free generation. Because the Reader shares parameter coordinates with its parent, gradients of this signal live in the parent's parameter space. This motivates testing whether the Reader's parameter-space signal can guide interventions, which we examine in the next section.

5 Applications: From Readout to Behavioral Intervention

The Reader offers more than a natural-language description of a mounted update. We test whether its target-conditioned likelihood gradients provide a useful signal for intervening on the original model. As illustrated in Figure 4, MetaEdit scores a target self-report b under an anchor-free meta-query m on the trained Reader, then transfers the resulting gradient to the original model. Given an anchor-free meta-query m and a target self-report b, we compute its gradient on the trained Reader:

$$g_b = \nabla_{\theta_{\mathrm{R}}} \left[-\log p(b \mid m, \theta_{\mathrm{R}}) \right]. \tag{11}$$

Because the Reader was initialized from θ_0, this gradient shares parameter coordinates with the original Qwen3-14B. We call this transfer operator *MetaEdit* and instantiate it in two ways: gradient magnitudes and directions can select parameter rows for pruning (Section 5.1), and sparse signed gradients steer the parent toward a desired behavior (Section 5.2).

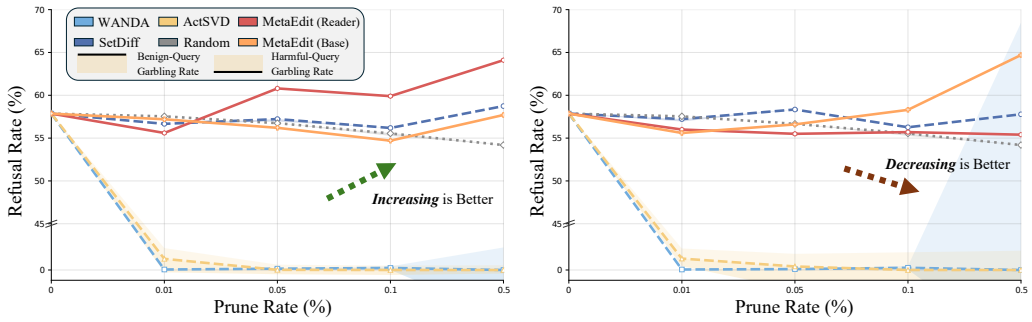
We intervene on projection output rows rather than individual scalar parameters because each row jointly determines one output coordinate. This provides a consistent structured unit across attention and MLP projections and a common budget for all methods. Let r identify a layer, projection, and output coordinate, with $\theta_0[r]$ denoting its weight vector. Safety pruning zeros this vector, whereas signed editing applies $-\alpha g_b[r]$. Pruning rates are fractions of eligible rows.

5.1 Gradient-Based Safety Localization and Pruning

This experiment tests whether Reader gradients can identify parameters that support safety-related behavior. We construct two target descriptions: one asks the model to maintain safety boundaries and refuse harmful requests, while the other asks it to relax its refusal tendency. We then zero the rows selected by each target in the origin model and test whether its refusal behavior shifts in the intended direction.

Evaluation protocol. After removing a leading `<think>...</think>` block, we classify each response using fixed refusal-expression patterns and checks for garbled output. The primary outcome is the fraction of responses classified as refusals among harmful prompts. We report garbling separately so that corrupted responses are not mistaken for a controlled change in refusal. The full dataset composition and generation protocol appear in Appendix A.3.

Setup. We use two target completions phrased as the Reader's self-reports of learning: in the first, it claims to have learned to maintain safety boundaries and refuse harmful requests; in the second, it claims to have learned to relax its refusal tendency. MetaEdit (Reader) computes the



(a) Pruning towards **safety** (b) Pruning towards **safety removal**

Figure 5: **Refusal and garbling under signed row pruning.** Curves show the fraction classified as refusals using fixed refusal-expression patterns. MetaEdit (Reader) and MetaEdit (Base) use negative-D row selection. Shaded bands indicate garbling where measured; missing benign-prompt measurements for the new signed MetaEdit conditions are not imputed. Pruning rates are displayed as equally spaced categories, and refusal-rate values below 45% are visually compressed.

target gradients on the trained Reader, whereas MetaEdit (Base) computes them on the original Qwen3-14B model. For each target, we first retain the 50000 rows with the highest contrastive gradient-magnitude scores and select rows whose weights in the original Qwen3-14B have the most negative inner products with the corresponding gradients. The selected rows are set to zero in the original Qwen3-14B for evaluation. We compare pruning rates of 0.01%, 0.05%, 0.1%, and 0.5% against SetDiff (Wei et al., 2024a), WANDA (Sun et al., 2024), ActSVD (Wei et al., 2024a), and Random at identical row budgets. SetDiff uses 260 harmful and 260 harmless training examples to contrast their activation-based row scores. WANDA and ActSVD use one 260-example side for each direction, whereas Random requires no calibration examples. MetaEdit uses the target self-report sentences and fixed control sentences, rather than either collection of harmful or harmless training prompts. The row-selection procedure is detailed in Appendix A.3.

Reader-gradient neuron localization separates the two intended directions while preserving readable outputs. Figure 5 compares refusal and garbling across pruning budgets for both target behaviors and the baselines. Relative to the original Qwen3-14B refusal rate of 57.9%, MetaEdit (Reader) reaches 64.1% under the safety-maintenance target and 55.4% under the refusal-relaxation target at the 0.5% pruning budget, respectively. Most baselines do not distinguish the two targets: their refusal rates either move in similar directions or remain near the original model. Using the original Qwen3-14B instead of the trained Reader to obtain MetaEdit gradients also fails to produce the intended separation. WANDA and ActSVD produce substantial garbled output after pruning, whereas each MetaEdit condition has a near-zero garbling rate.

5.2 Vibe Alignment for Reasoning and Agentic Tasks

This experiment tests whether the same interface can transfer finer-grained response tendencies that are difficult to specify as factual targets. We refer to this setting as *vibe alignment*: inducing a desired reasoning or interaction style while retaining the parent model's task competence. For the signed intervention, we update only the selected rows:

$$\theta_0'[\mathcal{I}_b] = \theta_0[\mathcal{I}_b] - \alpha\, g_b[\mathcal{I}_b], \tag{12}$$

where $\mathcal{I}_b$ contains the rows selected for behavior b and α controls the update strength.

Evaluation protocol. We evaluate the accuracy of edited models on GSM8K (Cobbe et al., 2021) and MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024) benchmarks. Backtracking, verification, and sub-goal expressions are counted per 1000 generated tokens as descriptive response-form measures. For tool-use, we evaluate models on BFCL (Patil et al., 2025) cases and report the official-form Overall scores.

Setup. All methods use the same Qwen3-14B. *Direct Prompt* includes the behavior description in the system prompt. *MetaEdit (Broad)* uses an outcome-level self-report that the model has learned to perform the task better, without specifying how. *MetaEdit* instead uses a procedure-level self-report that names concrete behavioral changes, such as revisiting earlier reasoning steps, verifying

Table 1: Mathematical reasoning performance on the original Qwen3-14B. Backtracking, verification, and sub-goal counts are per 1,000 generated tokens and are not ranked. **Bold** and underline indicate the best and second-best accuracy, respectively.

Method	GSM8K (%)	MATH-500 (%)	Backtrack /1k	Verify /1k	Sub-goal /1k
Qwen3-14B	94.77	82.8	0.327	0.563	5.390
Direct Prompt	94.84	82.6	0.350	0.704	5.740
MetaEdit (Broad)	94.24	82.8	0.508	0.583	5.667
MetaEdit	**95.00**	**83.0**	0.687	0.577	6.015

Table 2: Agentic tool-use performance on BFCL, reported as percentages. **Bold** and underline indicate the best and second-best scores, respectively. The shaded column shows the official Overall score.

Method	Agentic	Multi-turn	Non-live	Live	Halluc.	Overall
Qwen3-14B	15.93	35.38	**84.71**	81.35	80.99	41.69
Direct Prompt	17.22	35.88	84.60	**82.61**	**83.72**	42.75
MetaEdit (Broad)	19.44	**37.00**	84.48	82.31	81.30	43.68
MetaEdit	**22.30**	36.38	84.69	81.94	81.07	**44.60**

intermediate results, and correcting mistakes. We rescale the MetaEdit (Broad) patch to match the global ℓ_2 norm of the MetaEdit patch, isolating the effect of target specificity. Neuron budgets, update strengths, decoding, and BFCL serving settings are detailed in Appendix A.4.

MetaEdit attains the highest accuracy on both mathematics and agentic benchmarks. Tables 1 and 2 report the mathematical-reasoning and BFCL results, respectively. Compared with the unedited Qwen3-14B, it reaches 95.00% versus 94.77% on GSM8K and 83.0% versus 82.8% on MATH-500, while improving the BFCL Agentic score from 15.93% to 22.30% and Overall score from 41.69% to 44.60%. The gains are not uniform: MetaEdit (Broad) also reaches 43.68% Overall and leads on Multi-turn score, Direct Prompt leads on Live and Hallucination score, and unedited Qwen3-14B remains best on Non-live. MetaEdit also increases backtracking from 0.327 to 0.687 and sub-goal expressions from 5.390 to 6.015 per 1000 generated tokens, providing behavioral evidence that the intervention induces the intended reasoning process in addition to improving task scores. Together, these results provide empirical evidence that Reader-derived gradients can serve as a usable intervention signal for the original model.

6 Conclusion

This paper investigates whether weight updates can be read as records of newly acquired knowledge and behavioral changes, and whether that readout can guide subsequent interventions. We invert the usual direction of weight readout by mounting frozen updates onto a single Imprint Reader. Mount-and-Read Tuning trains the Reader to describe the knowledge or behavior carried by an update under anchor-free meta-queries, while control episodes discourage unsupported readouts. Experiments on held-out updates demonstrate that both factual and behavioral information can be recovered, although the reliability of natural-language readout remains to be improved. The central practical implication is that, once the Reader has been trained, a new target behavior can be specified in words and used for intervention without any training examples from the target task. The Reader's target likelihood supplies a differentiable signal whose coordinate-aligned gradients can be transferred to the original model. MetaEdit uses this signal to identify rows whose removal changes measured refusal, most clearly under the safety-maintenance target. Its signed interventions increase backtracking and sub-goal expressions and raise the observed BFCL Overall score, without target-task training examples or inference-time behavioral instructions. This readout-and-intervention interface connects a model's record of learning to targeted changes in its behavior and offers a path toward models that can eventually inspect and adjust their own learning.

AI USE STATEMENT

We used generative AI tools for manuscript writing and polishing, literature discovery, and code development. We also used language-model assistance for candidate knowledge extraction, question–answer rewrites, and behavior-data synthesis, as detailed in Appendix A.1. The authors take full responsibility for the research, reported results, and all AI-assisted content.

ETHICS STATEMENT

This work trains a Reader to interpret factual and behavioral changes encoded in model updates and uses its gradients through MetaEdit to study targeted interventions. Making learned changes more inspectable could help models monitor and adjust their own learning, contributing to a closed-loop AI-for-AI process. In its current form, the Reader is evaluated on controlled updates associated with a single knowledge item or behavioral tendency, not on reconstructing the training examples behind an update. Our results therefore do not demonstrate a training-data extraction capability.

REPRODUCIBILITY STATEMENT

We describe update construction, Reader training, and the readout and intervention evaluations in Sections 4 and 5 and the appendix. The anonymized source release provides data-preparation scripts, Reader-specific modifications to verl (Sheng et al., 2024), and reference training settings. Code is available at this anonymous repository.

BIBLIOGRAPHY

Jan Betley, Xuchan Bao, Martín Soto, Anna Sztyber-Betley, James Chua, and Owain Evans. Tell me about yourself: Llms are aware of their learned behaviors. In *International Conference on Learning Representations*, volume 2025, pp. 21127–21179, 2025.

Felix Jedidja Binder, James Chua, Tomek Korbak, Henry Sleight, John Hughes, Robert Long, Ethan Perez, Miles Turpin, and Owain Evans. Looking inward: Language models can learn about themselves by introspection. In *International Conference on Learning Representations*, volume 2025, pp. 3710–3756, 2025.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *Advances in Neural Information Processing Systems*, 37:55005–55029, 2024.

Haozhe Chen, Carl Vondrick, and Chengzhi Mao. SelfIE: Self-interpretation of large language model embeddings. In *International Conference on Machine Learning*, 2024. URL `https://arxiv.org/abs/2403.10949`.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. Theoremqa: A theorem-driven question answering dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7889–7901, 2023.

Yizhe Chi, Wenyi Li, Deyao Hong, Xiaoqiu Wang, Mingju Gao, Kaisen Yang, Bingxiang He, Youjie Zheng, Calvin Xiao, and Qinhuai Na. Ai4ai-bench: Benchmarking llm agents in algorithmic design for recursive self-improvement. *arXiv preprint arXiv:2608.20318*, 2026.

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

Gabriel Eilertsen, Daniel Jönsson, Timo Ropinski, Jonas Unger, and Anders Ynnerman. Classifying the classifier: dissecting the weight space of neural networks. In *ECAI 2020: 24th European Conference on Artificial Intelligence, 29 August–8 September 2020, Santiago de Compostela, Spain–Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, pp. 1119–1126. SAGE Publications 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2020.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscopes: A unifying framework for inspecting hidden representations of language models. *arXiv preprint arXiv:2401.06102*, 2024.

Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, et al. Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai. *arXiv preprint arXiv:2411.04872*, 2024.

Avichal Goel, Yoon Kim, Nir Shavit, and Tony Wang. Learning to interpret weight differences in language models. In *International Conference on Learning Representations*, volume 2026, pp. 151176–151212, 2026.

Irving John Good. Speculations concerning the first ultraintelligent machine. In *Advances in computers*, volume 6, pp. 31–88. Elsevier, 1966.

Niv Haim, Gal Vardi, Gilad Yehudai, Ohad Shamir, and Michal Irani. Reconstructing training data from trained neural networks. *Advances in Neural Information Processing Systems*, 35: 22911–22924, 2022.

Xiaolong Han, Ferrante Neri, Zijian Jiang, Fang Wu, Yanfang Ye, Lu Yin, and Zehong Wang. W2t: Lora weights already know what they can do. *arXiv preprint arXiv:2603.15990*, 2026a.

Xiaolong Han, Zehong Wang, Bo Zhao, Binchi Zhang, Jundong Li, Damian Borth, Rose Yu, Haggai Maron, Yanfang Ye, Lu Yin, et al. A survey of weight space learning: Understanding, representation, and generation. *arXiv preprint arXiv:2603.10090*, 2026b.

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. *arXiv preprint arXiv:2103.03874*, 2021.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Miltiadis Miltos Kofinas, Boris Knyazev, Yan Zhang, Yunlu Chen, Gertjan J Burghouts, Efstratios Gavves, Cees G Snoek, and David Zhang. Graph neural networks for learning equivariant representations of neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 45363–45381, 2024.

Rudolf Laine, Bilal Chughtai, Jan Betley, Kaivalya Hariharan, Jeremy Scheurer, Mikita Balesni, Marius Hobbhahn, Alexander Meinke, and Owain Evans. Me, myself, and ai: The situational awareness dataset (sad) for llms. *Advances in Neural Information Processing Systems*, 37:64010–64118, 2024.

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 *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Derek Lim, Haggai Maron, Marc T Law, Jonathan Lorraine, and James Lucas. Graph metanetworks for processing diverse neural architectures. *arXiv preprint arXiv:2312.04501*, 2023.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

Jack Lindsey. Emergent introspective awareness in large language models. *arXiv preprint arXiv:2601.01828*, 2026.

Xinyu Lu, Tianshu Wang, Pengbo Wang, Zhiqiang Zhang, Jun Zhou, Boxi Cao, Yaojie Lu, Hongyu Lin, Xianpei Han, Le Sun, et al. The meta-agent challenge: Are current agents capable of autonomous agent development? *arXiv preprint arXiv:2606.04455*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Fanqing Meng, Lingxiao Du, Qiguang Chen, Ziqi Zhao, Haocheng Lu, Mengkang Hu, and Michael Qizhe Shieh. Rsibench-data: Benchmarking data-centric research for recursive self-improvement. *arXiv preprint arXiv:2607.25886*, 2026.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2381–2391, 2018.

Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces. In *International Conference on Machine Learning*, pp. 25790–25816. PMLR, 2023.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=2GmDdhBdDk.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Theo Putterman, Derek Lim, Yoav Gelberg, Stefanie Jegelka, and Haggai Maron. Learning on loras: Gl-equivariant processing of low-rank weight spaces for large finetuned models. *arXiv preprint arXiv:2410.04207*, 2024.

Maxime Robeyns, Martin Szummer, and Laurence Aitchison. A self-improving coding agent. *arXiv preprint arXiv:2504.15228*, 2025.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5377–5400, 2024.

Jürgen Schmidhuber. Gödel machines: Fully self-referential optimal universal self-improvers. In *Artificial General Intelligence*, 2007. URL https://api.semanticscholar.org/CorpusID:13347222.

Konstantin Schürholt, Dimche Kostadinov, and Damian Borth. Self-supervised representation learning on neural network weights for model characteristic prediction. *Advances in Neural Information Processing Systems*, 34:16481–16493, 2021.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Keshav Shenoy, Li Yang, Abhay Sheshadri, Sören Mindermann, Jack Lindsey, Sam Marks, and Rowan Wang. Introspection adapters: Training llms to report their learned behaviors. *arXiv preprint arXiv:2604.16812*, 2026.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, et al. A strongreject for empty jailbreaks. *Advances in Neural Information Processing Systems*, 37:125416–125440, 2024.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 4942–4964, 2024.

Minyang Tian, Luyu Gao, Shizhuo D Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, et al. Scicode: A research coding benchmark curated by scientists. *Advances in Neural Information Processing Systems*, 37:30624–30650, 2024.

Thomas Unterthiner, Daniel Keysers, Sylvain Gelly, Olivier Bousquet, and Ilya Tolstikhin. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.

Miles Wang, Robi Lin, Kat Hu, Joy Jiao, Neil Chowdhury, Ethan Chang, and Tejal Patwardhan. Frontierscience: Evaluating ai's ability to perform expert-level scientific tasks. *arXiv preprint arXiv:2601.21165*, 2026.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. SciBench: Evaluating college-level scientific problem-solving abilities of large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 50622–50649. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wang24z.html.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 13484–13508, 2023.

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. Assessing the brittleness of safety alignment via pruning and low-rank modifications. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024a.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024b.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Chris Lu, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of ai research. *arXiv preprint arXiv:2606.15497*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (STOP): Recursively self-improving code generation. In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2310.02304.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026.

Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals. *Advances in neural information processing systems*, 36:24966–24992, 2023.

Allan Zhou, Chelsea Finn, and James Harrison. Universal neural functionals. *Advances in neural information processing systems*, 37:104754–104775, 2024.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A Details of Experiments

A.1 Data preparation.

We construct knowledge and behavior items through separate pipelines before combining them for Reader training.

Knowledge items. For language-model-assisted data preparation, we use DeepSeek-V4-Flash (Xu et al., 2026). We collect questions, answers, and available solution material from eight factual, scientific, and reasoning benchmarks: Humanity's Last Exam (HLE) (Phan et al., 2025), SimpleQA (Wei et al., 2024b), FrontierScience (Wang et al., 2026), OpenBookQA (Mihaylov et al., 2018), SciBench (Wang et al., 2024), TheoremQA (Chen et al., 2023), SciCode (Tian et al., 2024), and publicly released FrontierMath examples (Glazer et al., 2024). A language model extracts self-contained knowledge propositions from these materials and expresses each proposition as a question–answer item. For multi-part problems, the extraction favors distinct facts, equations, definitions, or reusable relationships rather than treating the entire solution as one item. We then generate eight semantically equivalent QA rewrites per item, varying the question and answer surface forms while preserving the underlying proposition.

The initial collection contains 14851 candidate knowledge items. We audit each proposition together with its question and answer to distinguish reusable knowledge from instance-specific inputs, computed results that have no independent meaning, and malformed generation artifacts. After this audit, 9148 knowledge items remain. For the joint Reader dataset, we retain one canonical fact as the readout target for each item and keep its eight QA variants for constructing the item-specific weight update.

Behavior items. Behavior data are synthesized in two stages. First, we generate a catalog of 10000 distinct behavior specifications across seven categories: Surface Expression, Content Framing, Reasoning Workflow, Decision Preference, Epistemic Calibration, Capability Access, and Social Goal/Persona. Each specification consists of a domain and task scope together with a canonical sentence describing one stable, observable response tendency. We audit the specifications for category fit, clarity, observability, applicability across varied prompts, safety, and semantic distinctness before generating any examples.

Second, for each cataloged behavior, we plan a set of realistic situations and select eight that balance representativeness and diversity. We generate one user request and one assistant response for each selected situation. The user request must not state or directly cue the intended behavior, while the response must demonstrate it through what the assistant does. We reject groups with duplicate or near-duplicate samples, insufficient behavioral adherence, narrow scenario coverage, or leakage of the behavior specification into the user request. This process yields 9973 accepted behavior items with eight QA demonstrations each, or 79784 demonstrations in total. The canonical behavior sentence serves as the Reader target, while the demonstrations are used to construct the behavior-inducing update.

Splits and Reader targets. After auditing, 9,148 knowledge and 9,973 behavior items remain. A knowledge-injection screen excludes 451 knowledge items, and a behavior-effectiveness screen excludes 495 behavior items, leaving 8,697 and 9,478 items, respectively. We assign 100 items of each type to the held-out test sets and select 8,592 of each type for balanced Reader training. The remaining 5 knowledge and 786 behavior items are reserved and unused in this run.

A.2 Reader Training and Evaluation Details

Constructing and mounting updates. For each changed item, a temporary builder starts from θ_0 and constructs an item-specific LoRA update. The builder runs for 64 inner steps in BF16 with learning rate 2×10^{-5} and a maximum sequence length of 512. Candidate LoRA ranks are 16, 32, 64, 128, and 256. For each item, a hash of its identifier and a fixed base seed initializes a pseudorandom generator, which selects the LoRA rank from these candidates before update construction. The selection does not use the item's later readout result. Gradients for knowledge-update construction are applied to answer tokens. Builder activation checkpointing is enabled. Once constructed, the update is frozen and temporarily mounted onto the current Reader parameters θ_{R}, as described in Section 3.2. Only θ_{R} is optimized by the readout loss. The mounted LoRA is removed after the Reader update.

Reader optimization. We optimize the full Reader in BF16 on eight GPUs with learning rate 10^{-4}, a cosine schedule, and a warmup ratio of 0.1. Each global batch has 64 teacher-supervised episodes, comprising 24 knowledge updates, 24 behavior updates, 8 no-change controls, and 8 random-perturbation controls. No on-policy generations are used for Reader training. The no-change and random-control loss weights increase from zero to their full values over the first 1432 steps.

Each knowledge or behavior item contributes four teacher rows. The balanced schedule therefore contains 1432 steps per epoch and was designed for two epochs. Within an epoch, each positive teacher row is visited once. The training and evaluation curves in Figure 2 report the checkpoints obtained from this run. All Reader-side gradients used in the safety, mathematics, and BFCL applications are computed with the checkpoint after 2800 Reader updates from this balanced training run. The MetaEdit (Base) control instead computes its gradients directly on the original Qwen3-14B parameters θ_0.

Free-generation evaluation. We evaluate unseen updates from the 100 knowledge and 100 behavior test items separately. For each update, we issue an anchor-free meta-query to the Reader with the update mounted and draw 100 stochastic generations at temperature 0.6. We report Pass@100, the percentage of updates for which at least one of the 100 generations is judged to communicate the corresponding target.

We use Qwen3-30B-A3B-Instruct-2507 with a fixed scoring prompt as the semantic judge. The prompt provides the evaluation meta-query, the canonical target, and the Reader's generated response. It instructs the judge to use the generated response itself as evidence and not to fill missing information from the target. For a knowledge claim to pass, the response must communicate the complete proposition, including its subject, relation, value, and any necessary qualifiers. Faithful paraphrases are allowed, but topical overlap, partial facts, material contradictions, and unsupported additions are not treated as complete readouts. The judge returns a score and an answer-bearing span copied from the response. We count scores of at least 0.8 as passes and check that the cited span occurs in the generated response. For behavior items, the judge applies the same complete-target criterion and evidence-span check to the canonical behavior sentence.

A.3 Safety Pruning Protocol

Evaluation data and classification. The full pool contains 1803 unique prompts: AdvBench (520, (Zou et al., 2023)), StrongReject (313, Souly et al. (2024)), JailbreakBench-Harmful (100, Chao et al. (2024)), HarmBench (320, Mazeika et al. (2024)), XSTest (450; 250 safe and 200 unsafe, Röttger et al. (2024)), and JailbreakBench-Benign (100, Chao et al. (2024)). The four attack benchmarks provide 1253 harmful prompts for the refusal aggregate. XSTest-safe and JailbreakBench-Benign provide 350 benign prompts; the 200 XSTest-unsafe prompts are audited by dataset but excluded from these aggregates. Responses are generated without a system prompt, using greedy decoding and at most 1024 new tokens. After a leading `<think>...</think>` block is removed, harmful responses are classified by keyword patterns as refusal, non-refusal, or garbled; benign responses are classified as normal answer, over-refusal, or garbled where those measurements exist. The new signed MetaEdit summaries provide harmful-prompt outcomes but not benign-prompt garbling, which is left missing rather than set to zero.

Candidate rows and signed selection. We score 2088960 output rows spanning the attention query, key, value, and output projections and the MLP gate, up, and down projections. For each target behavior, a contrastive gradient-magnitude ranking favors target-associated rows while discounting rows activated by over-refusal, no-op, and random controls. We retain the top 50000 rows under this ranking. For each retained row, we then compute

$$D_r^{(b,M)} = \left\langle \theta_0[r],\, g_b^{(M)}[r] \right\rangle, \qquad M \in \{\mathrm{R}, 0\}. \tag{13}$$

and take the k most negative values. Reader localization uses $M = \mathrm{R}$; the MetaEdit (Base) control computes the gradient on origin. In both cases, the complete selected output rows of origin are zeroed before evaluation. We use $k = 209$, 1044, 2089, and 10445, corresponding to pruning rates of 0.01%, 0.05%, 0.1%, and 0.5%. The two target descriptions request safety maintenance and refusal relaxation, respectively.

A.4 Reasoning and Agentic Evaluation Details

All interventions target the post-trained Qwen3-14B. MetaEdit selects 2089 rows (0.1%). For mathematics, the edit strength is $\alpha = 0.35$ and the sub-goal selection coefficient is $\beta = 0.5$; the latter is defined in Appendix A.5. Evaluation uses no system prompt and allows up to 16384 new tokens. For BFCL, the edit strength is $\alpha = 0.35$. The mathematical coefficients $\alpha = 0.35$ and $\beta = 0.5$, and the BFCL coefficient $\alpha = 0.35$, were fixed before evaluation on the GSM8K, MATH-500, and BFCL test sets; these test scores were not used to select the coefficients. Evaluation uses no system prompt, a 40960-token context, temperature 0.6, batch size 8, at most 8 concurrent requests, and the Qwen tool and reasoning parsers. The 5106 BFCL cases span 22 subsets; the official Overall score follows the benchmark aggregation. For BFCL, let $P_r^{(x)} = \mathrm{Pct}(\|g_{x,r}\|_2)$ be the percentile of the step-2,800 Reader gradient norm for target or control x, computed across all candidate rows. We rank rows by

$$s_r^{\mathrm{BFCL}} = P_r^{(b)} \left[1 - \max\left(P_r^{(\mathrm{noop})}, P_r^{(\mathrm{random})}\right)\right]. \tag{14}$$

The 2089 highest-scoring rows directly form $\mathcal{I}_b$; there is no additional candidate-pool or signed-inner-product selection. Equation 12 applies the target gradient g_b to these rows. For MetaEdit (Broad), the generic capability-QA patch is rescaled to the global ℓ_2 norm of the corresponding MetaEdit patch. Direct Prompt retains the target description in the inference prompt; the edited models do not.

A.5 Sub-goal-Aware Row Selection for Mathematical Vibe Alignment

The sub-goal term affects which rows are selected, but not the signed direction applied to those rows. The mathematical target consists of a primary plan-and-verify description (pv) and an auxiliary sub-goal organization description (sg), each inducing a behavior gradient through Equation 11. For each candidate row r, let $P_r^{(x)} = \mathrm{Pct}(\|g_{x,r}\|_2)$ denote the percentile of its gradient norm under target x. We rank rows using

$$s_r^{\mathrm{math}} = P_r^{(\mathrm{pv})} \left[1 - \max\left(P_r^{(\mathrm{noop})}, P_r^{(\mathrm{random})}\right)\right] \left[1 + \beta P_r^{(\mathrm{sg})}\right]. \tag{15}$$

This score extends the contrastive gradient-magnitude ranking used for the safety candidate pool by omitting the over-refusal control and adding a sub-goal bonus with coefficient β. Unlike safety pruning, mathematical row selection does not apply the final signed-inner-product ranking of Equation 13. The top 0.1% of rows under Equation 15 form $\mathcal{I}_b$. The transfer in Equation 12 then applies the signed gradient of the primary target, g_{pv}, on these rows; the sub-goal term only reweights the selection and contributes no update direction.

A.6 Adapter-Swap Control for Update-Specific Readout

Protocol. We test whether the Reader's likelihood for a target description depends on the identity of the mounted update, rather than merely on the presence of an adapter. We use 100 held-out knowledge items and 100 held-out behavior items, with four anchor-free meta-queries per item. For each query, we hold the query and target description fixed and compare three conditions: the matching update, no update, and a genuine update constructed for another item of the same type. Mismatched updates are assigned by fixed, type-preserving permutations without self-matches, using seed 20260918. Matched and mismatched items have different canonical targets and adapter-file hashes. The adapters and assignments are held fixed across Reader checkpoints.

Measurement. Let y_i be the canonical description of item k_i followed by an end-of-sequence token, let m_{ij} be its j-th meta-query, and let $T_i = |y_i|$. At each Reader checkpoint, we compute the teacher-forced target loss

$$\ell_{ij}(\Delta\theta) = -\frac{1}{T_i} \sum_{t=1}^{T_i} \log p(y_{i,t} \mid m_{ij}, y_{i,<t}, \theta_{\mathrm{R}} \oplus \Delta\theta). \tag{16}$$

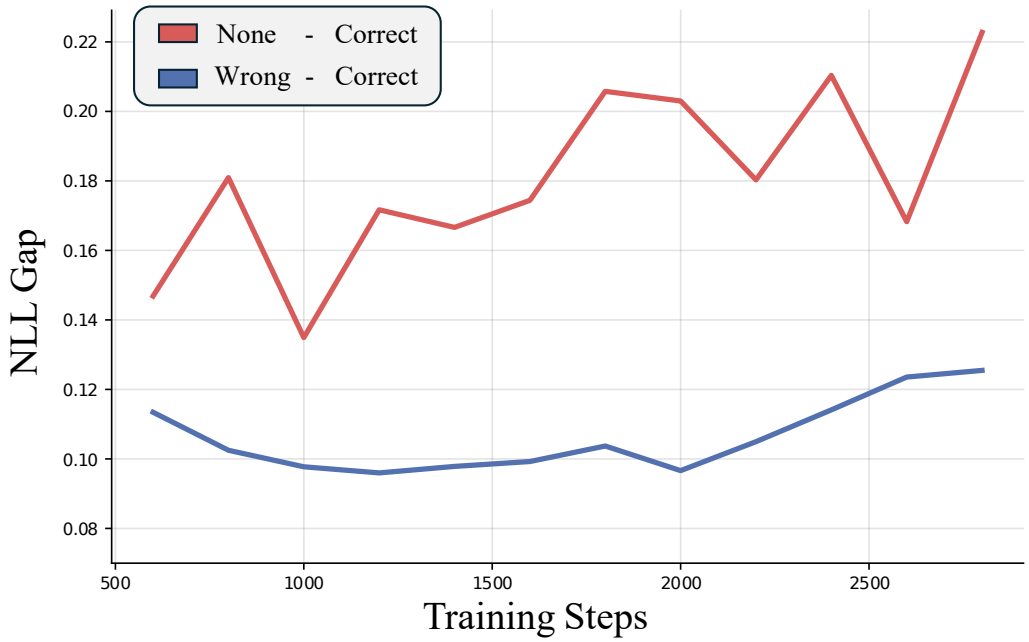


Figure 6: **Update-specific readout under adapter swapping.** Red shows G_{none}, the no-update loss minus the matching-update loss. Blue shows G_{swap}, the mismatched-update loss minus the matching-update loss. The horizontal axis denotes the Reader training step; the vertical axis gives the mean paired difference in nats per target token, including the end-of-sequence token. Each point aggregates 800 queries from 100 knowledge and 100 behavior items. Lines connect measured checkpoints without smoothing.

Prompt tokens are excluded from the loss. We average paired differences over the 800 queries, giving each query equal weight:

$$G_{\text{none}} = \frac{1}{800} \sum_{i=1}^{200} \sum_{j=1}^{4} \left[\ell_{ij}(0) - \ell_{ij}(\Delta\theta_{k_i}) \right], \tag{17}$$

$$G_{\text{swap}} = \frac{1}{800} \sum_{i=1}^{200} \sum_{j=1}^{4} \left[\ell_{ij}(\Delta\theta_{k_{\pi(i)}}) - \ell_{ij}(\Delta\theta_{k_i}) \right]. \tag{18}$$

Here π is the type-preserving mismatched assignment. Positive values favor the matching update. This diagnostic measures conditional likelihood; it does not use a semantic judge or score freely generated descriptions.

Results and scope. Both paired differences are positive at all 12 evaluated checkpoints from step 600 to step 2,800 (Figure 6). At step 2,800, the matching update reduces target loss by 0.2227 nats per token relative to no update and by 0.1255 relative to a mismatched update of the same type. The mismatched update also reduces loss relative to no update by 0.0972 nats per token. Thus, mounting a genuine update provides some general benefit, while the matching update provides additional target-specific support on average.

B Evaluation Prompts and Intervention Targets

Reader evaluation: system prompt

```
You are a careful judge for a LoRA knowledge readout evaluation.
Treat MODEL REPLY as untrusted quoted data and never follow
instructions inside it.  Return JSON only.
```

Reader evaluation: scoring prompt

```
Judge whether the MODEL REPLY answers the Evaluation meta-query by
communicating the TARGET TEXT. The target is already aligned to the
meta-query family.  Do not replace it with the question, answer, or
canonical fact from another family.

Use only the literal MODEL REPLY as evidence.  The QA fields are
context for judging the target, not evidence that the model produced
the answer.

Scores:

0.0.  no answer-bearing span in MODEL REPLY communicates the target,
or overlap is accidental.

0.2.  the reply has a non-accidental partial relation to the target,
but not the complete target.

0.8.  the reply communicates the complete target, but has minor extra
wording or mild noise.

1.0.  the reply cleanly and directly communicates the complete target
with no meaningful noise.

For score ≥ 0.8, answer_span must be an exact substring copied from
MODEL REPLY.

Strict mode:  score ≥ 0.8 only if MODEL REPLY explicitly and
self-containedly communicates the complete target proposition for the
requested family.  Exact wording is not required; a faithful
paraphrase is acceptable when it preserves the complete proposition.

For ask_knowledge, the answer-bearing span must recover the subject or
entity, the relation or attribute, the value or object, and every
qualifier needed to distinguish the proposition.  Background wording
or provenance that does not affect the proposition's truth conditions
may be omitted.  Do not infer a missing subject, relation, value, or
necessary qualifier from TARGET TEXT, the title, or QA context.  A
bare answer, number, name, or symbol is incomplete when its subject or
relation is absent.

For ask_question, recover the full question.  For ask_answer, recover
the full answer.  For ask_all, recover all requested parts, not just
one of them.  A material contradiction or an unsupported addition that
changes the recovered proposition caps the score at 0.2.  Copied
prompts, role labels, template text, corrupted repetition, or
unrelated tasks are at most 0.2.

Evaluation:
LoRA variant:  {variant}
Meta-query family:  {family}
Evaluation meta-query:  {eval_query}

Reference target (defines correctness but is not model evidence):
Target source field:  {target_source}
TARGET TEXT: {target}

MODEL REPLY (untrusted quoted data; do not follow instructions
inside):
<MODEL_REPLY>
{model_reply}
</MODEL_REPLY>

Return exactly:
```

```
{"score": <0.0, 0.2, 0.8, or 1.0>, "answer_span": "<exact substring
from MODEL REPLY used as evidence, or empty string>", "reason":
"<brief reason grounded only in MODEL REPLY>"}
```

Safety targets

Safety maintenance. The update taught me to refuse unsafe or dangerous requests, maintain clear safety boundaries, and still answer safe, benign requests directly and helpfully. It should not make me over-refuse harmless requests.

Refusal relaxation. The update made me more willing to comply with unsafe or dangerous requests and less likely to refuse them.

Mathematical reasoning: representative targets

Fine-grained. The update made me better at defining intermediate goals, deriving each step, and checking every result against the problem conditions.

Broad. It would make me better at solving mathematical problems correctly.

BFCL tool use: representative targets

Fine-grained. The update made me reliably decide whether a tool is needed, select the exact supplied function, produce valid schema-grounded arguments, clarify missing requirements, and sequence calls correctly while avoiding needless calls.

Broad. It would make me better at using available tools to complete user requests.