OPTD: ON-POLICY TRUNCATED DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy representation distillation (OPRD) trains a student language model on its own samples to match its teacher's hidden states at every layer, giving every layer's squared error the same weight. Hidden states grow in scale with depth, so this equal weighting sends most of the gradient on the hidden states to the last few layers. Splicing the student's hidden state into the teacher one layer at a time shows that errors in these layers damage the teacher's prediction far less than the loss implies. Weighting layers by the measured damage improves accuracy, yet a shuffled placebo that assigns the same weights to random layers improves it even more. Every reweighting scheme we test, this placebo and a uniform one on normalised errors included, starves the last layers of gradient. On-policy truncated distillation (OPTD) keeps only that shared property by dropping the last four layers from OPRD's sum. With a public teacher trained from its student by reinforcement learning, OPTD improves accuracy over OPRD by 3.97 points on problems from four mathematics and science benchmarks, beats every reweighting scheme, and outscores the teacher on all four.

1 INTRODUCTION

On-policy distillation trains a student on sequences it samples, with the teacher scoring every token (Agarwal et al., 2024; Gu et al., 2024). Training on teacher transcripts instead never shows the student the states its own mistakes lead to (Ross et al., 2011). For mathematical reasoning, where reinforcement learning (RL) drives much recent progress (Shao et al., 2024; DeepSeek-AI, 2025), on-policy distillation can outperform RL at a fraction of its compute (Yang et al., 2025). On-policy representation distillation (OPRD) moves this supervision to the hidden states. Feature distillation has long matched intermediate layers (Romero et al., 2015; Sun et al., 2019; Jiao et al., 2020), and OPRD matches every layer, summing the per-layer squared errors with equal weight (Yang et al., 2026).

We study DeepSeek-R1-Distill-Qwen-1.5B as the student and JustRL-DeepSeek-1.5B, the public RL checkpoint trained from it, as the teacher (DeepSeek-AI, 2025; He et al., 2025). The two share one 28-layer architecture, so every student layer has a teacher counterpart. Hidden states grow in scale with depth, as do the student's errors, and OPRD's equal weights ignore this. The last four layers receive 79% of OPRD's gradient on the student's hidden states, as Figure 1 shows. We propose on-policy truncated distillation (OPTD), which leaves these layers out of the sum and improves accuracy over OPRD by 3.97 points on problems from four mathematics and science benchmarks.

Equal weights would be harmless if deep-layer errors also cost the teacher the most. We test this with a training-free splice, a form of activation patching (Vig et al., 2020; Meng et al., 2022). The student's hidden state replaces the teacher's at one layer, the teacher's remaining layers run, and the damage at that layer is the KL divergence between the clean and spliced next-token distributions. The damage levels off in the later layers, which appear to sharpen a prediction already formed (Geva et al., 2022; Belrose et al., 2023), while the loss keeps growing. The last four layers hold only 24% of the summed damage, against 79% of the gradient.

Reweighting the layers by their damage, or removing the scale growth, improves on OPRD. We also train two placebos whose weights carry no information about depth. The uniform placebo weights all layers equally on a common scale, and the shuffled placebo gives the damage weights to random layers. On the pooled problems none of these schemes outperforms either placebo, and the shuffled placebo beats the damage weights it reuses. The gain thus seems to come from what the schemes share, and Table 2 shows that each of them starves the last layers of gradient.

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

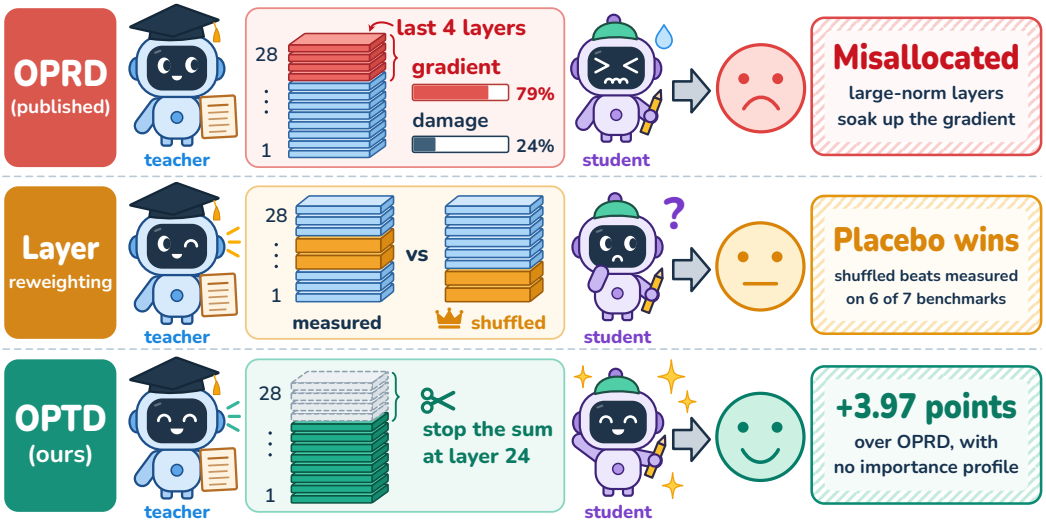

Figure 1: The published loss of Equation 1 gives the last 4 of 28 layers 79% of the gradient but only 24% of the damage. Reweighting by importance is beaten by its shuffled placebo on 6 of 7 benchmarks. OPTD stops the sum at layer 24 and gains +3.97 points on the pooled benchmarks.

OPTD builds in this shared property directly, and its only hyperparameter is how many layers to drop. It outperforms every reweighting scheme on the pooled problems and has the highest mean accuracy over all seven benchmarks, above its RL-trained teacher. The splice test needs only forward passes and can vet a new pair before training. Across eight public model pairs, the loss tracks the hidden-state norm in seven, and in the eighth its last-four share already matches the damage. Sections 3, 4, 5, and 6 give the diagnosis, the reweighting study, OPTD, and its evaluation.

2 Related work

Distillation signals. Model compression (Buciluǎ et al., 2006) and soft-target distillation (Hinton et al., 2015; Sanh et al., 2019) preceded sequence-level distillation on teacher transcripts (Kim & Rush, 2016), which can expose the student to a distribution unlike its test-time states (Ross et al., 2011). On-policy methods address this mismatch by scoring student trajectories with the teacher (Agarwal et al., 2024) or using reverse KL (Gu et al., 2024). Reasoning models add reinforcement learning with rule-based or learned verifiers (Shao et al., 2024; DeepSeek-AI, 2025; Lightman et al., 2024), while other methods supervise elicited chains, verified traces, or alternative generations (Wei et al., 2022; Zelikman et al., 2022; Magister et al., 2023; Ho et al., 2023; Fu et al., 2023; Hsieh et al., 2023; West et al., 2022; Muennighoff et al., 2025). These methods build on reward-driven alignment pipelines (Ziegler et al., 2019; Ouyang et al., 2022; Rafailov et al., 2023). These approaches choose *which tokens or trajectories* to supervise; we ask *which layers* should be supervised.

Matching internal representations. FitNets matched intermediate activations (Romero et al., 2015); later work extended this to selected layers, attention maps, relations, feature statistics, self-distillation, and full layer correspondence (Sun et al., 2019; Zagoruyko & Komodakis, 2017; Park et al., 2019; Tian et al., 2020; Heo et al., 2019; Zhang et al., 2019; Jiao et al., 2020). FitNets matches one intermediate layer, patient distillation selects and normalises several layers, and TinyBERT applies LayerNorm to the representations it transfers. These design choices reduce the risk that a layer dominates the loss merely because its representation has a larger scale. They do not, however, answer whether a layer's error matters to the final prediction. A pre-norm decoder offers no protection for its accumulated residual stream, so equal coefficients can still induce unequal effective weights even when every residual branch is internally normalised.

Layers differ in function. Transformer layers differ in computation and functional importance (Vaswani et al., 2017); later blocks can sometimes be removed with little behavioural degradation

(Gromov et al., 2025; Men et al., 2025), reflecting excess capacity (Frankle & Carbin, 2019; Sanh et al., 2020; Ma et al., 2023; Ashkboos et al., 2024). Block-pruning studies delete late blocks from the model, whereas OPTD keeps every block and drops only the late ones' loss terms. Mechanistic studies view feed-forward blocks as key-value stores (Geva et al., 2021; 2022), while the residual stream carries information between blocks and can be decoded at intermediate depths (Elhage et al., 2021; Belrose et al., 2023). Probes and similarity measures are observational (Alain & Bengio, 2017; Tenney et al., 2019; Raghu et al., 2017; Kornblith et al., 2019); our activation-patching splice is causal (Vig et al., 2020; Meng et al., 2022). It asks a more specific question than a probe: after replacing the teacher's state at one layer with the student's state, how much does the teacher's downstream output move? The answer therefore depends on the computation of every remaining layer, which a classifier reading the patched layer cannot capture.

Norms, curvature, and controls. LayerNorm and RMSNorm act inside residual branches, leaving the accumulated stream free to grow with depth (Ba et al., 2016; Zhang & Sennrich, 2019; Xiong et al., 2020); such growth is also an inductive bias in transformer dynamics (Merrill et al., 2021). Measuring an update by its output effect relates to natural-gradient geometry and Fisher-like importance (Amari, 1998; Martens, 2020; Kirkpatrick et al., 2017). Our splice is a non-parametric, causal analogue: it propagates the intervention through the teacher's remaining layers without fitting a probe or an auxiliary predictor. The shuffled placebo matches the treatment except for the layer-to-weight assignment and uses the same paired-resampling protocol (Efron & Tibshirani, 1993; Koehn, 2004; Dror et al., 2018).

These strands of work motivate two separate questions. Representation matching asks whether the student encodes a teacher-like state, whereas activation patching asks what the rest of the model does with a substituted state. The former is a training signal; the latter is a causal diagnostic. Our experiments use both roles explicitly: the hidden-state MSE defines the objective under study, and the splice KL evaluates whether that objective allocates its layer budget where downstream predictions remain sensitive. This distinction also places the norm-matched perturbation on the diagnostic side, as a control for the splice.

3 THE LOSS AND ITS IMPLICIT LAYER WEIGHTING

This section asks at which depths the published loss places its gradient, and what an error at each depth costs the teacher's prediction.

Setup. We study OPRD's representation-only objective (Yang et al., 2026), which replaces the output-space term in on-policy distillation (Li et al., 2026) with hidden-state matching. Let h_ℓ^S and h_ℓ^T be the student and teacher residual streams at layer ℓ on response positions of a student rollout:

$$\mathcal{L}_{\text{rep}} \;=\; \frac{1}{L}\sum_{\ell=1}^{L} \big\|h_\ell^S - h_\ell^T\big\|^2_{\text{mean}}, \qquad \big\|x\big\|^2_{\text{mean}} \;=\; \tfrac{1}{D}\textstyle\sum_d x_d^2, \tag{1}$$

We write m_ℓ for the per-layer mean squared error inside the sum. For the primary student–teacher pair described in Section 6, $L = 28$ and $D = 1536$. Gradients are stopped through the teacher states. Thus, every layer has the same explicit coefficient, $1/L$. Equal coefficients are equal in effect only if the layers' errors share a scale, and the rest of this section shows that they do not.

Gradient allocation. Differentiating Equation 1 gives $\partial\mathcal{L}/\partial h_\ell^S = \frac{2}{LD}(h_\ell^S - h_\ell^T)$, so direct gradient energy is proportional to the squared error and grows with residual-stream scale. That scale in turn rises with depth, and the teacher's mean squared residual norm grows by a factor of 273 from the first transformer layer to the last. In the primary pair, the per-layer error rises from 0.0023 at layer 1 to 41.75 at layer 28; over 256 rollouts, the final four layers receive 79.3% of direct hidden-state gradient energy, versus 0.03% for the first four. This is a property of the loss before the optimiser, model Jacobian, or parameter sharing is taken into account: it describes the energy of the signal that each layerwise term sends directly into the student residual stream. A large residual norm can therefore make a layer important to the objective even when the teacher's subsequent computation would attenuate the corresponding discrepancy.

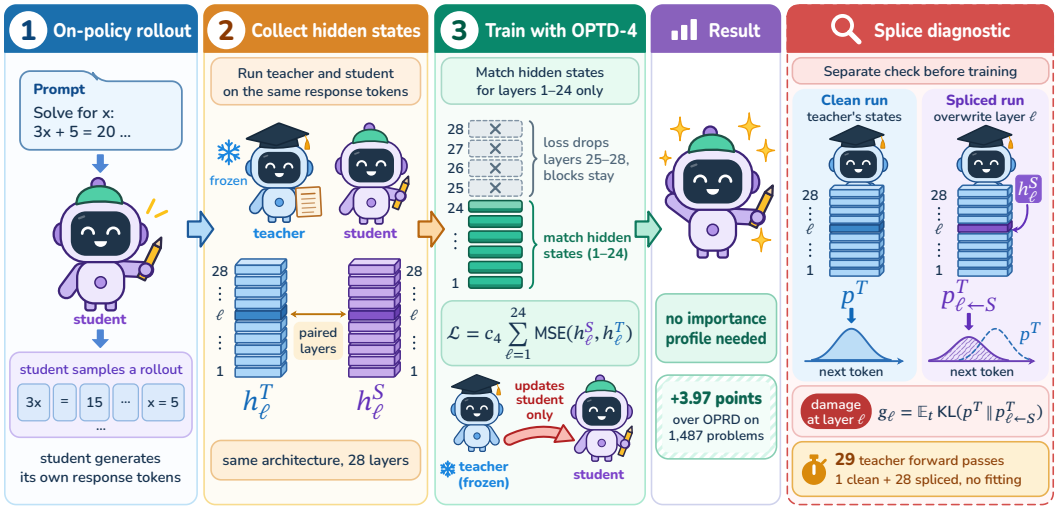

Figure 2: OPTD from rollout to result. Teacher and student run on the student's own response, and OPTD-4 matches layers 1 to 24 only, scaled by c_4 of Equation 3. It needs no importance profile and gains 3.97 points over OPRD on 1,487 problems from four benchmarks. The splice measures g_ℓ of Equation 2 once per pair before training, and the OPTD loss never uses it.

Table 1: Splice controls on the primary pair. Entries are mean output KL in nats; the Gaussian control is matched to the student's displacement norm at every token.

Splice at layer	1–8	9–16	17–27	28
Teacher's own state	0	0	0	0
Student's state	0.0172	0.1916	0.3184	0.3642
Norm-matched Gaussian	0.00183	0.0177	0.0314	0.1589
Student / Gaussian	9.4	10.8	10.1	2.3

Measuring functional cost. We ask how much the teacher's output changes when its layer-ℓ state is replaced by the student's. We run the teacher on student-generated tokens, repeat the pass with that substitution, run the remaining layers and unembedding, and measure the KL divergence between clean and spliced distributions (Figure 2):

$$g_\ell \;=\; \mathbb{E}_t\Big[\,\mathrm{KL}\big(\,p^T(\cdot \mid x_{<t}) \;\big\|\; p^T_{\ell \leftarrow S}(\cdot \mid x_{<t})\,\big)\Big]. \tag{2}$$

g_ℓ is an output-space KL divergence measured in nats, providing a common unit across layers. It therefore quantifies how the teacher's remaining computation transforms a layerwise representation error into an output-space deviation. We call g_ℓ the damage at layer ℓ. It is near zero when the teacher's later layers absorb the student's error, and it grows as that error changes the teacher's next-token prediction. As an identity check, self-splicing the teacher state yields $g_\ell = 0$ at every tested layer. Because the same student state is used in the clean and spliced passes, token positions, prompts, and decoding choices are held fixed. The measurement thus isolates the downstream consequence of the representation error from any change in the sampled trajectory.

Norm-matched control. Let $\delta_\ell = h_\ell^S - h_\ell^T$. We replace it with a Gaussian vector z_ℓ rescaled token-wise to the same norm. Through layer 27, the student displacement induces roughly ten times more output KL than this control: ratios are 9.4, 10.8, and 10.1 over layers 1–8, 9–16, and 17–27, respectively. At layer 28, only final normalisation and the nearly linear unembedding remain, so the ratio falls to 2.3. If the splice effect were explained only by the magnitude of a perturbation, the norm-matched Gaussian would produce a comparable output shift. The tenfold separation through layer 27 instead shows that the student's displacement has a structured direction that the teacher's downstream computation responds to differently from a random vector of the same size.

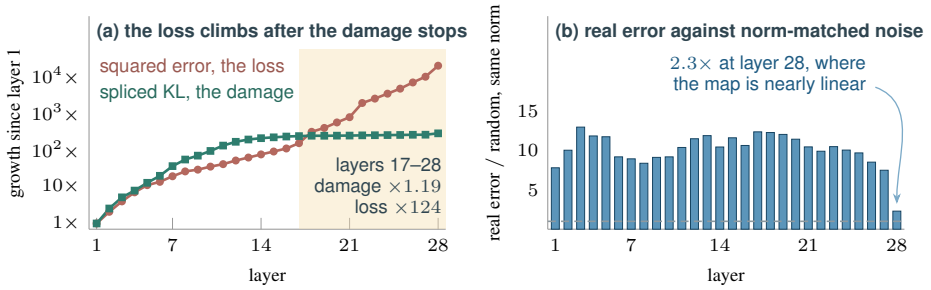

Figure 3: Loss against damage. (a) Through layer 17 the two rise together; over the shaded region the damage is flat and the loss grows by a factor of 124. (b) On 12 sequences, the student's displacement up to layer 27 costs the teacher about ten times more than a Gaussian perturbation rescaled to the same norm at every token; the dashed line at 1 is what a pure norm meter would read.

The diagnosis. Figure 3 shows that through layer 17 the loss and output KL rise by $\times 148$ and $\times 230$, respectively, but from layers 17–28 they rise by $\times 124$ and only $\times 1.19$. After normalisation, the final four layers receive 79.3% of raw-MSE mass but 23.5% of splice-KL mass: a 3.37-fold over-allocation ($[3.35, 3.38]$ bootstrap interval). A flat profile and the norm-matched control assign only 14.3% and 41.7%, respectively. Thus the discrepancy lies in allocation, a mismatch between where the objective places its layer budget and where the teacher's output remains sensitive to the student's error. The splice profile supplies the functional reference needed to decide whether a proposed normalisation or reweighting fixes that mismatch.

The splice quantity should not be confused with the full parameter-gradient norm. It is measured at the hidden-state interface, before the student's Jacobian, optimiser preconditioning, or shared parameters redistribute the signal. This makes it useful for diagnosing the layerwise objective itself, while leaving the optimiser-dependent route from a hidden-state error to a weight update for the training experiments. In particular, a layer can have a large direct MSE contribution and still have a small downstream effect, which is precisely the mismatch that truncation targets.

Consistency across evaluation domains. On generations from four benchmarks, including two not targeted during training, the final-four raw-MSE share is 75.6%–78.5%, or a $2.62\times$–$2.78\times$ over-allocation relative to splice KL.

Generality across model pairs. Across eight pairs from five public model families, the log per-layer MSE–log squared-norm correlation is at least 0.983 in seven pairs. On the primary pair, adjacent-layer changes correlate at $+0.56$ for log MSE but only $+0.05$ for splice damage. Qwen3-4B is a counterexample: its norm peaks mid-network, its correlation is 0.841, and the final-four shares of raw MSE and splice KL are nearly identical (27.5% and 28.0%). A single splice sweep therefore distinguishes norm-driven misallocation from a loss that already tracks functional damage.

Why deep-layer errors have limited functional cost. By layer 17, the residual stream appears to have largely determined the next-token distribution, leaving later layers to sharpen an already selected distribution (Geva et al., 2022; Belrose et al., 2023); late-layer pruning can therefore be cheap (Gromov et al., 2025; Men et al., 2025). The residual norm nevertheless keeps growing, so raw MSE tracks scale while splice KL tracks downstream function, producing the divergence in the final third of the network.

4 Reweighting and the placebo that challenged its explanation

The diagnosis suggests a direct repair, weighting each layer by what its error costs. We test it against two placebos. Every arm is trained as in Section 6 and scored on a pool of 1,487 problems from MATH-500, Minerva, AMC23 and OlympiadBench.

Table 2: Where each objective's direct hidden-state gradient energy lands, measured by differentiating the training loss on real hidden states from 64 held-out sequences. The two placebos put over 80% of their gradient energy on layers 1 to 4, whereas Functional places 61.7% on layers 13 to 20. Bold marks each row's largest share, amber columns are the placebos, and teal marks OPTD.

Layer band	OPRD	Scale-free	Direction	Functional	Uniform	Shuffled	OPTD-4
1–4	0.0%	17.6%	18.4%	0.4%	80.9%	**85.1%**	0.1%
5–12	0.5%	**39.7%**	35.5%	34.4%	16.1%	12.8%	2.5%
13–20	3.6%	32.0%	33.6%	**61.7%**	2.9%	2.0%	16.4%
21–24	17.9%	7.5%	8.9%	2.8%	0.1%	0.1%	**80.9%**
25–28	**77.9%**	3.1%	3.6%	0.6%	0.0%	0.0%	0.0%
Median depth	27	11	12	14	2	1	23

Four layerwise objectives. We compare raw MSE with three alternatives: *Scale-free* divides each error by the teacher's squared norm, *Direction-only* compares unit vectors, and *Functional* minimises $\sum_\ell w_\ell m_\ell / r_\ell$ with $w_\ell \propto g_\ell$ and r_ℓ the pre-training mean of m_ℓ. Appendix Table 5 lists every per-layer term. A single constant, computed at the first optimiser step and then fixed, matches every variant's initial representation-loss value to the published objective. The constants range from 2.7 to 1.0×10^5, isolating how a fixed initial loss budget is distributed across depth.

Two placebos. *Uniform* gives every layer the same w_ℓ, while *Shuffled* permutes the functional weights across layers. Shuffled preserves the coefficient multiset while destroying only its depth assignment; both placebos are scale-matched. If the measured depth profile were what helps, Functional would therefore beat the shuffled placebo.

Direct gradient allocation. Under $\sum_\ell w_\ell \, m_\ell / r_\ell$, expected gradient energy scales as w_ℓ^2 / r_ℓ. Since r_ℓ spans four orders of magnitude, Uniform concentrates energy in shallow layers. Autograd measurements in Table 2 confirm the contrast: Uniform assigns 80.9% to layers 1–4 and almost none to the final four, whereas Functional assigns 61.7% to layers 13–20 and only 0.4% to layers 1–4. The published loss instead concentrates energy in layers 27–28.

Results. Every variant beats the published objective, but the five schemes span only 1.16 accuracy points on the pool benchmarks. Six of ten paired-bootstrap intervals contain zero; all four separating contrasts favour a placebo. Shuffled beats Scale-free, Functional, and Direction-only by 1.16, 0.99, and 0.84 points, while Uniform beats Scale-free by 0.75. Rankings are consistent across seven benchmarks (mean rank correlation $+0.35$ versus a permutation null of 0.00, $p = 0.008$): Shuffled ranks first on six, Functional on none, and Scale-free last on five.

The shared property. Shuffled breaks Functional's depth correspondence yet uses the same weight multiset, outperforms Functional on the pooled evaluation, and ranks above it on six of seven benchmarks. All five schemes reduce the final-four gradient-energy share below 4% while redistributing the remainder to earlier layers. Where the rest goes differs widely, from layers 1 to 4 under both placebos to layers 13 to 20 under Functional, yet on the pool the placebos score at least as well as Functional.

5 On-policy truncated distillation

To reproduce this property while preserving the published allocation over retained layers, we remove the final N terms from the loss sum:

$$\mathcal{L}_{\text{OPTD-}N} \;=\; c_N \sum_{\ell=1}^{L-N} \left\| h_\ell^S - h_\ell^T \right\|_{\text{mean}}^2. \tag{3}$$

Equation 3 is Equation 1 with its final N terms removed and one prefactor chosen at the first optimisation step to match the initial published loss. It needs no measured importance profile or fitted weight vector; N is the only hyperparameter, and the code changes only the summation range.

```
# Published objective
rep_loss = sum(mse[l] for l in range(L)) / L
# OPTD-N, for example N = 4
rep_loss = c_N * sum(mse[l] for l in range(L - N))
```

Holding the initial loss fixed multiplies each retained coefficient by $1/\rho_N$, where ρ_N is the retained fraction of the published loss. The multiplier is $2.3\times$, $4.8\times$, and $10.8\times$ for $N = 2$, 4, and 6, respectively. OPTD-N is therefore the published loss restricted to the first $L - N$ layers and rescaled to the same initial level.

Truncation versus amplification. Truncation changes both retained layers and their amplification. Amplification rises from $2.3\times$ to $86\times$, but accuracy is flat for $N \in \{2, 4, 6\}$ and declines at larger cuts. OPTD-2 and OPTD-6 are indistinguishable despite nearly fivefold different amplification; OPTD-12 performs worst.

Truncation versus down-weighting. Each reweighting arm assigns under 4% of direct gradient energy to the final four layers and shifts the remainder to median depths 1–14 (Table 2); truncation instead leaves the retained-layer allocation unchanged. OPTD-4 beats Shuffled by 1.66 points (95% CI $[+0.95, +2.37]$) and the three measured schemes by 2.51–2.82 points.

Choosing N. Figure 4 shows a broad plateau: OPTD-2, -4, and -6 retain 43%, 21%, and 9% of the published loss and are indistinguishable, whereas -8 and -12 retain 4% and 1% and lie beyond it. The plateau is read off the trained arms. In the primary pair every cut that keeps 9% to 43% of the OPRD loss works about equally well, and $N = 4$ lies inside that range.

6 Experiments

Protocol. The student is DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025), a Qwen2.5-Math model fine-tuned on R1 traces (Yang et al., 2024b); the teacher is `JustRL-DeepSeek-1.5B` (He et al., 2025). All arms follow OPRD (Yang et al., 2026), except for the hidden-state metric, a $4{,}096$-token response limit, and `float32` arithmetic on V100 GPUs; each trains for 500 steps on DAPO-Math-17k (Yu et al., 2025). We evaluate $1{,}630$ problems with the released grader, using `avg@16` on the original three benchmarks and AMC23 and `avg@4` on the other pool benchmarks. Paired percentile-bootstrap intervals use the $1{,}487$ pool problems and $B = 10{,}000$ replicates; two-seed accuracies are averaged per problem before resampling. Every contrasted arm is retrained under this pipeline because pipelines differ by 3.3 AIME24 points on the same released weights.

The resampling unit is the problem, so all of a problem's completions are drawn together. Averaging the two seeds before resampling gives each problem one paired score per arm, while resampling individual generations would treat multiple answers to the same prompt as independent observations and make the intervals artificially narrow. We therefore report contrasts only on problem identifiers shared by both arms; the same paired set is used for every comparison in Table 4.

Main result. Table 3 shows that every reweighted arm improves on the published objective. OPTD-4 has the highest mean accuracy and beats the teacher on all four pool benchmarks; the published objective beats the teacher on none. Among the reweighted arms the shuffled placebo scores highest, and OPTD-4 higher still.

Seeds. Seven arms use two seeds and four use one; k in Table 3 reports these counts. Two-seed accuracies are averaged per problem before contrasts. The smallest significant contrast is 0.75 points (Uniform versus Scale-free); every OPTD-4 contrast exceeds twice that value.

OPTD against every alternative. Table 4 compares OPTD-4 on the same $1{,}487$ problems. The six reweighting contrasts whose intervals contain zero define the noise floor; Shuffled and Uniform assign 85.1% and 80.9% of direct gradient energy to layers 1–4, with a non-significant 0.41-point difference. The largest additional null contrast is 0.58; every OPTD-4 contrast exceeds all six.

Table 3: Accuracy on seven benchmarks, 1,630 problems. The first three columns and AMC23 are `avg@16`, the other three `avg@4`. k is the number of training seeds, averaged per problem before any contrast is taken. Bold marks the best trained arm in a column, amber rows are the two placebo objectives and teal rows are ours.

Arm	k	AIME24	AIME25	AIMO	MATH-500	Minerva	AMC23	Olymp.	Mean
Reference models									
Student, no distillation	1	24.17	20.21	53.31	64.75	19.03	57.97	34.93	39.19
Teacher	1	37.71	26.25	67.92	64.60	23.25	75.94	44.78	48.64
Published objectives									
OPD top-16, released	1	30.00	23.54	62.05	63.30	22.24	71.09	43.48	45.10
OPRD (published)	2	33.23	24.17	64.12	63.05	21.74	72.19	44.65	46.16
Reweighted, measured									
Scale-free	2	36.35	26.56	65.63	65.53	22.24	75.78	44.93	48.15
Direction-only	2	36.67	25.31	67.24	65.85	22.93	76.17	45.07	48.46
Functional	2	36.67	25.52	68.26	65.65	22.93	77.66	44.81	48.79
Reweighted, placebo									
Uniform	2	36.35	24.58	66.91	66.68	23.58	76.25	45.17	48.50
Shuffled	2	37.50	**29.06**	69.88	66.98	22.70	78.20	46.07	50.06
OPTD (ours)									
OPTD-2	1	37.71	27.29	70.18	68.80	**25.46**	**78.75**	45.93	50.59
OPTD-4	2	36.77	28.54	**70.75**	**69.90**	25.32	**78.75**	**46.48**	**50.93**
OPTD-6	1	**39.58**	27.71	70.71	69.20	23.90	**78.75**	46.00	50.84
OPTD-8	1	35.21	28.54	67.62	68.85	23.35	**78.75**	45.96	49.75
OPTD-12	1	29.58	26.67	64.16	67.60	19.67	72.03	41.30	45.86

Table 4: OPTD-4 against every alternative, and against the study's own noise floor, on the same 1,487 problems. The floor is the six reweighting contrasts whose intervals contain zero. $*$ marks an interval that excludes zero.

	Contrast	Δ	95% CI
treatment	OPTD-4 − OPRD (published)	$+3.97*$	$[+3.22, +4.73]$
	OPTD-4 − Scale-free	$+2.82*$	$[+2.09, +3.54]$
	OPTD-4 − Functional	$+2.65*$	$[+1.94, +3.37]$
	OPTD-4 − Direction-only	$+2.51*$	$[+1.79, +3.22]$
	OPTD-4 − Uniform (placebo)	$+2.07*$	$[+1.38, +2.76]$
	OPTD-4 − Shuffled (placebo)	$+1.66*$	$[+0.95, +2.37]$
floor	Functional − Uniform (placebo)	-0.58	$[-1.22, +0.04]$
	Direction-only − Uniform (placebo)	-0.44	$[-1.07, +0.19]$
	Shuffled (placebo) − Uniform (placebo)	$+0.41$	$[-0.23, +1.05]$
	Direction-only − Scale-free	$+0.31$	$[-0.32, +0.96]$
	Functional − Scale-free	$+0.17$	$[-0.48, +0.82]$
	Functional − Direction-only	-0.15	$[-0.76, +0.46]$

How many layers. Figure 4 shows that OPTD-2, -4, and -6 are indistinguishable on the 1,487 pool problems. OPTD-8 trails OPTD-4 by 0.95 points, while OPTD-12 is 0.37 points below the published objective and underperforms it on four of seven benchmarks. The plateau spans retained loss masses from 43% to 9%.

Cost of the instrument. The layerwise splice measurement can be run before selecting a training objective. A sweep over all L layers requires $L + 1$ teacher forward passes and no backward pass, because the student's hidden states are treated as constants. For the primary pair this comes to 29 forward passes per sweep.

What the reweighting gain tracks. Across seven benchmarks, reweighting gain correlates with the teacher's remaining advantage ($r = +0.84$, permutation $p = 0.019$), but not with the total distillable gap ($r = +0.53$, $p = 0.22$). On OlympiadBench, where the teacher leads by only

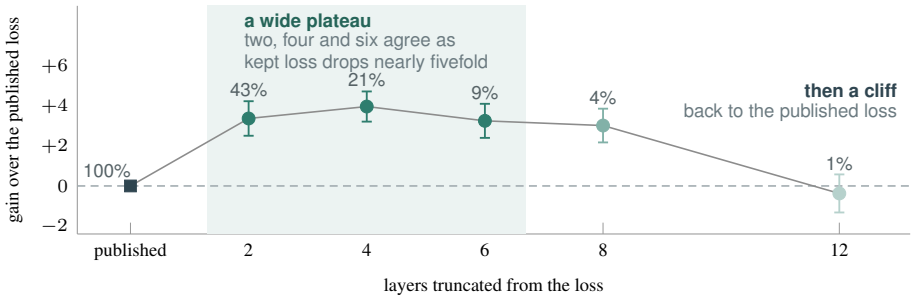

Figure 4: How much to truncate. Points are gains over the published objective on the 1,487 problems of the pool set, with 95% paired bootstrap intervals; the grey figure above each point is the share of the published loss that survives the cut.

0.13 points, reweighting averages +0.56 and Shuffled reaches +1.43; OPTD-4 gains +1.83. This pattern suggests that reweighting recovers transfer the published objective missed, and the untrained student's gap to the teacher does not significantly predict it. It also explains why a method can show a modest average gain while producing a large improvement on a benchmark where the published student is close to, but still below, the teacher.

Near and far from the teacher's distribution. Reweighting gains are larger near the teacher's distribution: the near–far gap is +2.6–+3.8 points, partly because additional extractable answers convert to correct solutions at 83% near the teacher but only 26% away. Three reweighting arms increase the rate of producing an extractable answer by similar amounts on the near and far pairs, +3.4 and +3.2 points, respectively, but the additional answers are much more likely to be correct near the teacher's distribution. Those conversion rates correspond to accuracy gains of +2.86 points near and +0.83 points far, with both intervals excluding zero. OPTD-4 gains +6.71 points near and +2.71 far, beating the teacher on every pool benchmark; it also beats the teacher by +5.30 points on MATH-500 and +1.70 on OlympiadBench. OPTD-12 instead gains only +2.20 points near and loses 2.71 points on the far pair. Thus moderate truncation transfers broadly, whereas excessive truncation is particularly costly out of distribution; OPTD-2, OPTD-6, and OPTD-8 still improve the far pair by at least +1.46 points.

7 Conclusion

Squared hidden-state error implicitly weights layers by representation scale. In the primary pair, the final four layers receive 79.3% of direct gradient energy but account for only 23.5% of splice-measured damage; across four domains, the over-allocation is 2.62–2.78×. Measured functional weights improve accuracy, but their random permutation performs better on six of seven benchmarks. The gain therefore comes from what the schemes share, and their measured assignment to depth adds nothing.

All reweighting arms reduce the final-four gradient share below 4%. OPTD reproduces this shared intervention by dropping the final terms while preserving the published relative coefficients on retained layers. OPTD-4 gains 3.97 points over OPRD, beats every reweighting scheme and the teacher on the pool benchmarks, and remains effective when two to six layers are dropped. It needs only the integer N and a first-step scale match; no layer-importance profile or fitted weight vector is required.

The splice reports each layer's downstream cost as KL divergence in nats using forward passes on student rollouts. Use it before training when the loss grows faster than the splice profile, then test a moderate cut; for pairs such as Qwen3-4B, whose norm peaks mid-network and whose profiles already agree, retain the published objective. With the norm-matched control and the shuffled placebo, the splice yields a three-step procedure. Measure downstream damage first, test the assignment second, and choose the smallest intervention the profile supports. Because the measurements precede training, this decision can be audited independently of optimiser dynamics and repeated for each new compatible model pair. The diagnosis adds no parameters and requires no retraining.

REPRODUCIBILITY STATEMENT

The supplementary material includes the training patch, the splice instrument, the analysis scripts, and the derived data. Its README gives the command for regenerating every table and figure, while `checks/paper_numbers.py` verifies each reported number against the derived data. The patch that introduces each variant is implemented in a single function and reproduced in Appendix A; Appendix I records the seeds, hardware, evaluation protocol, and grading procedure.

AI USE STATEMENT

During the experiments, we used large language models (LLMs) as an auxiliary tool for experiment monitoring and management. Specifically, the LLMs were used to monitor experiment logs and runtime information, identify potential execution issues or anomalies, and assist in reporting the status of ongoing experiments. The LLMs did not determine the research questions, experimental methodology, hyperparameter settings, or final experimental conclusions. All experimental configurations, result verification, analysis, and scientific conclusions were determined and validated by the authors. We take full responsibility for the final content and results of this work.

ETHICS STATEMENT

This work studies a training objective on public models and benchmarks and collects no new data.

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 (ICLR)*, 2024.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations (ICLR), Workshop Track*, 2017.

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2): 251–276, February 1998.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations (ICLR)*, 2024.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, July 2016.

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

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 535–541. ACM, 2006.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. The hitchhiker's guide to testing statistical significance in natural language processing. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1383–1392, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*, volume 57 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, New York, 1993.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning (ICML)*, 2023.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers. In *International Conference on Learning Representations (ICLR)*, 2025.

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

Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, Ning Ding, and Zhiyuan Liu. JustRL: Scaling a 1.5B LLM with a simple RL recipe. *arXiv preprint arXiv:2512.16649*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

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.

Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

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 *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL*, 2023.

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.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

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

Philipp Koehn. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu (eds.), *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 388–395, Barcelona, Spain, 2004. Association for Computational Linguistics.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning (ICML)*, 2019.

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 *ACM Symposium on Operating Systems Principles (SOSP)*, 2023.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan-ang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, April 2026.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri 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 (ICLR)*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-Pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. ShortGPT: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20192–20204, 2025.

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

William Merrill, Vivek Ramanujan, Yoav Goldberg, Roy Schwartz, and Noah A. Smith. Effects of parameter norm growth during transformer training: Inductive bias from gradient descent. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321, Suzhou, China, 2025. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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 (ICLR)*, 2015.

Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Victor Sanh, Thomas Wolf, and Alexander M. Rush. Movement pruning: Adaptive sparsity by fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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. In *European Conference on Computer Systems (EuroSys)*, pp. 1279–1297, 2025.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for BERT model compression. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4323–4332, 2019.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations (ICLR)*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. Symbolic knowledge distillation: from general language models to commonsense models. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2022.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning (ICML)*, 2020.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2.5-Math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024b.

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

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. *arXiv preprint arXiv:2606.06021*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations (ICLR)*, 2017.

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

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A IMPLEMENTATION

A.1 THE OBJECTIVE AND THE SINGLE CHANGED FUNCTION

Each arm uses the published training recipe and changes only one function. The recipe can combine a token-level on-policy distillation loss with a representation term over the L transformer blocks. Every arm here uses its representation-only configuration, and the arms differ only in that term. Let $m_\ell = \|h_\ell^S - h_\ell^T\|^2_{\text{mean}}$ denote the per-layer mean squared error and let $n_\ell = \|h_\ell^T\|^2_{\text{mean}}$ denote the teacher's squared norm. The published term and its six variants are listed in Table 5.

A single function in the training code selects the per-layer term. The schematic below omits bookkeeping but shows the operations shared by the arms. The `unit` operation rescales a state to unit norm. Functional and Shuffled use the normalised-error parameterisation with different weight vectors; Uniform uses the corresponding unweighted parameterisation; and OPTD keeps the published coefficients while setting the final entries to zero.

```
raw = ((h_s - h_t.detach())**2).mean(-1)
if mode == "none":     term = raw
if mode == "rel":      term = raw / (h_t**2).mean(-1)
if mode == "l2":       term = ((unit(h_s) - unit(h_t))**2).mean(-1)
if mode == "funcw":    term = w[l] * raw / r[l]   # also OPTD-N
loss = c * sum(term over layers)   # c fixed at step 1, Eq. 4
```

Table 5: The published per-layer term and its six variants. Each variant is a branch of the same function; teal marks OPTD and amber marks the two placebos.

Name	Code name	Per-layer term	Free quantities
OPRD (published)	`none`	$\frac{1}{L}\, m_\ell$	none
Scale-free	`rel`	$\frac{1}{L}\, m_\ell / n_\ell$	none
Direction-only	`l2`	$\frac{1}{L}\|\hat{h}_\ell^S - \hat{h}_\ell^T\|^2$	none
Functional	`funcw`	$w_\ell\, m_\ell / r_\ell$	w from g, r measured
Uniform (placebo)	`uniform`	$\frac{1}{L}\, m_\ell / r_\ell$	r measured
Shuffled (placebo)	`funcwP`	$w_{\pi(\ell)}\, m_\ell / r_\ell$	π a fixed permutation
OPTD-N	`drop`N	$c_N\, m_\ell\, [\ell \le L - N]$	N only

$\hat{h}$ denotes a unit vector. The functional weights satisfy $w_\ell \propto g_\ell$, where g_ℓ is defined in Equation 2; they are normalised to sum to one. The quantity r_ℓ is the mean of m_ℓ measured once before training. The permutation π is selected before training: among the 168 of 200 seeded draws whose total-variation distance from the functional weights lies between 0.30 and 0.50, we use the draw at the median distance and then keep it fixed. The OPTD weight file is the published reference vector with its final N entries set to zero. Thus, the retained layers keep the published relative weighting, while only the overall level changes. As in every other arm, OPTD uses the single scale-matching constant in Equation 4. This constant fixes the total initial loss and therefore multiplies each retained layer by $1/\rho_N$, the reciprocal of the loss mass that those layers originally carried.

A.2 REMOVING THE LEARNING-RATE CONFOUND

Dividing by r_ℓ or n_ℓ changes the overall scale of the representation term. Without correction, a variant that merely trains at a different effective learning rate could appear to use a better layer weighting. Each arm therefore has one scalar c. We compute it at the first optimiser step as the ratio of the across-layer total of the published loss to that of the variant, and then keep it fixed for all 500 steps:

$$c \;=\; \Big(\textstyle\sum_\ell \frac{1}{L} m_\ell\Big) \Big/ \Big(\textstyle\sum_\ell \text{term}_\ell\Big)\Big|_{\text{step 1}}, \qquad \mathcal{L}_{\text{rep}} \;=\; c \textstyle\sum_\ell \text{term}_\ell. \tag{4}$$

Using the first of the 64 held-out sequences, the constants are 60.3 for scale-free, 1.02×10^5 for direction-only, 3.28 for functional, 3.20 for uniform, 3.25 for shuffled, and 3.66, 3.53, 3.40, 3.27

and 2.73 for OPTD-2, OPTD-4, OPTD-6, OPTD-8 and OPTD-12, respectively. Each run computes its constant from its own first batch. A unit test verifies that the rescaled and published losses agree at step one: for the published, scale-free, layer-relative and direction-only metrics, the discrepancy is below 10^{-6}, against an asserted tolerance of 10^{-4}. Thus every arm starts from the published loss on its first micro-batch. Averaged over all 64 sequences, the scaled losses range from 2.63 to 3.11, compared with 3.16 for the published loss.

A.3 Generating the OPTD Weight Files

The OPTD arms use the same weight-file format as the reweighting arms. The retained layers carry the published coefficients, whereas the truncated layers carry zero. Before use, we validate the generator by asking it to produce the four-layer file and comparing the result with our hand-built file; the two agree after normalisation. We then generate the two-, six-, eight- and twelve-layer variants with the same rule. This check guarantees that OPTD differs from the published objective only through the terms that it removes.

A.4 Cost

The representation term reuses hidden states already produced by the forward pass, and removing the final four blocks reduces its element-wise operations by 14%. The arms do not differ in parameter count, number of forward passes, or number of optimiser steps.

B The splice instrument

B.1 Procedure

For a fixed batch of student rollouts, we run the teacher twice. The first pass is clean and produces $p^T(\cdot \mid x_{<t})$ at every position. The second pass is identical except that the output of teacher block ℓ is overwritten with the output of the student's corresponding block at the same position. The teacher's blocks $\ell + 1, \ldots, L$, final normalisation, and unembedding then run unchanged. Equation 2 is the mean, over response positions, of the KL divergence between the two output distributions. A sweep over all L layers requires $L + 1$ teacher forward passes and no backward pass, because the student's states are treated as constants.

Three properties make this quantity comparable across depth. First, it is measured in nats of output-distribution divergence, the same unit at layer 1 and layer 28. Second, it is causal: it measures what the teacher's remaining computation does with the injected error, regardless of the error's distance from the teacher's state. Third, it requires no fitting, so no probe can absorb the effect through its capacity.

B.2 Controls

Table 6 reports the two controls used by the instrument alongside the measurement itself. The identity control checks the construction. The Gaussian control matches the student's displacement in norm, token by token, but replaces its direction with an arbitrary one. It therefore distinguishes directional sensitivity from a purely norm-based effect.

Table 6: Splice controls on the primary pair, 12 sequences. Entries are mean splice KL in nats per layer band, with the Gaussian norm-matched to the student's displacement at every token.

Splice at layer	1–8	9–16	17–27	28
Teacher's own state	0	0	0	0
Student's state	0.0172	0.1916	0.3184	0.3642
Norm-matched Gaussian	0.00183	0.0177	0.0314	0.1589
Ratio, student / Gaussian	9.4	10.8	10.1	2.3

The ratio remains close to ten through layer 27 and falls to 2.3 at layer 28. Splicing into the last block leaves only the final normalisation and unembedding, so little computation remains to amplify the direction of the student's error. Across the final four blocks, the student's state accounts for 24.4% of the splice cost, whereas the norm-matched Gaussian accounts for 41.7%.

C PER-LAYER MEASUREMENTS

Table 7 reports the measurement underlying Figure 3, averaged over 256 student rollouts from the primary pair. The two profiles agree through the first half of the network and diverge in the second. The loss column rises by four orders of magnitude, whereas the damage column rises by two orders of magnitude through layer 17 and then levels off.

Table 7: Per-layer loss and damage over 256 rollouts. m_ℓ is the raw mean squared error, g_ℓ is the splice KL in nats, and n_ℓ is the teacher's mean squared norm. The last two columns give the share assigned by each profile across depth.

ℓ	m_ℓ	g_ℓ	n_ℓ	w_ℓ^{raw}	w_ℓ^{g}	ℓ	m_ℓ	g_ℓ	n_ℓ	w_ℓ^{raw}	w_ℓ^{g}
1	0.0023	0.0014	0.63	0.00%	0.02%	15	0.200	0.300	6.1	0.18%	4.92%
2	0.0047	0.0036	0.99	0.00%	0.06%	16	0.242	0.315	6.6	0.22%	5.17%
3	0.0090	0.0072	1.36	0.01%	0.12%	17	0.337	0.323	8.0	0.31%	5.31%
4	0.0158	0.0109	1.66	0.01%	0.18%	18	0.683	0.331	10.2	0.62%	5.43%
5	0.0247	0.0177	2.12	0.02%	0.29%	19	0.870	0.330	13.1	0.79%	5.42%
6	0.0304	0.0273	2.26	0.03%	0.45%	20	1.223	0.333	16.5	1.11%	5.46%
7	0.0430	0.0499	2.78	0.04%	0.82%	21	1.696	0.334	21.3	1.54%	5.48%
8	0.0581	0.0752	3.22	0.05%	1.23%	22	4.103	0.339	32.4	3.72%	5.56%
9	0.0649	0.0959	3.25	0.06%	1.57%	23	5.389	0.341	45.4	4.88%	5.60%
10	0.0783	0.1284	3.45	0.07%	2.11%	24	7.333	0.345	60.2	6.64%	5.65%
11	0.0913	0.1792	3.84	0.08%	2.94%	25	10.02	0.347	93.4	9.07%	5.69%
12	0.1145	0.2269	4.40	0.10%	3.72%	26	14.73	0.351	111.9	13.34%	5.76%
13	0.1377	0.2616	5.07	0.12%	4.29%	27	21.04	0.353	162.5	19.05%	5.79%
14	0.1673	0.2840	5.66	0.15%	4.66%	28	41.75	0.384	173.6	37.80%	6.30%

Aggregated into the bands of Table 2, the loss assigns 0.03%, 0.46%, 3.49%, 16.77% and 79.25% to layers 1–4, 5–12, 13–20, 21–24 and 25–28, respectively, whereas the damage assigns 0.38%, 13.14%, 40.65%, 22.30% and 23.54%. Bootstrapping the 256 rollouts 2,000 times gives $[0.7905, 0.7944]$ for the final-four loss share, $[0.2345, 0.2364]$ for the final-four damage share, and $[3.348, 3.385]$ for their ratio.

D GENERALITY

D.1 ACROSS MODEL PAIRS

Table 8 repeats the layer-profile measurement for eight student–teacher pairs from five released model families. Six pairs consist of a base model and its instruction-tuned sibling (Yang et al., 2024a;b; Grattafiori et al., 2024); one pairs the post-trained Qwen3-4B with its Instruct-2507 update (Yang et al., 2025); and the eighth is this paper's primary pair, a distilled student and the reinforcement-learning checkpoint trained from it. The key quantity is r_{norm}, the correlation between the log per-layer loss and the log teacher norm. It is at least 0.983 for seven of the eight pairs. The norm range reported in the table is unsquared, so the primary pair's $17\times$ corresponds, before rounding, to a $273\times$ ratio in the mean squared norm.

Qwen3-4B behaves differently. Its residual norm does not grow monotonically with depth: the profile peaks in the middle of the network and then declines. The loss is therefore not concentrated in the final blocks; its final-four share is 0.275, compared with a damage share of 0.280. For this pair, the loss already allocates its gradient approximately where the damage occurs.

Because both quantities in r_{norm} increase with depth, their correlation could arise from depth alone. We therefore also correlate layer-to-layer changes, which removes this shared trend. For the primary

Table 8: Raw MSE follows the teacher norm across eight pairs. $r_{\text{norm}} = \text{corr}(\log m_\ell, \log n_\ell)$ over layers, and the range columns give the ratio of the largest to the smallest layer, with the norm unsquared. The first row is the 256-rollout profile of Table 7 and the others use 19 to 64 sequences; rose marks Qwen3-4B, whose norm peaks mid-network.

Student	Teacher	Family	L	r_{norm}	$\|h^T\|$ range	Loss range	Last-4 share loss	Last-4 share damage
R1-Distill 1.5B	JustRL 1.5B	Qwen2.5-Math	28	0.993	17×	1.8×10^4	0.793	0.235
Qwen2.5 1.5B	+ *Instruct*	Qwen2.5	28	0.984	12×	1.6×10^3	0.788	0.251
Qwen2.5 3B	+ *Instruct*	Qwen2.5	36	0.987	19×	9.7×10^3	0.750	0.181
Qwen3 4B	+ *Instruct*	Qwen3	36	0.841	84×	3.4×10^4	0.275	0.280
Llama 3.2 3B	+ *Instruct*	Llama-3.2	28	0.991	35×	6.4×10^3	0.686	0.300
Qwen2.5 7B	+ *Instruct*	Qwen2.5	28	0.991	38×	6.8×10^4	0.792	0.211
Qwen2.5-Math 7B	+ *Instruct*	Qwen2.5-Math	28	0.983	65×	2.1×10^4	0.734	0.397
Llama 3.1 8B	+ *Instruct*	Llama-3.1	32	0.984	80×	8.2×10^4	0.593	0.177

pair, changes in log loss track changes in log norm at $+0.56$, whereas changes in log damage track them at only $+0.05$.

D.2 Across evaluation domains

Table 9 repeats the measurement on generations drawn from four benchmarks, two of which were not used to target the model during training. The over-allocation remains between 2.62 and 2.78 on all four benchmarks, indicating that it is a property of the metric itself.

Table 9: Over-allocation measured on generations from four benchmarks, with 10 sequences per benchmark. The shares are computed relative to each profile's own total over the final four blocks.

Benchmark	Domain	Last-4 loss	Last-4 damage	Over-allocation
MATH-500	competition	75.6%	28.8%	2.62×
AMC23	competition	77.1%	28.3%	2.73×
Minerva	undergraduate	78.5%	28.3%	2.78×
OlympiadBench	olympiad	76.7%	28.4%	2.70×

E Where the gradient goes

Table 2 in the main text is measured with autograd. We feed the student's and teacher's hidden states from 64 held-out rollouts into the training loss imported from the training code. The student states are leaf tensors, so autograd reports $\partial\mathcal{L}/\partial h_\ell$ exactly, without backpropagating through the network. The share attributed to layer ℓ is $\|\partial\mathcal{L}/\partial h_\ell\|^2$, normalised over layers. Shares are normalised separately within each arm, so the scale-matching constant cancels.

For the published loss, this measurement and the loss-mass profile in Table 7 are the same quantity computed in two ways: the gradient of Equation 1 at layer ℓ is proportional to the residual. The estimates agree within 1.3 percentage points: autograd assigns 77.95% to the final four blocks, compared with 79.25% from the 256-rollout profile.

Uniform weights concentrate the gradient on the shallowest layers. Under the parameterisation $\mathcal{L} = \sum_\ell w_\ell m_\ell / r_\ell$, the gradient energy at layer ℓ scales as w_ℓ^2/r_ℓ, while r spans a factor of 1.8×10^4 across depth. Equal w_ℓ therefore concentrates the gradient on the first block, making Uniform an extreme arm. Uniform and Functional are approximately the two most separated members of this family, yet their accuracies are statistically indistinguishable.

Table 10: All pairwise contrasts among the eleven arms on the 1,487 pool problems, paired bootstrap with $B = 10{,}000$, read A minus B and sorted by Δ; $*$ marks an interval excluding zero. OPRD is the published objective, rel scale-free, dir direction-only, func functional, unif uniform, shuf shuffled, and $-N$ stands for OPTD-N.

A	B	Δ	95% CI	A	B	Δ	95% CI
-4	-12	$+4.34*$	$[+3.49, +5.21]$	rel	-12	$+1.52*$	$[+0.60, +2.45]$
-4	OPRD	$+3.97*$	$[+3.22, +4.73]$	-2	unif	$+1.47*$	$[+0.68, +2.25]$
-2	-12	$+3.74*$	$[+2.72, +4.77]$	dir	OPRD	$+1.46*$	$[+0.80, +2.10]$
-6	-12	$+3.63*$	$[+2.66, +4.59]$	-6	unif	$+1.35*$	$[+0.57, +2.17]$
-8	-12	$+3.39*$	$[+2.48, +4.30]$	func	OPRD	$+1.32*$	$[+0.69, +1.95]$
-2	OPRD	$+3.37*$	$[+2.51, +4.24]$	shuf	rel	$+1.16*$	$[+0.52, +1.83]$
-6	OPRD	$+3.25*$	$[+2.41, +4.11]$	rel	OPRD	$+1.15*$	$[+0.45, +1.82]$
-8	OPRD	$+3.02*$	$[+2.18, +3.86]$	-8	unif	$+1.12*$	$[+0.33, +1.89]$
-4	rel	$+2.82*$	$[+2.09, +3.54]$	-2	shuf	$+1.07*$	$[+0.25, +1.87]$
shuf	-12	$+2.68*$	$[+1.78, +3.63]$	shuf	func	$+0.99*$	$[+0.37, +1.63]$
-4	func	$+2.65*$	$[+1.94, +3.37]$	-4	-8	$+0.95*$	$[+0.21, +1.69]$
-4	dir	$+2.51*$	$[+1.79, +3.22]$	-6	shuf	$+0.95*$	$[+0.13, +1.75]$
shuf	OPRD	$+2.31*$	$[+1.61, +3.01]$	shuf	dir	$+0.84*$	$[+0.21, +1.51]$
unif	-12	$+2.27*$	$[+1.40, +3.17]$	unif	rel	$+0.75*$	$[+0.09, +1.42]$
-2	rel	$+2.22*$	$[+1.41, +3.05]$	-4	-6	$+0.71$	$[-0.04, +1.46]$
-6	rel	$+2.11*$	$[+1.30, +2.92]$	-8	shuf	$+0.71$	$[-0.06, +1.50]$
-4	unif	$+2.07*$	$[+1.38, +2.76]$	-4	-2	$+0.60$	$[-0.13, +1.34]$
-2	func	$+2.06*$	$[+1.23, +2.89]$	unif	func	$+0.58$	$[-0.04, +1.22]$
-6	func	$+1.94*$	$[+1.14, +2.75]$	unif	dir	$+0.44$	$[-0.19, +1.07]$
-2	dir	$+1.91*$	$[+1.10, +2.75]$	shuf	unif	$+0.41$	$[-0.23, +1.05]$
unif	OPRD	$+1.90*$	$[+1.25, +2.56]$	OPRD	-12	$+0.37$	$[-0.57, +1.33]$
-8	rel	$+1.87*$	$[+1.04, +2.71]$	-2	-8	$+0.35$	$[-0.55, +1.27]$
dir	-12	$+1.83*$	$[+0.91, +2.76]$	dir	rel	$+0.31$	$[-0.32, +0.96]$
-6	dir	$+1.79*$	$[+0.96, +2.62]$	-6	-8	$+0.24$	$[-0.62, +1.08]$
-8	func	$+1.70*$	$[+0.92, +2.53]$	func	rel	$+0.17$	$[-0.48, +0.82]$
func	-12	$+1.69*$	$[+0.76, +2.61]$	dir	func	$+0.15$	$[-0.46, +0.76]$
-4	shuf	$+1.66*$	$[+0.95, +2.37]$	-2	-6	$+0.12$	$[-0.75, +0.97]$
-8	dir	$+1.56*$	$[+0.79, +2.34]$				

Table 11: The layer-count ablation in full. Retained mass is the share of the published loss surviving the truncation; contrasts are against the published objective and against OPTD-4 on the 1,487 problems of the pool set.

Arm	k	Layers kept	Retained mass	vs. published Δ	vs. published 95% CI	vs. OPTD-4 Δ	vs. OPTD-4 95% CI
OPRD (published)	2	28	100.0%	*reference*		$-3.97*$	$[-4.73, -3.22]$
OPTD-2	1	26	43.2%	$+3.37*$	$[+2.51, +4.24]$	-0.60	$[-1.34, +0.13]$
OPTD-4	2	24	20.7%	$+3.97*$	$[+3.22, +4.73]$	*reference*	
OPTD-6	1	22	9.2%	$+3.25*$	$[+2.41, +4.11]$	-0.71	$[-1.46, +0.04]$
OPTD-8	1	20	4.0%	$+3.02*$	$[+2.18, +3.86]$	$-0.95*$	$[-1.69, -0.21]$
OPTD-12	1	16	1.2%	-0.37	$[-1.33, +0.57]$	$-4.34*$	$[-5.21, -3.49]$

F FULL RESULTS

Table 10 reports every pairwise contrast among the eleven trained arms. Table 11 reports the layer-count ablation together with the retained loss mass. OPTD-2 and OPTD-6 retain amounts of the published loss that differ by nearly a factor of five, yet their accuracies are statistically indistinguishable.

The OPTD arms form the top group, with OPTD-12 as the sole exception. In all four statistically separated contrasts among the five reweighting arms, a placebo is on the winning side.

Table 12: Gain over the published objective for the five reweighting arms, benchmark by benchmark. Bold marks the best arm in a row. The mean pairwise rank correlation between rows is $+0.35$ against a permutation null of 0.00, $p = 0.008$.

Benchmark	Scale-free	Direction-only	Functional	Uniform†	Shuffled†	Winner
AIME24	$+3.13$	$+3.44$	$+3.44$	$+3.13$	$\mathbf{+4.27}$	Shuffled†
AIME25	$+2.40$	$+1.15$	$+1.35$	$+0.42$	$\mathbf{+4.90}$	Shuffled†
AIMO	$+1.51$	$+3.13$	$+4.14$	$+2.79$	$\mathbf{+5.76}$	Shuffled†
MATH-500	$+2.48$	$+2.80$	$+2.60$	$+3.63$	$\mathbf{+3.93}$	Shuffled†
Minerva	$+0.51$	$+1.19$	$+1.19$	$\mathbf{+1.84}$	$+0.97$	Uniform†
AMC23	$+3.59$	$+3.98$	$+5.47$	$+4.06$	$\mathbf{+6.02}$	Shuffled†
OlympiadBench	$+0.28$	$+0.43$	$+0.17$	$+0.52$	$\mathbf{+1.43}$	Shuffled†
Times first	0	0	0	1	6	
Times last	5	0	1	2	0	

† placebo, weights that encode nothing about depth, shown in amber; a tie counts for both arms.

F.1 The ordering of the reweighting arms

Table 12 gives the ordering benchmark by benchmark. Across the four pool benchmarks, the five arms lie within 1.2 points of one another yet retain a consistent ordering: the mean pairwise rank correlation is $+0.35$ against a permutation null of zero ($p = 0.008$). Shuffled ranks first on six of seven benchmarks, Functional ranks first on none, and Scale-free ranks last on five. The measured assignment of weights to depth therefore offers no advantage over a random assignment. All five arms allocate under 4% of the gradient to the final four layers and redistribute the remainder. Section 5 removes the tail while leaving the remaining allocation unchanged from the published objective.

G Mechanism

Table 13: Gain over the published objective per benchmark, in points. The first five columns are competition mathematics near the teacher's training distribution and the next two are undergraduate science and olympiad mathematics. The last column is the mean gain on MATH-500 and AMC23 minus the mean gain on Minerva and OlympiadBench.

Arm	k	AIME24	AIME25	AIMO	MATH-500	AMC23	Minerva	Olymp.	In − out
Scale-free	2	$+3.13$	$+2.40$	$+1.51$	$+2.48$	$+3.59$	$+0.51$	$+0.28$	$+2.64$
Direction-only	2	$+3.44$	$+1.15$	$+3.13$	$+2.80$	$+3.98$	$+1.19$	$+0.43$	$+2.58$
Functional	2	$+3.44$	$+1.35$	$+4.14$	$+2.60$	$+5.47$	$+1.19$	$+0.17$	$+3.35$
Uniform (placebo)	2	$+3.13$	$+0.42$	$+2.79$	$+3.63$	$+4.06$	$+1.84$	$+0.52$	$+2.67$
Shuffled (placebo)	2	$+4.27$	$+4.90$	$+5.76$	$+3.93$	$+6.02$	$+0.97$	$+1.43$	$+3.77$
OPTD-2	1	$+4.48$	$+3.13$	$+6.06$	$+5.75$	$+6.56$	$+3.72$	$+1.28$	$+3.66$
OPTD-4	2	$+3.54$	$+4.38$	$+6.63$	$+6.85$	$+6.56$	$+3.58$	$+1.83$	$+4.00$
OPTD-6	1	$+6.35$	$+3.54$	$+6.59$	$+6.15$	$+6.56$	$+2.16$	$+1.35$	$+4.60$
OPTD-8	1	$+1.98$	$+4.38$	$+3.50$	$+5.80$	$+6.56$	$+1.61$	$+1.31$	$+4.72$
OPTD-12	1	-3.65	$+2.50$	$+0.04$	$+4.55$	-0.16	-2.07	-3.35	$+4.91$

G.1 What the reweighting gain tracks

Across seven benchmarks, the gain from reweighting correlates with the headroom left by the published objective (the teacher's score minus the published objective's score), with $r = +0.842$ and a permutation-test $p = 0.019$. The total distillable gap (teacher minus untrained student) is not predictive, with $r = +0.529$ and $p = 0.218$. In both analyses, the gain is the mean improvement of the five reweighting arms over the published objective, computed benchmark by benchmark. This pattern suggests that reweighting recovers transfer that the published loss failed to provide, and that its benefit grows with the missing transfer. On OlympiadBench, the published objective is already

within 0.13 points of the teacher; the five reweighting arms average a gain of $+0.56$, whereas OPTD gains 1.83 points, more than three times the reweighting average.

G.2 Alternative explanations

1. *A learning-rate effect.* Every arm is scale-matched by Equation 4 and starts at the same loss value. The constants differ by four orders of magnitude between arms and the accuracies do not order with them.
2. *Less representation pressure is better.* OPTD-12 keeps the least and is the worst arm in the study, 0.37 points below the published objective.
3. *The OPTD arms train longer or see more data.* Every arm runs 500 steps on the same prompt set, with the same batch composition and the same seed schedule throughout.
4. *The gain is a decoding artefact.* Three reweighting arms raise the rate at which the model emits an extractable answer by nearly the same amount in and out of domain, $+3.43$ and $+3.15$ points, both with intervals excluding zero. What differs is how much of that converts to a correct answer, 83% in domain against 26% out.
5. *The functional weights are mis-measured.* The shuffled placebo reuses the same weights in a scrambled order and performs better, while weights fitted on a disjoint prompt sample move only 0.4% of the layer budget.

G.3 Functional distance to the teacher

On shared rollouts, Scale-free, Direction-only, Functional, and Uniform each end closer to the teacher in splice KL than the published objective. The paired differences are -0.066, -0.067, -0.089, and -0.098, respectively, and all corresponding intervals exclude zero. The untrained student has splice KL 5.41; the published objective closes 95.5% of this gap, whereas the variants close between 96.7% and 97.3%. Thus, every arm moves the representation in the direction requested by its loss, but representation distance alone does not explain the accuracy ranking.

G.4 Where the gain lands

Table 13 breaks down the gain by benchmark. The reweighting arms concentrate their gains on benchmarks close to the teacher's training distribution, with a near–far gap of $+2.6$ to $+3.8$ points. The largest gap among the five is Shuffled's, at $+3.77$. OPTD-2 through OPTD-8 improve both sides simultaneously. OPTD-4 gains $+6.71$ points on the near pair and $+2.71$ on the far pair, more than twice the best far-side gain from any reweighting arm ($+1.20$ for Shuffled). OPTD-12 loses 2.71 points on the far pair while still gaining on the near pair, which a ceiling effect alone cannot explain.

H Limitations

The diagnosis assumes a residual norm that grows with depth, as observed in seven of the eight pairs measured. Training experiments use only one 1.5B student–teacher pair.

I Reproducibility

I.1 Code and checks

The submitted code contains the training patch, weight generators and their checks, the splice instrument, and the analysis scripts. Every number is generated at runtime from the raw generation files, and each table emitter exits with a non-zero status if any cell lacks a measurement. Table 14 lists the properties on which the claims depend and the script that checks each one.

I.2 Models, data and hardware

The student is DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025), obtained by fine-tuning Qwen2.5-Math-1.5B (Yang et al., 2024b) on reasoning traces generated by DeepSeek-R1. The

Table 14: Properties checked by each verification script, independently of the analysis it guards.

Script	What it checks
`test_repnorm`	On synthetic states, layers with norms 500 and 30 and the same 1% relative error produce losses that differ by $259\times$. This is close to the $278\times$ ratio expected from a loss that depends only on the norm, with the teacher state detached.
`test_repnorm2`	The scale-matching constant makes the step-one loss equal to the published loss for the published, scale-free, layer-relative and direction-only metrics, asserted to within 10^{-4}.
`grad_check`	The splice divergence is differentiable end-to-end, with finite, non-vanishing gradients at layers 4, 10, 16, 22 and 28, and a backward pass that fits in memory at 4,096 tokens.
`sanity`	Splicing the teacher's own state into the teacher, in place of the student's, returns $g_\ell = 0$ at layers 1, 7, 14, 21 and 28.
`leak_check`	Weights fitted on a disjoint prompt sample move only 0.4% of the layer budget, even though the raw-MSE and functional profiles differ by $\mathrm{TV} = 0.567$.
`integrity`	Every score file parses, is complete, and holds the expected number of samples per problem.
`mkdrop`	The OPTD weight generator reproduces our hand-built four-layer file before writing its siblings.
`paper_numbers`	Every number printed in the text and tables matches the data on disk; the script parses the source and exits on any mismatch.

teacher is `JustRL-DeepSeek-1.5B` (He et al., 2025), the public reinforcement-learning checkpoint trained from that student. Training prompts come from DAPO-Math-17k (Yu et al., 2025). All arms use the representation-only configuration: no arm includes an output-space term, and the representation loss covers all L layers and the final 512 response positions. The original work trains with a 16,384-token response limit on eight A100 GPUs in bf16. Every arm in this study, including the reproduced baseline, instead uses a 4,096-token limit. Each arm runs 500 optimiser steps with the published optimiser (Kingma & Ba, 2015; Loshchilov & Hutter, 2019), learning-rate schedule, and batch composition (Yang et al., 2026) on eight V100 GPUs in `float32`, using verl for the training loop (Sheng et al., 2025) and vLLM for rollout generation (Kwon et al., 2023).

I.3 Seeds

The results are based on eighteen training runs. Seven objectives were run twice: the published loss, Scale-free, Direction-only, Functional, Uniform, Shuffled, and OPTD-4. Four were run once: OPTD-2, OPTD-6, OPTD-8, and OPTD-12. Two-seed arms use seeds 0 and 1, while one-seed arms use seed 0, for both data ordering and sampling. When two seeds are available, we average per-problem accuracies across seeds before computing any contrast.

I.4 Evaluation

We use two evaluation sets. The headline set contains AIME24, AIME25, and AIMO, for a total of 143 problems scored with `avg@16`, the protocol reported in the published work. The pool set contains MATH-500, Minerva, AMC23, and OlympiadBench, for 1,487 problems (Hendrycks et al., 2021; Lightman et al., 2024; Lewkowycz et al., 2022; He et al., 2024). We use `avg@4` except on AMC23, where we use `avg@16`. Generation uses temperature 0.7 and top-p 0.95, with a maximum of 8,192 new tokens on the headline set, AMC23, and OlympiadBench, and 4,096 on MATH-500 and Minerva, whose answers are shorter. The token budget is fixed per benchmark across all twenty-two evaluated models, and grading uses the released baseline verifier without modification. Contrast intervals are computed on the pool set.

I.5 BASELINE RETRAINING

Every arm used in a contrast is retrained under the same evaluation pipeline, because two pipelines can disagree when scoring the same weights. The released checkpoints include the authors' generation outputs. At an $8{,}192$-token budget, scoring those outputs with the authors' script gives 31.67 for the published objective and 32.29 for OPD top-16; scoring the same weights with our pipeline gives 35.00 and 30.00, respectively. The scores therefore differ by 3.3 points in one direction and 2.3 in the other; AIME25 and AIMO also shift by 0.21 and 1.66 points. The published table uses a $16{,}384$-token budget. The OPD top-16 row in Table 3 is the released checkpoint rescored with our pipeline; it is retained as a reference and is not used in any contrast.

I.6 STATISTICS

We compute paired 95% percentile-bootstrap intervals over problems with $B = 10{,}000$ resamples. The same resampled problem indices are used for both arms, so each interval is on the between-arm difference (Efron & Tibshirani, 1993; Koehn, 2004). The problem is the resampling unit because significance tests assume independent observations (Dror et al., 2018); resampling generations would treat sixteen samples from one problem as sixteen independent observations and artificially narrow every interval. Contrasts use only problem identifiers shared by both arms. The rank-correlation null is exact: it enumerates every permutation of arm labels within each benchmark, preserving tied ranks. A $10{,}000$-draw Monte Carlo run gives the same result.

The study's noise floor comprises the six null contrasts among the reweighting arms. The largest null contrast is 0.58 points, and every reported significant OPTD contrast exceeds all six.