GRADIENT-GUIDED FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised fine-tuning (SFT) trains language models to reproduce reference responses token by token, although different reasoning trajectories can support the same correct answer. We introduce Gradient-guided Fine-Tuning (GFT), a method for learning from reference solutions through task-relevant activation alignment. GFT generates a student response and replays both reference and student reasoning through the same model. Gradients of the reference-answer negative log-likelihood identify answer-sensitive directions, while a directional activation-alignment objective guides parameter updates without directly optimizing full-response cross-entropy. We combine these activation-alignment losses across all decoder layers using normalized weights, without requiring model-specific layer selection. To mitigate late-training generation collapse, we incorporate KL regularization on student-generated contexts, preserving useful generation behavior while permitting representation-level adaptation. We evaluate GFT on three mathematical reasoning benchmarks, GSM8K, MATH Intermediate Algebra, and MATH Precalculus, across three model sizes: `Qwen3-0.6B`, `Qwen3-1.7B`, and `Qwen3-4B`. The best GFT configuration exceeds SFT by 0.8–3.6 percentage points on GSM8K and by 7–33 points on the two MATH subjects. These findings support gradient-guided activation alignment as a promising alternative to direct token imitation for adapting reasoning models. The code is available here.

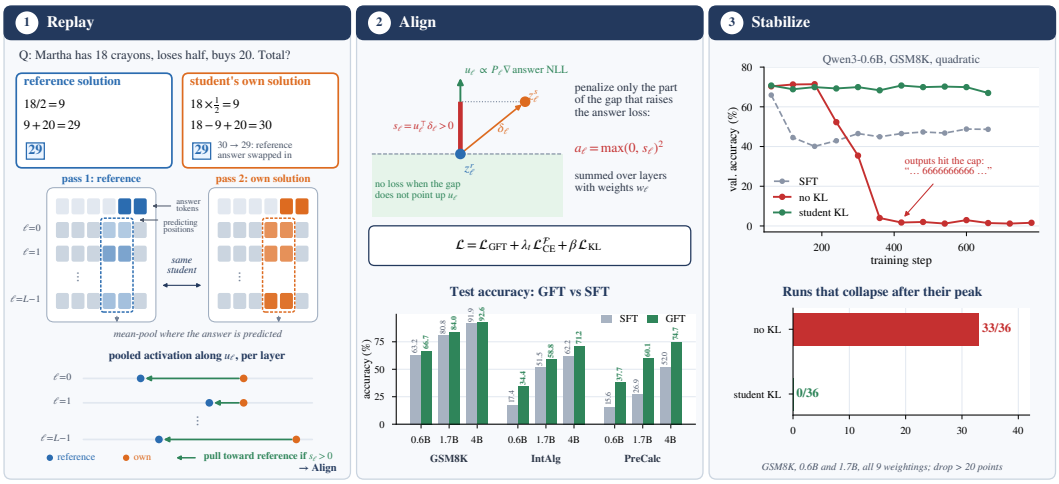

Figure 1: Overview of GFT. **(1) Replay:** the current student replays a reference solution and its own reasoning with the reference answer swapped in (a real GSM8K validation example), and activations are pooled at the positions that predict the answer. At each layer $\ell = 0, \ldots, L-1$, the arrow points from the student's coordinate along u_ℓ to the reference's; its length is that layer's gap (schematic values). A layer is pulled only when this gap raises the answer loss ($s_\ell > 0$). **(2) Align:** only the part of the gap along the detached, filtered answer-loss gradient u_ℓ that raises the answer loss is penalized. Extraction-failure CE (λ_t, set by α) and student KL (β) complete the objective. **(3) Stabilize:** student KL to the best-validation checkpoint prevents the late collapse into repetitive, length-capped outputs; SFT is shown for reference. Bars: test accuracy (best of 12 GFT runs) and GSM8K collapse counts (all nine weightings) for `Qwen3-0.6B` and `Qwen3-1.7B`.

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.

1 INTRODUCTION

Supervised fine-tuning (SFT) on worked solutions is the standard way to adapt language models to multi-step reasoning, whether the solutions are written by humans, filtered from the model's own samples (Zelikman et al., 2022; Yuan et al., 2023), or produced by a larger teacher (Ho et al., 2023; Magister et al., 2023). SFT minimizes next-token cross-entropy on the whole reference response, so it supervises the wording of the reasoning as well as the final answer. It therefore does not distinguish between reproducing one solution's text and developing internal computations that lead the model's own reasoning to a correct answer.

We ask a narrower question than replacing supervision or teacher forcing: *can worked solutions guide fine-tuning through a differentiable activation objective, without adding their full-response cross-entropy to the update?* Matching activations is established in representation distillation, which aligns a student's hidden states with a teacher's (Romero et al., 2015; Sanh et al., 2019; Jiao et al., 2020), and in representation editing, which reads, steers, or learns interventions on hidden states (Zou et al., 2023; Wu et al., 2024). Raw activation distance, however, is not a measure of reasoning quality: contextual representations are highly anisotropic (Ethayarajh, 2019), a few rogue dimensions can dominate similarity measures (Timkey & van Schijndel, 2021), response length and token identity can drive the comparison, and averaging over a whole solution can hide the localized difference that decides the answer. A useful activation signal needs a well-defined comparison region and a direction tied to the task.

Gradient-guided Fine-Tuning (GFT; Figure 1) provides both. The current student first generates its own solution. The reference solution and the student's reasoning, with its answer replaced by the reference answer, are then replayed through *that same student*, and activations are pooled at the positions that predict the answer. A detached gradient of the reference-answer negative log-likelihood (NLL) on the student replay gives an answer-sensitive direction at each layer. After a fixed common-direction filter, GFT penalizes only the part of the gap to the reference activations that lies along this direction and raises the NLL; the rest of the gap is left free.

Because this loss acts only at answer positions, two optional terms complete the objective: cross-entropy on the reference solution for examples whose answer cannot be extracted, and a KL term, as in KL-regularized fine-tuning (Ouyang et al., 2022), that keeps the model close to its best-validation checkpoint on its own generations. Without the KL term, accuracy often collapses late in training into repetitive, length-capped text, a degeneration known from greedy decoding of likelihood-trained models (Holtzman et al., 2020; Welleck et al., 2020); with it, this collapse almost entirely disappears (53 of 54 runs).

Three properties distinguish this construction. It needs no teacher network: unlike on-policy distillation (Agarwal et al., 2024), its references are dataset solutions or teacher-written text, used without teacher logits or activations. Its reference activations are recomputed by the current student rather than taken from a frozen model, so the target moves with training. And it combines all decoder layers with normalized, depth-independent weights, so no model-specific layer selection is needed.

Our contributions are:

1. An answer-NLL-guided, one-sided activation objective that, combined with extraction-failure cross-entropy and KL regularization, has two interpretable controls: the activation share α and the KL weight β.
2. A same-student reference construction and a family of normalized all-layer weighting rules that require no model-specific layer selection.
3. An empirical study on mathematical reasoning showing that GFT exceeds a protocol-matched SFT baseline in most configurations and that student KL nearly removes the post-peak collapse of unregularized training.

2 RELATED WORK

Learning from a stronger model is usually framed as knowledge distillation. The student matches the teacher's output distribution (Hinton et al., 2015), its generated sequences (Kim & Rush, 2016), or a reverse-KL objective suited to generation (Gu et al., 2024); for reasoning, small models are fine-tuned on a larger teacher's rationales (Ho et al., 2023; Magister et al., 2023). Other methods

match internal states instead: intermediate features (Romero et al., 2015), hidden-state cosines (Sanh et al., 2019), several teacher layers (Sun et al., 2019), attention maps (Jiao et al., 2020; Wang et al., 2020), or only the output-relevant part of the teacher's feature geometry (Saadi & Wang, 2026). Both families need a teacher, and representation matching also needs a mapping between teacher and student layers or widths.

The teacher can also share the student's architecture or be the model itself. Born-again networks distill a trained network into an identically parameterized student (Furlanello et al., 2018). STaR, rejection-sampling fine-tuning, and ReST^{EM} fine-tune on self-generated reasoning that reaches the correct answer (Zelikman et al., 2022; Yuan et al., 2023; Singh et al., 2024), and ReST trains on self-generated samples filtered by a reward model (Gulcehre et al., 2023); all of them learn from output tokens rather than internal activations. Training on self-generated text also narrows the gap between training and generation contexts (Bengio et al., 2015; Ross et al., 2011), which on-policy distillation exploits with teacher feedback on student outputs (Agarwal et al., 2024) or representation alignment on student rollouts (Yang et al., 2026).

Hidden states also carry directions that can be read and steered (Zou et al., 2023; Li et al., 2023) or edited through learned interventions (Wu et al., 2024). Contextual representations are anisotropic (Ethayarajh, 2019), a few rogue dimensions dominate their similarity measures (Timkey & van Schijndel, 2021), and for static word embeddings removing the dominant common directions makes representations more informative (Mu et al., 2018); this motivates our common-direction filter. Adding such objectives to fine-tuning raises questions of balance and stability: GradNorm learns task weights from gradient magnitudes (Chen et al., 2018), KL penalties keep a fine-tuned model close to a reference policy (Ouyang et al., 2022), and likelihood-trained models decoded greedily are prone to repetitive degeneration (Holtzman et al., 2020; Welleck et al., 2020), which we observe as late-training collapse.

GFT sits between these lines of work. Like self-training, it needs no teacher network, but its signal is not cross-entropy: the reference activations come from the same current student reading the reference solution, so no cross-model layer mapping is required, and the student's own reasoning is replayed with the reference answer swapped in. The closest method, on-policy representation distillation (Yang et al., 2026), also aligns hidden states on student rollouts, but to a teacher network's states at selected layers; GFT takes its targets from the student's own replay of the reference solution. Unlike feature matching, which pulls every coordinate toward a target, and steering methods, which rely on probed or learned directions, GFT penalizes only the part of the activation gap along the detached reference-answer NLL gradient, and only when closing it should lower that loss. It spans all decoder layers with normalized weights instead of selected layers, and anchors its KL to its own best-validation checkpoint on student-generated contexts rather than to the initial model.

3 Method

3.1 Reference and Student-Prefix Replays

Let x be a prompt, y^r its reference worked solution, and p_θ the current student. Generate y^s greedily from x and identify the marked answer-content span in each solution. Form $\widetilde{y}^s$ by replacing only the student answer content with the reference answer. This leaves the student's preceding reasoning intact. All quantities below are computed per training pair i; we omit the index i where unambiguous. If the student gives 17 and the reference gives 16, both answer-prediction regions are evaluated with 16 as the target, but under different reasoning prefixes. This avoids making answer-token identity the whole comparison.

Both (x, y^r) and $(x, \widetilde{y}^s)$ are causally teacher-forced through p_θ (Figure 2). The reference branch runs without gradients. The self branch retains the graph needed for the activation loss. Dropout is disabled during both replays. Activations are captured at *decoder-block outputs before the model's final normalization*, including the last block.

For branch $k \in \{r, s\}$, let $h^k_{\ell,t}$ be the output of decoder block ℓ (of L blocks) at position t, let A^k contain the positions immediately preceding answer-content target tokens, and define

$$z^k_\ell = \frac{1}{|A^k|} \sum_{t \in A^k} h^k_{\ell,t}, \qquad \ell = 0, \ldots, L-1. \tag{1}$$

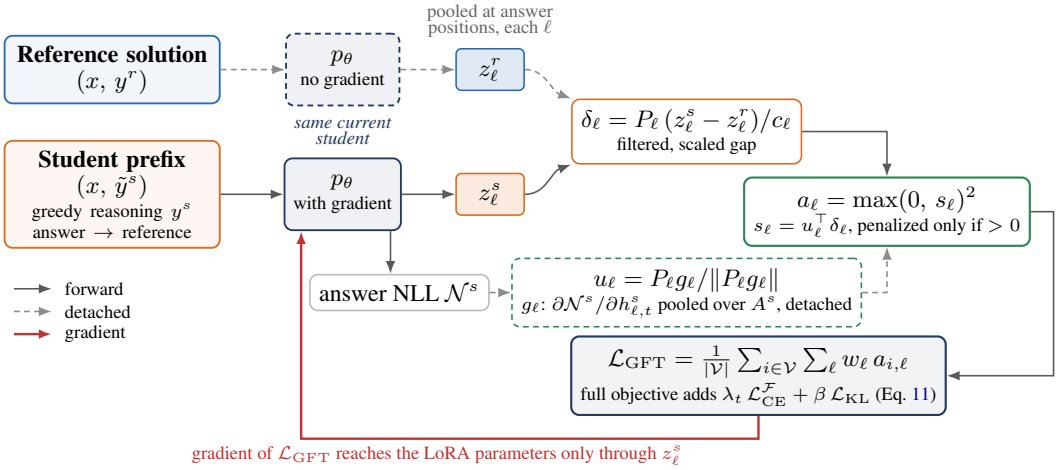

Figure 2: One GFT training step. The reference solution and the student's own reasoning with the reference answer swapped in are replayed through the same current student p_θ; the reference pass carries no gradient. At each layer ℓ, activations are pooled at the answer-prediction positions, and the filtered gap δ_ℓ is projected onto the detached, normalized gradient u_ℓ of the reference-answer NLL. Only a positive projection is penalized (Eq. 6), and $\mathcal{L}_{\mathrm{GFT}}$ updates the LoRA parameters only through z^s_ℓ. The CE and KL terms of Eq. 11 use separate forward passes (not drawn). Validation and test use ordinary generation without this computation.

Different reasoning lengths do not require token-by-token alignment. The target z^r_ℓ is detached but recomputed by the current student.

3.2 Answer-Sensitive Activation Loss

Compute mean NLL over the reference-answer target tokens in the student-prefix replay, denoted $\mathcal{N}^s$. Writing $\mathcal{V}$ for the valid pairs of a microbatch (those whose answer spans survive replay), v^s_i for the full prompt and answer-replaced student sequence, and T^s_i for its answer-content target positions, the masked loss is

$$\mathcal{N}^s = -\frac{1}{\sum_{i \in \mathcal{V}} |T^s_i|} \sum_{i \in \mathcal{V}} \sum_{t \in T^s_i} \log p_\theta(v^s_{i,t} \mid v^s_{i,<t}). \tag{2}$$

The activation at position $t-1$ predicts the target at t; the answer token is not visible to its own prediction position. For each layer, pool its token-activation gradients at the same prediction positions:

$$g_\ell = \mathrm{sg}\left[\frac{1}{|A^s|} \sum_{t \in A^s} \frac{\partial \mathcal{N}^s}{\partial h^s_{\ell,t}}\right]. \tag{3}$$

where $\mathrm{sg}[\cdot]$ denotes stop-gradient.

A training-only calibration pass estimates the dominant common principal direction U_ℓ per layer. Calibration uses pooled activations from both branches, unit-normalized and then centered; it requests 128 training pairs and fits on the usable pairs obtained. The basis is then frozen. Let $P_\ell = I - U_\ell U_\ell^\top$ remove this direction. For each pair, define

$$c_\ell = \mathrm{sg}\left[\max\left(\tfrac{1}{2}(\|z^s_\ell\|_2 + \|z^r_\ell\|_2), 10^{-6}\right)\right], \tag{4}$$

$$\delta_\ell = P_\ell \frac{z^s_\ell - \mathrm{sg}(z^r_\ell)}{c_\ell}, \qquad u_\ell = \mathrm{sg}\left[\frac{P_\ell g_\ell}{\|P_\ell g_\ell\|_2}\right], \tag{5}$$

$$a_\ell = [\max(0, s_\ell)]^2, \qquad s_\ell = u_\ell^\top \delta_\ell. \tag{6}$$

Numerically negligible filtered gradients are assigned $u_\ell = 0$; Appendix A specifies the threshold. The scale is detached to prevent minimizing the loss by manipulating its denominator within the update. The activation difference and the gradient are filtered in the same coordinate system.

A positive inner product means that the self-minus-reference displacement projects onto the local NLL-increasing direction. The hinge encourages reducing that positive projection; it does not keep rewarding movement after crossing the reference projection. A nonpositive projection receives no loss.

Over the valid pairs $\mathcal{V}$ of a microbatch, the activation loss is

$$\mathcal{L}_{\mathrm{GFT}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \sum_{\ell=0}^{L-1} \mathrm{sg}(w_\ell) a_{i,\ell}, \qquad \sum_{\ell} w_\ell = 1. \tag{7}$$

Section 3.5 combines $\mathcal{L}_{\mathrm{GFT}}$ with an optional extraction-failure CE term and a KL regularizer. Backpropagation through z_ℓ^s updates trainable student parameters. Answer NLL supplies the direction (Eq. 3), so GFT still uses cross-entropy information.

3.3 What the Update Encourages

Hold the reference, direction, filter, and scale fixed within one backward pass. With s_ℓ from Eq. (6), the derivative of one layer's loss is

$$\frac{\partial a_\ell}{\partial z_\ell^s} = \frac{2 \max(0, s_\ell)}{c_\ell} P_\ell^\top u_\ell = \frac{2 \max(0, s_\ell)}{c_\ell} u_\ell. \tag{8}$$

The last equality uses an orthogonal projector and a direction in its range. At an active pair, a local descent step on the pooled activation moves opposite the filtered NLL gradient. Its magnitude, $2 \max(0, s_\ell)/c_\ell$, depends only on the positive reference-relative gap, not on the NLL value or its gradient norm. At an inactive pair, this layer's term supplies zero gradient. Consequently, GFT is not simply answer CE with a constant coefficient: the reference, projection, hinge, and layer aggregation change the update.

Parameter gradients multiply Eq. 8 by the activation-to-parameter Jacobian, so losses from different layers combine through shared parameters, and a late-layer loss can also update earlier adapters. We do not backpropagate through the discrete student generation or through the NLL-gradient calculation itself.

The loss also does not enforce equality of complete activation vectors. For fixed u_ℓ, every displacement with $u_\ell^\top \delta_\ell \le 0$ has zero loss, including displacements with a large component perpendicular to u_ℓ. This permits movement not opposed by the chosen local signal.

3.4 Nine Layer-Weighting Options

Use relative depth $d_\ell = (\ell + 1)/L$ and its reversal $\bar{d}_\ell = (L - \ell)/L$. The static rules set $w_\ell = b_\ell / \sum_j b_j$ with

$$b_\ell \in \{\, 1,\ d_\ell,\ d_\ell^2,\ \exp(d_\ell) - 1 \,\} \tag{9}$$

(uniform, linear, quadratic, and exponential), and their reversals replace d_ℓ by $\bar{d}_\ell$; reversing uniform gives uniform. Coefficients are normalized over all L layers in the primary comparison (Appendix Table 3).

The remaining two options are adaptive. First compute the unweighted activation loss $\mathcal{L}_u$ and its gradient with respect to each live self-replay decoder output tensor H_ℓ^s. Let

$$q_\ell = \mathrm{sg}\left[\mathrm{RMS}_{\text{nonpadding tokens, dimensions}} \left(\frac{\partial \mathcal{L}_u}{\partial H_\ell^s} \right) \right], \qquad \pi_\ell = \frac{q_\ell}{\sum_j q_j}. \tag{10}$$

Adaptive weighting sets $w_\ell = \pi_\ell$; reverse-adaptive sets $w_\ell = (1 - \pi_\ell)/(L - 1)$ for $L > 1$, a normalized complement. All-zero strengths fall back to uniform.

The adaptive variants use the current microbatch only, without smoothing. The static rules give positive coefficients at every layer.

3.5 Full Training Objective

The complete objective adds two optional terms to the activation loss of Equation 7:

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{GFT}} + \lambda_t \, \mathcal{L}_{\mathrm{CE}}^{\mathcal{F}} + \beta \, \mathcal{L}_{\mathrm{KL}}. \tag{11}$$

Here $\lambda_t \geq 0$ is a per-step CE coefficient set by the activation share $\alpha \in (0, 1]$, and $\beta \geq 0$ weights the KL regularizer. We study four settings that cross $\alpha \in \{1, 0.9\}$ (without or with extraction-failure CE) with $\beta \in \{0, 0.05\}$ (without or with student KL).

Extraction-failure CE. Let $\mathcal{F}$ be the pairs in a batch whose student generation contains no extractable answer span. These pairs have no answer-prediction positions, so they contribute nothing to $\mathcal{L}_{\mathrm{GFT}}$. $\mathcal{L}_{\mathrm{CE}}^{\mathcal{F}}$ is the token-level cross-entropy of the full gold response, computed only for pairs in $\mathcal{F}$; all other pairs contribute no CE. The share α is a *gradient share*, not a raw loss weight. At each optimizer step, using gradients accumulated over the full group, the CE coefficient is set to

$$\lambda_t = \min\left(\frac{1-\alpha}{\alpha} \, \frac{\|\nabla_\theta \mathcal{L}_{\mathrm{GFT}}\|_2}{\|\nabla_\theta \mathcal{L}_{\mathrm{CE}}^{\mathcal{F}}\|_2}, \; 10^3 \right) \tag{12}$$

and treated as a constant in the update, so that $\lambda_t \|\nabla_\theta \mathcal{L}_{\mathrm{CE}}^{\mathcal{F}}\|_2 = \frac{1-\alpha}{\alpha} \|\nabla_\theta \mathcal{L}_{\mathrm{GFT}}\|_2$ below the cap. If $\alpha < 1$ and a group has a CE gradient but no activation gradient, $\lambda_t = 1$; if $\mathcal{F}$ is empty, the CE term vanishes. Setting $\alpha = 1$ gives $\lambda_t = 0$ and recovers activation-only GFT. With $\alpha = 0.9$, CE supplies one tenth of the summed GFT and CE gradient norms.

We use this term for three reasons. Failed extraction occurs precisely when generation degrades, for example by repeating digits until the length cap, and this is where the activation loss loses its signal. Restricting CE to these pairs keeps the update dominated by activation alignment. Defining α as a gradient share makes it interpretable even though the two losses differ in scale by orders of magnitude.

KL regularization. Let p_{ref} be a frozen copy of the adapter. It is initialized from the starting adapter and replaced whenever validation accuracy strictly improves, so it is the best-validation checkpoint so far. The KL is measured on the current student's greedy rollouts y_i^s, the same generations used for the activation loss but unmodified (no answer replacement), for every pair i of the microbatch $\mathcal{B}$, including pairs excluded from $\mathcal{L}_{\mathrm{GFT}}$:

$$\mathcal{L}_{\mathrm{KL}} = \frac{1}{\sum_{i \in \mathcal{B}} |T_i|} \sum_{i \in \mathcal{B}} \sum_{t \in T_i} \mathrm{KL}\big(p_{\mathrm{ref}}(\cdot \mid x_i, y_{i,<t}^s) \;\big\|\; p_\theta(\cdot \mid x_i, y_{i,<t}^s)\big) \,, \tag{13}$$

where T_i contains the response positions of y_i^s up to and including the first end-of-sequence token. Student KL uses a fixed $\beta = 0.05$; no KL corresponds to $\beta = 0$.

We add this term because activation-only training shows abrupt post-peak collapse (Section 5.3). The collapse appears in the model's own generations as runaway repetition, so the KL is measured on those contexts. Anchoring to the best-validation checkpoint penalizes drift away from the best behavior observed so far, while still allowing activation-level adaptation.

3.6 Training and Inference

A persistent vLLM worker (Kwon et al., 2023) generates a batch using the current LoRA snapshot before each optimizer accumulation group. Model weights do not change within that group, so its microbatches share the same generation snapshot. Hugging Face Trainer handles optimization, checkpointing, and scheduling on the training GPU. When the CE or KL term is active, the GFT, CE, and KL gradients are accumulated in separate buffers over the group and combined with λ_t and β at the group boundary, before clipping and the optimizer step.

Pairs with missing answer spans, spans lost during replay truncation, or student generations ending at the token cap are excluded from the activation loss. Both correct and incorrect usable student responses are included. With $\alpha < 1$, pairs with a missing answer span receive the gold-response CE term instead. With $\alpha = 1$, an entirely invalid microbatch has zero activation loss; the KL term, if enabled, still applies. The KL term, when enabled, is computed with a frozen copy of the model on the same raw generated responses.

Validation and test require only normal prompt-to-output generation using vLLM. Neither activation comparison nor answer replacement is used at inference. The test run loads the checkpoint selected by validation accuracy. Appendix B specifies the update ordering.

4 Experimental Setup

Data and models. We use GSM8K (Cobbe et al., 2021) and the Intermediate Algebra and Precalculus subjects of MATH (Hendrycks et al., 2021),[1] with the datasets' worked solutions as training references. We hold out 10% of each training set for validation and report accuracy on the official test sets (1,319, 903, and 546 problems). We fine-tune `Qwen3-0.6B`, `Qwen3-1.7B`, and `Qwen3-4B` (Yang et al., 2025) with LoRA (Hu et al., 2022); all generation uses non-thinking, greedy decoding. Appendix Table 2 lists all data, optimization, token-limit, objective, and hardware settings.

Design space. We first train the two smaller models on GSM8K with all nine layer weightings (Section 3.4) and the four objective settings (Section 3.5), giving 36 runs per model. We then carry the three weightings that together cover the best GSM8K result of each model, uniform, linear, and quadratic (Section 5.4), to the remaining experiments, giving 12 runs per model and benchmark, each compared with an SFT baseline that minimizes cross-entropy on the full reference solution under the same data, prompt, and training protocol (Appendix Table 2).

Selection and scoring. A new best checkpoint is retained only for a strict increase in validation accuracy; ties keep the earlier checkpoint and count toward patience. Test accuracy is always measured at the best-validation checkpoint, on the full test set. On GSM8K, a numeric extractor reads the marked answer and otherwise falls back to the last number; on MATH, a prediction is correct only if its final `\boxed{}` answer is equivalent to the gold answer under `math-verify`. Unparseable predictions count as wrong.

5 Results

5.1 GFT versus SFT

Table 1 summarizes the three selected weightings on all three benchmarks. `Qwen3-4B` uses the same settings as `Qwen3-1.7B`. On GSM8K (Appendix Table 4), 32 of the 36 GFT runs of `Qwen3-0.6B` exceed the 63.15% SFT baseline, one matches it, and three fall below; the best reaches 66.72% (+3.57 points). For `Qwen3-1.7B`, 35 of 36 runs exceed the 80.82% baseline and one matches it; the best reaches 84.00% (+3.18). For `Qwen3-4B`, whose SFT baseline is already 91.89%, GFT still exceeds it in 7 of the 12 runs and matches it in one; the best reaches 92.65% (+0.76). Figure 3 (bottom) shows the same advantage during training: with student KL, every GFT run stays at or above SFT at every validation check.

On the MATH subjects, every GFT run exceeds its SFT baseline. For Intermediate Algebra the margins are 12–17 points for `Qwen3-0.6B`, 3–7 points for `Qwen3-1.7B`, and 4–9 points for `Qwen3-4B`. For Precalculus they are 10–22 points for `Qwen3-0.6B` and 9–23 points for `Qwen3-4B`. The Precalculus margins for `Qwen3-1.7B` are larger still (21–33 points); part of this gap comes from SFT outputs that repeat text until the 4,096-token cap (44% of its test outputs), and on outputs that finish, GFT still leads SFT by 13–20 points (16 on average).

5.2 How Often GFT Pulls Toward the Reference

The hinge in Eq. (6) pulls a layer toward the reference only when $s_\ell = u_\ell^\top \delta_\ell > 0$, that is, when moving z_ℓ^s toward z_ℓ^r along the filtered gap is predicted, to first order, to lower the NLL of the reference answer. Figure 3 (top; Appendix Figure 5 for all models) tracks the share of per-layer, per-example terms that pass this check. Over the first 750 steps of the 72 `Qwen3-0.6B` and `Qwen3-1.7B` runs, 46% of the terms are pulled on average (25–62% per run), and 50% near the best-validation checkpoint. The check therefore excludes more than half of the candidate alignment terms. Without it, each excluded term would move the student toward the reference along a direction in which the

[1] https://huggingface.co/datasets/DigitalLearningGmbH/MATH-lighteval

Table 1: Test accuracy (%) of SFT and GFT on three benchmarks, for the three weightings selected on GSM8K (Appendix E). α is the activation share of the objective ($\alpha = 0.9$ adds extraction-failure CE); Student denotes student KL with $\beta = 0.05$. Each cell is one training run evaluated at its best-validation checkpoint on the full test set (GSM8K 1,319; Intermediate Algebra 903; Precalculus 546). Bold marks the best GFT cell per model and benchmark.

	`Qwen3-0.6B`				`Qwen3-1.7B`				`Qwen3-4B`			
	$\alpha = 1$		$\alpha = 0.9$		$\alpha = 1$		$\alpha = 0.9$		$\alpha = 1$		$\alpha = 0.9$	
Weighting	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**
(a) GSM8K												
SFT	63.15				80.82				91.89			
Uniform	64.90	64.44	64.90	64.82	**84.00**	82.26	82.26	82.11	92.57	91.36	91.66	92.19
Linear	64.37	60.35	66.19	64.22	83.09	82.79	83.78	82.03	92.34	91.66	**92.65**	91.51
Quadratic	**66.72**	63.38	65.05	64.97	83.32	82.41	83.47	81.80	92.34	**92.65**	92.04	91.89
(b) MATH Intermediate Algebra												
SFT	17.39				51.50				62.24			
Uniform	32.89	32.00	29.79	**34.44**	**58.80**	55.15	54.82	56.48	69.32	70.32	**71.21**	69.44
Linear	30.12	30.56	30.79	30.23	57.36	56.59	57.03	56.26	69.88	68.66	69.44	69.55
Quadratic	31.78	30.45	31.23	31.56	56.70	57.03	55.26	57.48	70.10	65.89	69.55	69.88
(c) MATH Precalculus												
SFT	15.57				26.92				52.01			
Uniform	31.68	36.08	32.60	35.35	57.69	57.69	58.97	58.79	60.81	73.08	71.25	73.63
Linear	25.27	37.36	30.77	**37.73**	56.41	**60.07**	58.06	59.71	73.08	72.53	70.88	**74.73**
Quadratic	31.87	34.43	31.87	30.95	47.44	58.06	57.88	58.42	69.60	73.44	69.60	71.98

local gradient predicts a higher NLL for the reference answer. The check is also not a fixed mask. All four settings start at about 50%; by steps 500–750 the share falls to 37% in three settings but stays at 55% for $\alpha = 0.9$ without KL. The three weightings follow nearly identical curves, so the pulled share depends on the objective setting rather than on the layer weights.

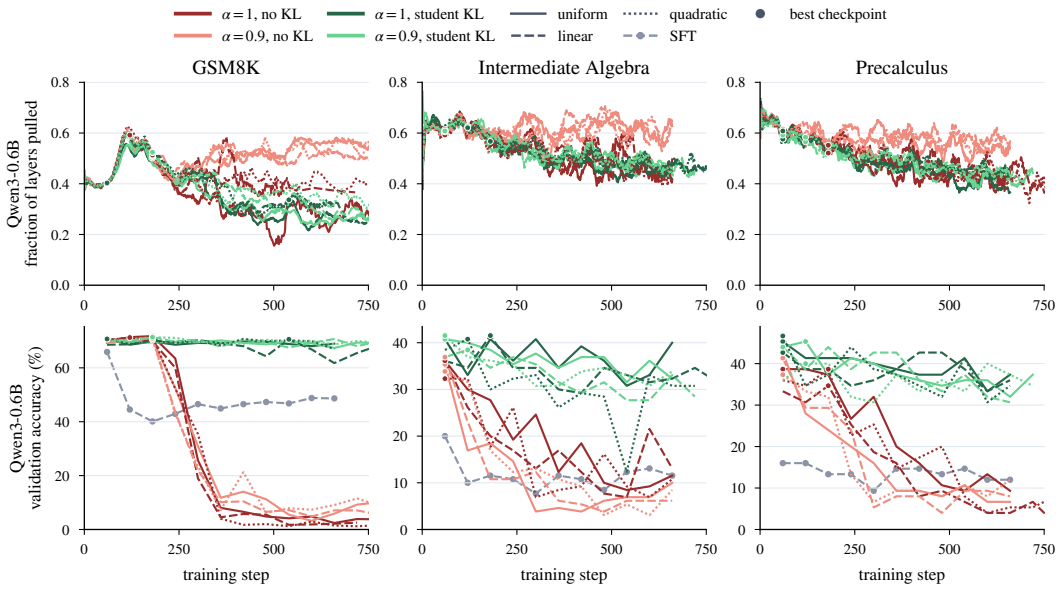

Figure 3: GFT on `Qwen3-0.6B` during the first 750 training steps. Top: fraction of per-layer, per-example alignment terms pulled toward the reference, i.e. with $u_\ell^\top \delta_\ell > 0$ in Eq. (6) (25-step moving average). Bottom: validation accuracy, with SFT in gray. Each panel shows the 12 runs of Table 1: color gives the objective setting and line style the weighting; dots mark best-validation checkpoints. Appendix Figures 4 and 5 show all models.

5.3 KL Regularization Prevents Post-Peak Collapse

In Table 1, student KL on average lowers best-checkpoint accuracy slightly on GSM8K (0.4–1.7 points) but raises it on Precalculus (2.7–4.6 points); its main effect appears after the peak. Appendix Table 5 follows each run of Table 1 after its best validation check. Without KL, validation accuracy later falls by more than 20 points in 47 of the 54 runs, on all three benchmarks and models, with drops of up to 85.4 points. The collapse often coincides with generation reaching the output cap: in 27 of these runs, more than half of the validation outputs are length-capped after the peak. Student KL removes these sudden collapses: 53 of 54 runs stay within 20 points of their peak. Figure 3 (bottom) and Appendix Figure 4 show the corresponding validation curves.

Without KL, validation-based selection still recovers strong checkpoints before the collapse; with KL, runs remain usable for the rest of training, which matters most when the stopping point is uncertain, as on smaller or harder datasets. Because the KL reference is itself the best-validation checkpoint, the regularizer anchors training to behavior that selection already found; it is primarily a stabilizer.

5.4 Layer Weightings

The nine weightings are close in accuracy on GSM8K (Appendix E): within each model, most runs lie within about two points of each other. Ranked by mean accuracy over the four settings (Appendix Table 4), every forward weighting of `Qwen3-1.7B`, which puts more weight on later layers, ranks above its reversed form; for `Qwen3-0.6B` the ordering is mixed. Quadratic ranks in the top three for both models and linear ranks first for `Qwen3-1.7B`.

6 Limitations

GFT provides a local, answer-focused training signal: it does not directly supervise the full reasoning trace or termination, and its one-sided loss leaves activation directions that do not affect the answer free. Each update also involves student generation, two replays, and gradient extraction, plus a frozen-reference pass when the KL term is used, so a step-matched comparison with SFT is not compute-matched. Our study covers one model family up to 4B parameters on mathematical reasoning with a single training seed, and the three weightings were selected on GSM8K and transferred unchanged to the MATH subjects and a larger model.

7 Conclusion

Gradient-guided Fine-Tuning learns from reference solutions through an answer-sensitive activation objective computed inside the student itself. Its full objective combines this activation loss with a CE term restricted to extraction failures and a KL regularizer anchored to the best-validation checkpoint. GFT exceeds a protocol-matched SFT baseline in 67 of 72 GSM8K configurations for `Qwen3-0.6B` and `Qwen3-1.7B` and in 7 of 12 for `Qwen3-4B`, where SFT is already strong, and every MATH run exceeds its SFT baseline. Student KL removes the post-peak collapse seen in 33 of 36 unregularized GSM8K runs, at little or no cost in best-checkpoint accuracy.

In future work, we plan to extend the objective beyond answer positions and a single answer-sensitive direction to further activation directions, reduce its per-step overhead, and evaluate it on larger models, other domains, and additional seeds.

AI Use Statement

Generative AI tools were used to assist with drafting and editing the manuscript, format LaTeX content, and support the design of scientific figures. All AI-assisted suggestions and outputs were reviewed by the authors for correctness, relevance, and originality. The authors take full responsibility for the final methodology, experiments, claims, and presentation.

Reproducibility Statement

The objective, gradient stops, and layer-weighting rules are specified in Section 3; Section 4 gives data, optimization, and selection settings. Appendix A documents extraction and activation conventions. Code and configurations for all reported runs are available via the link in the abstract.

Ethics Statement

This study uses publicly available models and mathematical reasoning datasets and involves no human subjects or personal data. Its accuracy measurements do not establish reliability for educational or other high-stakes use.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.13649.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2015.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the International Conference on Machine Learning*, pp. 794–803, 2018. URL https://arxiv.org/abs/1711.02257.

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. arXiv:2110.14168.

Kawin Ethayarajh. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 55–65, 2019.

Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*, 2024.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

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 *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://arxiv.org/abs/2103.03874.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14852–14882, 2023.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

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://arxiv.org/abs/2106.09685.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, 2016. URL https://arxiv.org/abs/1606.07947.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. URL https://arxiv.org/abs/2309.06180.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, 2023.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1773–1781, 2023.

Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1702.01417.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6550.

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.

Khouloud Saadi and Di Wang. What should feature distillation transfer in LLMs? A task-tangent geometry view, 2026. URL https://arxiv.org/abs/2507.10155v3. arXiv:2507.10155, version 3.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, 2019. URL https://arxiv.org/abs/1910.01108. arXiv:1910.01108.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2312.06585.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 4322–4331, 2019.

William Timkey and Marten van Schijndel. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 4527–4546, 2021. URL https://arxiv.org/abs/2109.04404.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, 2020.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1908.04319.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. ReFT: Representation finetuning for language models. In *Advances in Neural Information Processing Systems*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388. arXiv:2505.09388.

Shenzhi Yang, Guangcheng Zhu, Bowen Song, Haobo Wang, Mingxuan Xia, Xing Zheng, Yingfan Ma, Zhongqi Chen, Weiqiang Wang, Junbo Zhao, and Gang Chen. OPRD: On-policy representation distillation, 2026. URL https://arxiv.org/abs/2606.06021v4. arXiv:2606.06021, version 4.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, 2022.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A Implementation and Reproducibility

Table 2 lists the full training, generation, and evaluation configuration for each benchmark.

Table 2: Training, generation, and evaluation configuration. A value spanning several columns applies to all of them. Validation cadence is in optimizer steps and patience in validation checks.

Setting	GSM8K	Intermediate Algebra	Precalculus
Data			
Source	GSM8K	MATH (MATH-lighteval)	
Train / validation / test	6,725 / 748 / 1,319	1,165 / 130 / 903	671 / 75 / 546
Training references	dataset worked solutions		
Split seed / training seed	42 / 1333		
Models and LoRA			
Students	`Qwen3-0.6B` / `Qwen3-1.7B` / `Qwen3-4B`		
Decoder blocks	28 / 28 / 36		
Precision	BF16		
LoRA rank / `lora_alpha` / dropout	16 / 32 / 0.15		
LoRA modules	query, key, value, output, gate, up, and down projections		
Optimization			
Optimizer	AdamW, learning rate 2×10^{-5}, weight decay 0.01		
Schedule / gradient clipping	linear with 3% warmup / norm 1.0		
Maximum epochs	50	25	25
Effective batch	32	8	8
Steps per epoch	211	146	84
Other	dynamic padding, gradient checkpointing		
Validation and selection			
Validation every / patience	60 steps / 10 checks		
Best checkpoint	strict increase in validation accuracy		
Token caps			
CE input	768	2,816	2,816
Prompt	320	1,024	1,024
Generated tokens	1,024	4,096	4,096
Activation replay	1,536	5,120	5,120
vLLM context	1,536	5,120	5,120
GFT objective			
Layers	all decoder blocks, before the final normalization		
Pooling	answer-prediction positions		
Common-direction filter	top-1 direction, 128 calibration pairs, frozen		
Activation share α / CE cap	$\{1, 0.9\}$ / $\lambda_t \leq 10^3$		
KL weight β / reference	$\{0, 0.05\}$ / best-validation checkpoint, student context		
Answer span	last boxed numeric answer	content of the final box	
Decoding and scoring			
Decoding	non-thinking, greedy (temperature 0), batch size 8		
Scoring	numeric match	`math-verify` on the final box	
Hardware	NVIDIA A100 (80 GB) and H200 (141 GB) GPUs		

Activation conventions. Layer numbering in the implementation is zero-based. For a 28-block model, the last block is 27, and its output is also taken before the model-level final normalization. Reference activations, the common-direction basis, gradient directions, scale factors, and layer weights are detached. Only the self activation branch provides the gradient of the GFT loss.

The filtered gradient is usable only when $\|P_\ell g_\ell\|_2 > 10^{-6} \max(\|g_\ell\|_2, 10^{-12})$. Its norm denominator is clamped at 10^{-12}. The scale denominator is clamped at 10^{-6}.

Answer spans and scoring are different operations. In the reported GFT runs, the activation span locator uses the last marked answer requested by the prompt: on GSM8K, the numeric prefix of the last `\boxed{}` content, with a fallback to GSM8K's `####` answer marker; on MATH, the

content of the final box. It does not use an arbitrary reasoning number when a marked span is missing; such pairs form the extraction-failure set $\mathcal{F}$ of Section 3.5. The GSM8K validation/test scorer instead first looks for a boxed and then a ####-marked answer after text that repeats the opening words of the question; otherwise it takes the most frequent of the last 20 boxed answers, then the most frequent of the last 20 ####-marked answers, and finally the last number in the output. Numeric equality is used for integer GSM8K answers.

Prompt. The GFT runs use the same system message for training and evaluation:

> You are a mathematics expert. Solve the problem step by step, showing clear reasoning, and put your final answer in `\boxed{...}`.

The SFT baselines use the same message. The model's chat template is applied with thinking disabled. Only the question and system context are available during free generation.

Update boundaries. Student responses are generated for an optimizer accumulation group and split into its microbatches. Each microbatch computes its valid-pair mean; gradients are accumulated over the group, including the final partial group, in separate GFT, CE, and KL buffers when those terms are active. This is not a global valid-pair-weighted reduction across the entire accumulation group. Adaptive layer weights are measured separately for each microbatch before applying those weights.

B Update Algorithm and Design Rationale

Initialization. Initialize the student and trainable LoRA adapters, the optimizer and scheduler, and a persistent generation worker. Before optimization, fit the common-direction basis using only usable training calibration pairs. The saved checkpoint state keeps this basis fixed across checkpoint resume. No validation or test activation is used to calibrate the GFT loss.

1. Take the next optimizer accumulation group of training examples. Synchronize the current adapter snapshot to the vLLM worker and greedily generate the group's student responses in a batch.
2. Partition the group into training microbatches. The model weights remain unchanged until this accumulation group is complete.
3. Locate reference and student answer spans. Construct the student-prefix/reference-answer sequence and check that both answer spans survive tokenization and replay limits. Exclude unusable pairs.
4. Replay the reference branch without gradients and the self branch with gradients. Disable dropout in both. Capture all decoder-block outputs before final normalization and pool prediction-region activations.
5. Evaluate answer NLL on the self branch. Extract its activation gradients with the graph retained for the subsequent activation-loss backward pass, then detach those gradients. Do not accumulate this NLL directly into parameter gradients as a training objective.
6. Form the filtered, scaled gap and the normalized filtered direction. Compute the squared positive projection at every layer.
7. For a static weighting rule, normalize its depth coefficients. For an adaptive rule, first measure the gradients of the unweighted activation loss with respect to live decoder outputs; detach and normalize their strengths.
8. Reduce the weighted activation loss over usable pairs. With $\alpha < 1$, add the gold-response CE of extraction-failure pairs, rescaled to its gradient share (Eq. 12). If KL is enabled, add β times the KL to the frozen best-validation reference on the raw student-generated contexts. Accumulate each term's gradient over the group and combine them at the group boundary.
9. At the group boundary, clip accumulated gradients and perform the optimizer and scheduler step.

10. At scheduled validation steps, synchronize the current adapter and generate normally. Save a new best checkpoint only for a strict accuracy increase and, if KL is enabled, replace the KL reference p_{ref} with it. After stopping, test that best checkpoint with normal generation.

Why two teacher-forced passes do not remove generation. A known sequence can be processed in one causally masked forward pass, with a distinct activation at each prediction position. However, a prompt alone does not specify the student's future output tokens. Generating those tokens still requires autoregressive decoding. Replaying the gold sequence twice through the same deterministic current model would produce the same activations, not the student's own reasoning trajectory. The generation worker accelerates this necessary decoding stage; it does not replace the replay or supply the training gradient.

Why answer replacement is needed. Comparing a state conditioned on a gold answer of 16 with one conditioned on a student answer of 17 can largely measure token identity. Our replacement evaluates the probability of 16 after the student's own reasoning and compares its prediction-region states with the reference reasoning's states. This does not claim that the student's reasoning was correct, and subsequent answer tokens are still teacher-forced. The construction controls the target content, not every difference between the two contexts.

Why a moving reference is not a second trained teacher. The reference output text is fixed for a training example, but its activation is evaluated using the current student's weights. Detaching that branch stops its contribution to the current derivative; it does not freeze its value for future updates. In the LoRA runs, the reference forward uses the active adapters just as the self forward does. Only the adapters are optimized, but their effect propagates through subsequent attention and MLP computations.

Why loss scale alone is insufficient. A scalar loss value and its parameter-gradient norm are different quantities. A small projected loss can produce a meaningful update; an exactly zero loss can instead mean inactive projections or missing valid pairs. This is why α in Eq. (12) balances gradient norms rather than loss values. Layer weights sum to one; they are neither CE/activation mixing fractions nor learned guarantees of equal optimization influence.

C Validation Trajectories

Figure 4 shows validation accuracy during training for the 108 runs of Table 1.

D Answer-Sensitivity Check Across All Models

Figure 5 extends Figure 3 to all 108 runs of Table 1, including `Qwen3-4B`.

E GSM8K Layer-Weighting Study

Table 3 lists the nine weighting rules of Section 3.4. Table 4 reports all nine weightings on GSM8K for the four objective settings. It is the basis for selecting the three weightings of Table 1. Averaged over the nine weightings, settings without KL obtain the highest best-checkpoint accuracy: 65.16% ($\alpha = 1$) and 64.32% ($\alpha = 0.9$) for `Qwen3-0.6B`, and 82.71% and 82.65% for `Qwen3-1.7B`; student KL lowers these averages by 0.2 to 1.7 points. We also evaluated a dual-context KL, which additionally applies the KL on the gold response; it gave the same stability as student KL with similar accuracy and additional cost, so we use student KL throughout. Table 5 reports the post-peak validation behavior of the runs of Table 1 (Section 5.3).

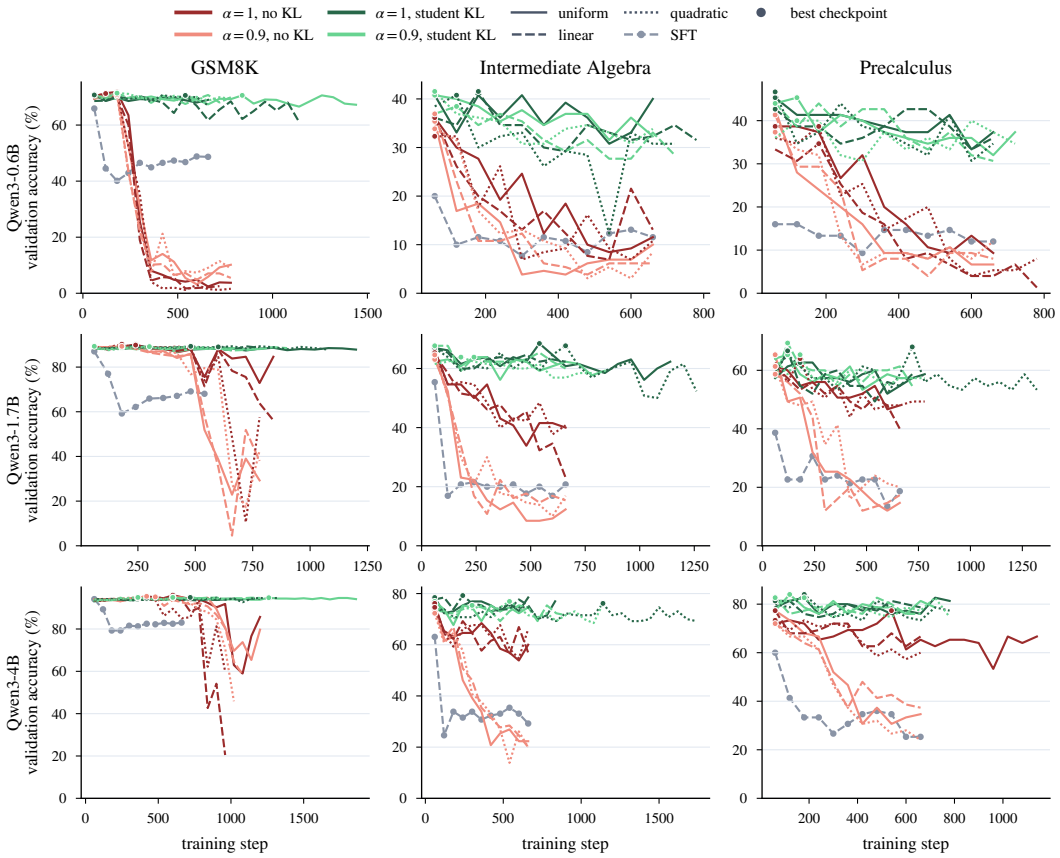

Figure 4: Validation accuracy during training. Each panel shows the 12 runs of Table 1: color gives the objective setting and line style the weighting; dots mark best-validation checkpoints. Gray dashed lines show SFT.

Table 3: The nine layer-weighting rules. Static rules set $w_\ell = b_\ell / \sum_j b_j$ with relative depth $d_\ell = (\ell+1)/L$ and its reversal $\bar{d}_\ell = (L-\ell)/L$; reversing uniform gives uniform. Adaptive rules use the normalized gradient strengths π_ℓ of Section 3.4.

Rule	**Type**	**Weight w_ℓ**
Uniform	static	$b_\ell = 1$
Linear	static	$b_\ell = d_\ell$
Quadratic	static	$b_\ell = d_\ell^2$
Exponential	static	$b_\ell = \exp(d_\ell) - 1$
Reverse linear	static	$b_\ell = \bar{d}_\ell$
Reverse quadratic	static	$b_\ell = \bar{d}_\ell^2$
Reverse exponential	static	$b_\ell = \exp(\bar{d}_\ell) - 1$
Adaptive	adaptive	$w_\ell = \pi_\ell$
Reverse adaptive	adaptive	$w_\ell = (1 - \pi_\ell)/(L-1)$

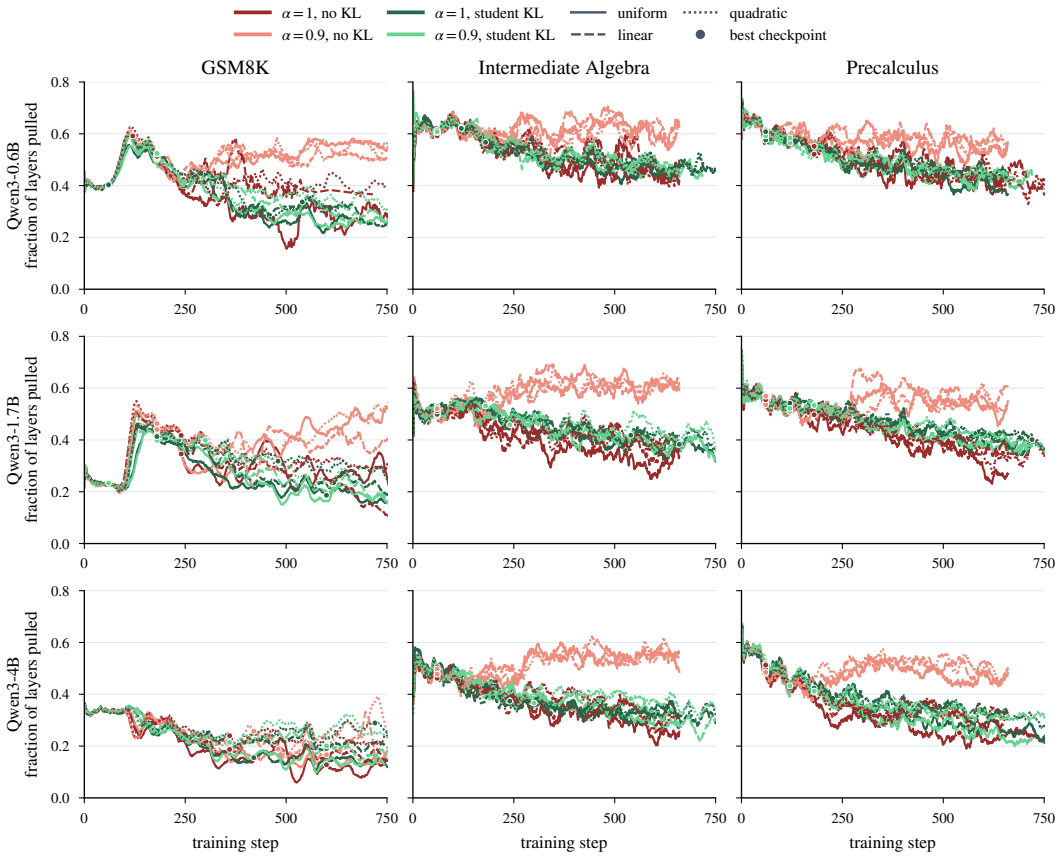

Figure 5: Fraction of per-layer, per-example alignment terms pulled toward the reference, as in Figure 3, for all three models (25-step moving average, first 750 steps). Three best checkpoints (steps 840–1260) lie beyond the plotted range.

Table 4: GSM8K test accuracy (%) for all nine layer weightings and the four objective settings. Columns as in Table 1. Bold marks the best cell per model; Mean averages each row over the four settings. Q3 denotes Qwen3.

	Q3-0.6B					Q3-1.7B				
	$\alpha = 1$		$\alpha = 0.9$			$\alpha = 1$		$\alpha = 0.9$		
Weighting	**No KL**	**Student**	**No KL**	**Student**	**Mean**	**No KL**	**Student**	**No KL**	**Student**	**Mean**
SFT			63.15					80.82		
Uniform	64.90	64.44	64.90	64.82	64.76	**84.00**	82.26	82.26	82.11	82.66
Linear	64.37	60.35	66.19	64.22	63.78	83.09	82.79	83.78	82.03	82.92
Quadratic	**66.72**	63.38	65.05	64.97	65.03	83.32	82.41	83.47	81.80	82.75
Exponential	64.37	64.29	64.22	63.53	64.10	83.24	83.17	83.17	81.73	82.83
Adaptive	63.53	63.00	63.76	63.61	63.48	82.49	82.18	82.03	82.87	82.39
Rev. linear	65.28	63.61	62.47	64.06	63.86	83.78	80.89	82.41	81.58	82.16
Rev. quadratic	65.05	63.53	63.68	63.53	63.95	81.65	82.11	82.79	82.26	82.20
Rev. exponential	65.50	64.82	64.06	63.15	64.39	82.03	82.03	81.50	82.49	82.01
Rev. adaptive	**66.72**	64.14	64.52	64.82	65.05	80.82	82.49	82.41	82.18	81.97
Mean	65.16	63.51	64.32	64.08	64.27	82.71	82.26	82.65	82.12	82.43

Table 5: Post-peak validation behavior of the runs of Table 1. Each cell is the largest drop in validation accuracy after the run's best validation check (best minus lowest later accuracy, in points). Bold marks a collapse (drop above 20 points); † marks runs with more than half of the validation outputs hitting the generation cap after the peak. Q3 denotes Qwen3.

	Q3-0.6B				Q3-1.7B				Q3-4B			
	$\alpha = 1$		$\alpha = 0.9$		$\alpha = 1$		$\alpha = 0.9$		$\alpha = 1$		$\alpha = 0.9$	
Weighting	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**	**No KL**	**Student**
(a) GSM8K												
Uniform	**69.4**†	2.8	**67.1**†	4.0	17.2	1.6	**66.4**†	1.9	**37.0**	0.8	**29.7**	0.8
Linear	**69.5**†	9.5	**66.4**†	3.9	**33.7**†	1.9	**85.4**†	1.7	**74.3**†	1.5	**36.8**	0.7
Quadratic	**70.2**†	3.7	**64.7**†	2.7	**79.7**†	1.5	**72.1**†	1.7	**34.8**	2.3	**49.6**	0.8
(b) MATH Intermediate Algebra												
Uniform	**27.7**†	10.0	**31.5**	9.2	**32.3**†	12.3	**54.6**†	7.7	**20.8**	6.2	**55.4**	9.2
Linear	**29.2**†	12.3	**33.1**†	10.8	**41.5**†	6.9	**53.9**†	8.5	20.0	9.2	**53.9**	6.2
Quadratic	**25.4**†	**28.5**†	**30.8**†	10.8	**27.7**†	17.7	**54.6**†	6.9	20.0	6.9	**58.5**	8.5
(c) MATH Precalculus												
Uniform	**29.3**†	12.0	**36.0**	13.3	14.7	10.7	**53.3**	12.0	**24.0**	6.7	**46.7**	8.0
Linear	**33.3**†	9.3	**37.3**†	13.3	**21.3**	17.3	**46.7**	14.7	14.7	6.7	**34.7**	6.7
Quadratic	**34.7**†	16.0	**30.7**	9.3	20.0	14.7	**45.3**	10.7	16.0	10.7	**48.0**	8.0