SHORTOPD: RECOVERING PRUNED LLMS WITH SHORT-TO-LONG ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Structured pruning is a hardware-friendly way to compress LLMs, but preserving recognition scores does not ensure useful free-form generation. On-Policy Distillation (OPD) offers a recovery signal by having the frozen pre-compression model supervise the student's own response prefixes. In the pruning regime we study, however, long rollouts frequently develop repetitive suffixes. The teacher can also assign these suffixes high probability, so low distillation loss can coexist with poor generation. We propose **ShortOPD**, an OPD rollout schedule guided by terminal repetition and clean truncation. Repetition permits reducing the next budget toward an estimated prefix length, whereas non-repetitive responses that reach the cap permit growth. The controller reuses existing tokens and teacher scores, requires no additional model forward pass, and preserves the current batch's full-token distillation update. Its short-to-long phase follows an initial contraction from the maximum budget. Across four Qwen3 model settings with 25% of layers removed, ShortOPD attains 82.3–84.6% of the teacher's eight-task aggregate generation score, outperforming Vanilla OPD by 11.4–18.2 points. Under an 8192-token ceiling, it also uses 14.7–30.2% fewer rollout tokens while scoring 13.95–16.21 points higher. We hope this recipe helps move structured pruning beyond marginal gains on perplexity and multiple-choice benchmarks, a step closer to deployment-ready generation quality.

1 INTRODUCTION

Serving LLMs at full size is expensive (Grattafiori et al., 2024; Qwen Team, 2025), and structured pruning is one of the most deployment-friendly ways to cut this cost: it removes coupled parameter blocks while preserving ordinary dense transformer execution (Ma et al., 2023; Ashkboos et al., 2024; Men et al., 2025; Xia et al., 2024; Muralidharan et al., 2024), with no need for the dedicated kernel or hardware support that unstructured sparsity and low-bit quantization lean on (Frantar & Alistarh, 2023; Sun et al., 2024; Lin et al., 2024). The obstacle is evaluation: most pruning work (Ma et al., 2023; Ashkboos et al., 2024; Men et al., 2025; An et al., 2024; Kim et al., 2024; Gromov et al., 2025; Muralidharan et al., 2024) reports strong retention on multiple-choice recognition benchmarks such as MMLU and HellaSwag (Hendrycks et al., 2021a; Zellers et al., 2019), yet the same compressed checkpoints collapse on the free-form generation that deployment actually requires (He et al., 2026; Wang et al., 2025; Shrestha et al., 2026). Removing just 4 of 36 layers from Qwen3-4B-Instruct with Block-Influence (BI) depth pruning (Men et al., 2025) already devastates generation across math, code, and open-ended tasks (Table 1, $k{=}1$ versus teacher greedy). This recognition-generation gap is what blocks practical use of structured pruning.

Two observations identify the recovery signal. First, the missing ability is demoted rather than erased. Wen et al. (2026) show this for short QA answers, and we find that the same holds for entire generation trajectories: in Table 1, best-of-k performance climbs steadily with more sampled attempts on every benchmark, and on GSM8K (Cobbe et al., 2021) it even reaches 91% at $k{=}64$, above the unpruned teacher's greedy 88%. Correct trajectories thus remain in the compressed model's own sampling distribution, within reach of on-policy search. Second, the repair must be token-level: ShortGPT-gen restores much of the lost quality by routing only *generated* tokens through the full model (Men et al., 2025), consistent with generation failure being a chain of local next-token mis-rankings that compound along the student's own decoding path (Ranzato et al., 2016; Agarwal et al., 2024; He et al., 2026). Sparse rewards (Shao et al., 2024; DeepSeek-AI,

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.

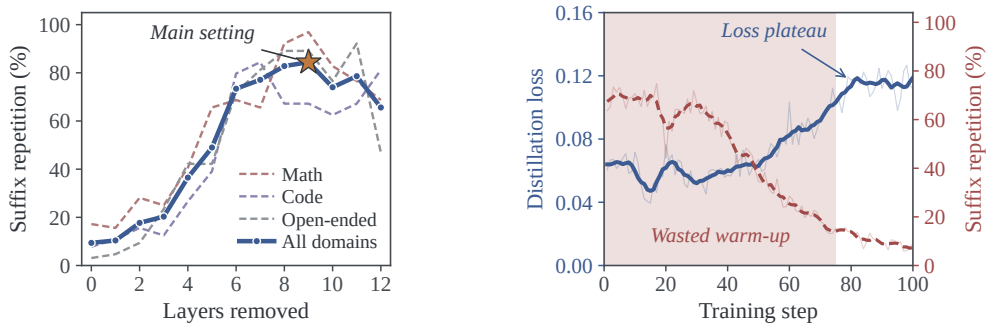
Table 1: Sampling diagnostic for Qwen3-4B-Instruct with 4 of 36 layers removed. Each cell reports the oracle best-of-k score over k samples. A math/code problem counts as solved (%) if any sample is correct, and open-ended tasks take the maximum LLM-judge score. Teacher is the unpruned model's greedy score, shown in gray for reference. Cell shading grows with the fraction of the teacher score reached, and **bold** marks best-of-k scores that match or exceed the teacher.

	Pruned (-4 layers), best-of-k							
Benchmark	k=1	2	4	8	16	32	64	Teacher
Math								
GSM8K	49.0	62.7	72.9	80.1	85.2	**88.6**	**91.2**	88.1
MATH-500	17.7	25.6	33.4	40.5	46.9	53.0	58.6	89.4
Code								
HumanEval	2.7	4.1	6.0	8.4	11.6	15.4	20.1	75.0
MBPP	6.6	10.6	15.2	20.2	25.3	30.4	35.0	80.5
Open-ended								
Alpaca	2.91	3.74	4.34	4.76	5.09	5.39	5.75	6.64
QA	0.33	0.56	0.84	1.14	1.43	1.76	2.13	4.38
Sum	3.36	4.66	5.85	6.91	7.80	8.44	**8.88**	8.86
MT-Bench	2.19	2.92	3.63	4.27	4.83	5.35	5.88	6.95

2025) and fixed off-policy labels (Kim & Rush, 2016) leave these local errors unspecified. Recovery therefore needs dense distributional targets on the states the compressed model actually visits.

OPD (Agarwal et al., 2024; Yuan et al., 2026) is the direct instantiation of these requirements. The compressed student samples rollouts from its own distribution. Its frozen pre-compression self scores the same trajectories as teacher, and the student matches the teacher's next-token distribution at every response position. The recipe is on-policy and dense, and it needs no labels, no verifier, and no external teacher. The remaining difficulty is that, consistent with recent evidence of pruning-induced reasoning loops (Wang et al., 2025), early on-policy rollouts are dominated by suffix repetition (Holtzman et al., 2020; Xu et al., 2022; Pipis et al., 2026; Hiraoka & Inui, 2025; Duan et al., 2026). Probing BI-pruned Qwen3-4B-Instruct on a fixed 192-prompt math/code/open-ended set, suffix repetition reaches 84% at our 25%-depth testbed (Figure 1a). With stronger compression, outputs turn incoherent instead of looped (Appendix A). At 25% depth reduction, failures are still dominated by repetitive suffixes, so we adopt it as our main testbed. These loop tails also carry little measured marginal distillation signal: in a diagnostic sample of 12 high-repetition generations, repeated-suffix tokens have a mean full-vocabulary reverse $\mathrm{KL}(\pi_S \| \pi_T)$ of 0.0549 versus 0.6012 on ordinary tokens (about $11\times$ lower), and a mean teacher NLL of 3.24×10^{-5} versus 0.678. Together with the conditional gradient analysis in Appendix B, these measurements identify deep repeated suffixes as low-information under the implemented OPD update: they consume early rollout and teacher compute under a long, fixed rollout horizon H (the per-response token budget) while providing little additional teacher–student correction. Worse, because the OPD loss is averaged over the whole response, these near-zero-gradient tokens also dilute the correction on the informative prefix that precedes them. Figure 1(b) follows the first 100 steps of a fixed-H=2048 OPD run: during the early phase, 55–75% of rollouts end in suffix loops and the distillation loss stays near its minimum, rising only as repetition subsides over steps 50–80.

We propose ShortOPD (Short-to-Long On-Policy Distillation) to make OPD spend its early budget where the supervision is useful. A lightweight controller tracks suffix repetition, clean truncation, and mean effective length. High repetition opens the shrink gate, whose target is an exponential moving average (EMA) of effective length, while low repetition and high clean truncation open the growth gate. A second EMA moves the actual budget toward either target: the first smooths noisy batch estimates, while the second prevents abrupt horizon changes. Training therefore contains repetitive early rollouts and restores long generation as recovery proceeds, at no extra forward cost. In practice, ShortOPD narrows the high-repetition region and reduces tokens spent in repeated suffixes (Figure 3). It improves generation scores over Vanilla OPD (Table 3) and reduces rollout cost in separate 8192-ceiling comparisons (Table 5).



(a) Suffix repetition vs. pruned depth

(b) Repetitive suffixes coincide with low loss

Figure 1: Repetition during early on-policy recovery. **(a)** Suffix-repetition rate versus BI-pruned depth on a fixed 192-prompt probe, where the star marks our main 9-layer setting. **(b)** Distillation loss (blue) and suffix-repetition rate (red) during a fixed-H=2048 diagnostic run, with shading marking the repetition-dominated warm-up. Figure 3(a) overlays ShortOPD.

Our contributions are threefold:

- **We propose ShortOPD**, a recipe for recovering structurally compressed LLMs built on our diagnosis of what recovery requires: supervision on the student's own on-policy states, dense token-level teacher distributions, and a rollout budget matched to what the damaged policy can currently generate. ShortOPD delivers the third ingredient with repetition-gated, truncation-aware horizon control.
- **Extensive experiments demonstrate the effectiveness and efficiency of this recovery recipe.** Across four Qwen3 model settings with 25% of layers removed, ShortOPD reaches 82.3–84.6% of the unpruned teacher's generation score over eight tasks, outperforming Vanilla OPD by 11.4–18.2 points and exceeding SFT w/o KD, KD, and SeqKD. On a shared Math+Code corpus, it also outperforms the tested PPO/GRPO recovery recipes. Separate 8192-ceiling comparisons use 14.7–30.2% fewer rollout tokens and 15.3–24.8% less training time while raising the aggregate score by 13.95–16.21 points. The recovery recipe also extends to width and hybrid pruning at both 4B and 8B scales.
- **The recovery corpus matters.** On-policy distillation only repairs the states the student's own search visits: leave-one-domain-out ablations show that removing math or code prompts sharply damages recovery on the corresponding capabilities. Corpus coverage is therefore a first-class design choice for post-compression recovery, not a detail.

2 Method

ShortOPD combines on-policy distillation from the pre-compression model, which supplies distributional targets on student-generated states, with feedback control of how far subsequent rollouts explore those states. Figure 2 shows the two loops.

2.1 On-policy recovery

Let π_θ be a structurally compressed student and π_T its frozen pre-compression teacher. Our main experiments remove the nine lowest Block-Influence layers (Men et al., 2025) out of 36, or 25% of transformer depth, and the recovery objective applies unchanged to the width and hybrid students evaluated in Section 3.5. For each prompt x, the student samples a response y and the teacher scores the same response prefixes. The rollout-averaged loss is

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_x \, \mathbb{E}_{y\sim\pi_\theta(\cdot|x)} \left[\frac{1}{|y|} \sum_{t=1}^{|y|} \mathrm{KL}\big(\pi_\theta(\cdot \mid x, y_{<t}) \,\big\|\, \pi_T(\cdot \mid x, y_{<t})\big) \right]. \tag{1}$$

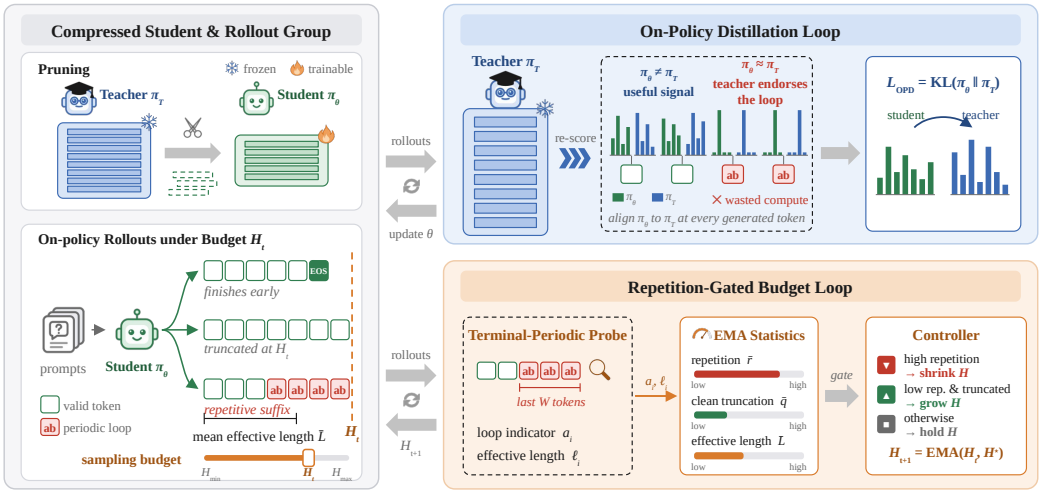


Figure 2: ShortOPD couples two training loops. In the on-policy distillation loop (blue), the frozen teacher re-scores student rollouts and supplies distributional targets on every generated token. In the repetition-gated budget loop (orange), terminal repetition and clean truncation set the next rollout budget: high repetition shrinks it toward a prefix-length estimate, and low repetition with frequent clean truncation grows it.

Gradients are taken through the per-token losses with sampled rollouts held fixed. Reverse-KL objectives have been studied in on-policy distillation (Gu et al., 2024; Agarwal et al., 2024). Every generated token receives a distributional target, including tokens in responses whose final answers are incorrect. Recovery therefore uses only prompts and the frozen teacher, without reference responses or a reward verifier. At the algorithmic level, ShortOPD adds rollout-budget feedback to the Vanilla OPD objective.

2.2 Short-to-Long On-Policy Distillation

A fixed rollout budget faces a dilemma under strong compression: a long budget reaches repetitive suffixes before the student recovers, while a short one truncates legitimate long generations later. ShortOPD instead adapts a per-step budget $H_s \le H_{\max}$ from rollout feedback. Training starts at $H_{\max}$, contracts while repetition dominates, and then grows back, so the short-to-long phase follows an initial contraction.

Generation-side feedback. For rollout y_i at step s, a terminal-periodic detector inspects only its last W tokens: $a_i = 1$ if it finds a severe loop, and $b_i = 1$ if the rollout fills the budget without one ($|y_i| = H_s$, $a_i = 0$). The effective length ℓ_i is the loop onset, locally refined with the OPD divergence and teacher NLL ($\ell_i = |y_i|$ if no loop is found). The batch statistics and their EMAs are

$$r_s = \frac{1}{B}\sum_{i=1}^{B} a_i, \quad q_s = \frac{1}{B}\sum_{i=1}^{B} b_i, \quad L_s = \frac{1}{B}\sum_{i=1}^{B} \ell_i, \quad (\bar{r}_s, \bar{q}_s, \bar{L}_s) = \mathrm{EMA}_\beta(r_s, q_s, L_s). \tag{2}$$

These statistics reuse the tokens and teacher scores of OPD, requiring no extra forward pass, and only steer future budgets while the current batch keeps the full OPD loss.

Repetition-gated horizon control. Starting from $H_1 = H_{\max}$, the controller forms a gated target H_s^* and moves the budget toward it:

$$H_s^* = \begin{cases} \min(H_s, \lambda \bar{L}_s), & \bar{r}_s > \rho_{\text{high}}, \\ \gamma_\uparrow H_s, & \bar{r}_s < \rho_{\text{low}} \wedge \bar{q}_s > \tau, \\ H_s, & \text{otherwise}, \end{cases} \qquad H_{s+1} = \mathrm{clip}\big(\beta_H H_s + (1-\beta_H) H_s^*\big). \tag{3}$$

Here $\lambda > 1$ leaves headroom beyond the usable prefix, $\gamma_\uparrow > 1$ sets the growth rate, β_H smooths both directions, and clip keeps the budget within $[H_{\min}, H_{\max}]$. The hysteresis band $\rho_{\text{low}} < \rho_{\text{high}}$

Table 2: Recovery methods used in this study. Distributional targets mean matching teacher token distributions. Label-free means no required reference responses, and verifier-free means no correctness reward. The final column denotes horizon control from rollout feedback, and — marks methods without on-policy rollouts.

Method	On-policy states	Distributional targets	Label-free	Verifier-free	Horizon control
SFT w/o KD	×	×	×	✓	—
SeqKD (Kim & Rush, 2016)	×	×	✓	✓	—
KD (Hinton et al., 2015)	×	✓	×	✓	—
RLVR (Schulman et al., 2017; Shao et al., 2024)	✓	×	×	×	×
Vanilla OPD (Agarwal et al., 2024)	✓	✓	✓	✓	×
ShortOPD (ours)	✓	✓	✓	✓	✓

prevents oscillation. Because shrinking is gated by repetition, short clean responses never reduce the budget merely by lowering the average length.

Recovery signals. Table 2 places ShortOPD and its baselines along three axes: trajectory source (on-policy student rollouts vs. fixed data), supervision (dense teacher distributions vs. hard labels or sparse reward), and horizon control. Our experiments instantiate SFT w/o KD, SeqKD, KD, Vanilla OPD, and ShortOPD on the same prompt distribution, and add a matched Math+Code RLVR comparison for the sparse on-policy cell.

3 Experiments

We assess overall recovery, then the evidence for rollout control:

- **RQ1 (Effectiveness).** Compared with standard recovery recipes, how well does ShortOPD restore generation across model sizes, modes, and tasks? **(Table 3)**
- **RQ2 (Supervision).** On the same Math+Code recovery corpus, how does distributional supervision compare with sparse outcome rewards? **(Table 4)**
- **RQ3 (Rollout control).** What changes in quality, training behavior, and cost accompany rollout feedback relative to Vanilla OPD? **(Tables 3 and 5, Figure 3)**
- **RQ4 (Scope).** How does prompt coverage affect the capabilities recovered, and does recovery extend beyond layer deletion? **(Tables 6 and 7)**

3.1 Experimental setup

Models and recovery data. We use Qwen3-4B-Instruct-2507, Qwen3-4B-Thinking-2507, and Qwen3-8B in both modes (Qwen Team, 2025). Removing the nine lowest-Block-Influence layers (Men et al., 2025) gives 25% depth reduction, and the unpruned teacher stays frozen. The recovery corpus contains approximately 45k math, code, and instruction prompts disjoint from the evaluation sets (see Appendix D for its composition).

Baselines and budgets. SFT w/o KD trains on the original responses, SeqKD on responses generated by the teacher, and KD on the teacher's token distributions over these fixed contexts, with reasoning traces included for the Thinking models. KD minimizes forward KL, whereas OPD minimizes reverse KL, so these comparisons differ in objective, state source, and rollout exposure. Vanilla OPD uses a fixed budget $H = 2048$, and ShortOPD starts at 2048 and adapts within $[1024, 2048]$. All methods train for 710 steps with batch size 64 and learning rate 2×10^{-6}.

Evaluation. We measure answer accuracy on GSM8K/MATH-500 (Cobbe et al., 2021; Hendrycks et al., 2021b), execution accuracy on HumanEval/MBPP (Chen et al., 2021; Austin et al., 2021), and GPT-5.5 judge scores on Alpaca, QA, summarization, and MT-Bench (Zheng et al., 2023). Avg averages the eight tasks after multiplying judge scores by ten, and T% expresses this aggregate as a percentage of teacher Avg.

Table 3: Generation recovery (RQ1) and the Vanilla OPD comparison (RQ3) after 25% depth pruning (9/36 layers). Math/code scores are in %, and open-ended scores are on a 1–10 scale. Avg averages the eight tasks with judge scores scaled by ×10, and T% is relative to each dense teacher. Gray rows are references. **Bold** and underline mark the best and second-best recovery results per column within each block.

	Math		Code		Open-ended				Overall	
Method	GSM8K	MATH	HumanEval	MBPP	Alpaca	QA	Sum	MT-B	Avg	T%
Qwen3-4B-Instruct										
Teacher	88.10	89.40	75.00	80.54	6.64	4.38	8.86	6.95	75.17	100.0
Pruned	1.14	1.60	0.00	0.00	1.14	1.00	1.06	1.09	5.71	7.6
SFT w/o KD	25.93	9.40	26.83	29.18	1.91	1.19	3.12	1.59	21.18	28.2
SeqKD	32.83	12.20	29.88	41.25	2.52	1.38	5.26	2.10	28.60	38.0
KD	29.34	9.60	29.27	39.30	2.94	1.48	6.89	2.36	30.53	40.6
Vanilla OPD	61.87	45.00	45.12	51.75	4.61	2.83	7.42	4.28	49.39	65.7
ShortOPD	**75.06**	**65.20**	**57.93**	**64.20**	**5.79**	**3.64**	**8.21**	**5.63**	**61.89**	**82.3**
Qwen3-4B-Thinking										
Teacher	92.87	88.80	76.83	87.16	5.71	3.40	8.68	6.65	73.76	100.0
Pruned	0.45	1.20	0.00	0.00	1.00	1.00	1.00	1.01	5.22	7.1
SFT w/o KD	18.73	15.20	18.90	28.02	1.91	1.13	2.19	1.69	18.76	25.4
SeqKD	36.01	13.20	29.27	33.85	2.03	1.29	6.35	2.48	29.23	39.6
KD	44.81	17.80	29.27	38.91	3.10	1.41	7.04	3.48	35.14	47.6
Vanilla OPD	59.67	41.60	40.24	46.69	4.19	2.51	6.87	3.70	45.11	61.2
ShortOPD	**81.50**	**70.60**	**62.20**	**69.65**	**5.02**	**2.98**	**8.09**	**5.46**	**62.43**	**84.6**
Qwen3-8B (no-thinking)										
Teacher	93.10	81.80	86.59	80.54	6.63	4.26	8.80	6.95	76.05	100.0
Pruned	0.83	1.80	0.00	0.00	1.03	1.01	1.00	1.01	5.39	7.1
SFT w/o KD	39.58	10.80	15.24	21.01	2.16	1.51	3.10	2.03	21.83	28.7
SeqKD	62.32	26.00	28.66	33.07	2.59	1.33	4.03	2.53	31.86	41.9
KD	58.30	24.40	26.83	29.57	2.70	1.28	4.33	2.19	30.51	40.1
Vanilla OPD	70.28	44.20	52.44	54.86	4.68	2.91	7.45	4.41	52.04	68.4
ShortOPD	**84.31**	**61.80**	**70.12**	**66.15**	**5.71**	**3.52**	**8.28**	**4.97**	**63.40**	**83.4**
Qwen3-8B (thinking)										
Teacher	94.47	81.80	81.71	85.21	6.80	4.66	8.80	7.29	77.34	100.0
Pruned	0.15	2.00	0.00	0.00	1.01	1.00	1.00	1.01	5.29	6.8
SFT w/o KD	61.33	16.20	23.78	33.07	2.84	1.14	7.51	2.49	34.27	44.3
SeqKD	58.68	25.40	16.46	32.68	2.39	1.24	7.49	2.68	33.90	43.8
KD	58.68	23.80	24.39	29.18	3.33	1.66	6.70	2.81	35.13	45.4
Vanilla OPD	63.08	40.20	42.68	47.86	4.28	2.63	7.18	4.02	46.87	60.6
ShortOPD	**85.22**	**65.60**	**67.68**	**69.65**	**5.82**	**3.79**	**8.32**	**5.31**	**65.07**	**84.1**

3.2 RQ1: Effectiveness

Table 3 compares teachers, pruned students, and recovered variants across eight tasks.

ShortOPD recovers most of the generation ability that pruning removes. Removing a quarter of the layers reduces generation Avg to 5.22–5.71 in all four blocks, and every pruned student scores zero on HumanEval and MBPP. The best offline baseline recovers only 30.53–35.14 Avg. Fixed-response training recovers summarization comparatively well but leaves reasoning weak, and no offline recipe exceeds 26.0 on MATH-500.

On-policy supervision and rollout control contribute cumulatively. Vanilla OPD already exceeds the best offline baseline by 10.0–20.2 Avg, and ShortOPD adds another 11.4–18.2 points, reaching 82.3–84.6% of the teacher. ShortOPD leads on every task within each block. Its gains over Vanilla OPD concentrate on math and code (+11.3 to +29.0 points per task), whereas open-ended judge scores rise by 0.5–1.8 on the 1–10 scale. On average, the remaining gap to the teacher is largest on MATH-500 and MT-Bench, where ShortOPD reaches 72.9–80.2% and 71.5–82.1% of the teacher, and smallest on summarization (92.7–94.5%). RQ2 and RQ3 separate the contributions of the supervision signal and of rollout control.

Table 4: Distributional versus outcome supervision (RQ2) on the matched Math+Code corpus. Scores are accuracies (%), and Avg averages the four tasks. Gray rows are references from Table 3. **Bold** and underline mark the best and second-best recovery results per column within each block.

	Math		Code		
Method	GSM8K	MATH-500	HumanEval	MBPP	Avg
Qwen3-4B-Instruct					
Teacher	88.10	89.40	75.00	80.54	83.26
Pruned	1.14	1.60	0.00	0.00	0.69
PPO	0.00	0.00	0.00	0.00	0.00
GRPO	0.91	0.80	0.00	0.00	0.43
ShortOPD	**57.47**	**36.60**	**19.51**	**38.52**	**38.03**
Qwen3-8B (no-thinking)					
Teacher	93.10	81.80	86.59	80.54	85.51
Pruned	0.83	1.80	0.00	0.00	0.66
PPO	14.78	2.40	0.00	0.00	4.30
GRPO	0.99	2.20	0.00	0.00	0.80
ShortOPD	**79.23**	**42.00**	**42.68**	**42.41**	**51.58**

3.3 RQ2: SUPERVISION

We compare PPO, GRPO, and ShortOPD on a shared Math+Code corpus of approximately 30k prompts from the same pruned initialization. Each code prompt carries one hidden executable test. PPO/GRPO receive the resulting binary reward, whereas ShortOPD receives teacher token distributions on its rollouts. Appendix D gives the corpus composition and the remaining settings.

ShortOPD reaches 38.03/51.58 Avg at 4B/8B, versus at most 0.43/4.30 for PPO and GRPO (Table 4). Both reward-based recipes score zero on both code benchmarks, whereas ShortOPD recovers nonzero execution accuracy. Dense teacher distributions thus provide a far stronger recovery signal than sparse outcome rewards from such a collapsed starting point, where successful rollouts are too scarce for the reward to fire. RQ3 next examines rollout control within OPD.

3.4 RQ3: ROLLOUT CONTROL

Quality relative to Vanilla OPD. With a common 2048 ceiling, ShortOPD improves Avg over Vanilla OPD by 12.50, 17.32, 11.36, and 18.20 on 4B Instruct, 4B Thinking, 8B no-thinking, and 8B thinking (Table 3). Adaptive budgeting thus consistently outperforms a fixed 2048-token budget across model sizes and thinking modes. Under a fixed budget, repetitive suffixes that run to the cap dilute the length-normalized OPD gradient on the informative prefix, whereas ShortOPD keeps rollouts close to that prefix and concentrates each update on it (Appendix B).

Does the budget follow the intended feedback? In Figure 3, the budget falls from 2048 to 1024 during early repetition and returns as clean truncations become frequent. Repetitive-suffix tokens fall sooner, mean loss reaches its later range after 40–50 steps versus 80, and generation time over the first 100 steps falls from 35.4 to 25.1 seconds per step (29%). These curves confirm that the controller tracks the rollout regime and removes wasted early generation.

Table 5: Rollout cost and quality under an 8192-token ceiling (RQ3).

Method	Tokens (M)	Time (h)	Avg
Qwen3-4B-Instruct			
Vanilla OPD	869	35.9	50.17
ShortOPD	**741**	**30.4**	**64.12**
Δ	−14.7%	−15.3%	+13.95
Qwen3-8B (no-thinking)			
Vanilla OPD	659	51.3	49.12
ShortOPD	**460**	**38.6**	**65.33**
Δ	−30.2%	−24.8%	+16.21

What quality–cost trade-off is observed? Under an 8192-token ceiling (Table 5), ShortOPD cuts rollout tokens by up to 30% and training time by up to 25% while raising Avg by 13.95–16.21 points over Vanilla OPD, improving quality and cost at the same time.

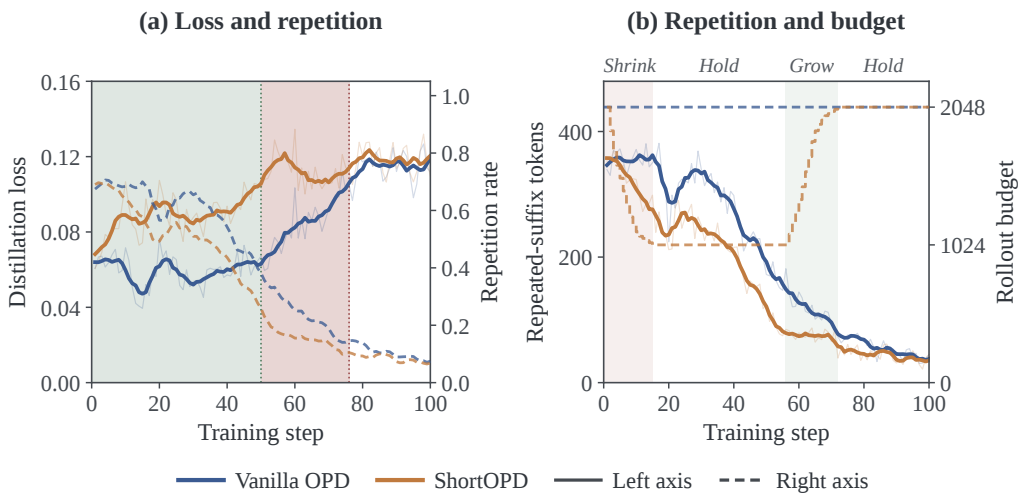


Figure 3: How the budget responds during recovery (RQ3), in a separate 2048-ceiling diagnostic. **(a)** Mean distillation loss and suffix-repetition rate. **(b)** Repetitive-suffix tokens and rollout budget. Solid/dashed lines use the left/right axes, respectively. ShortOPD moves from 2048 to 1024 and back as the observed regime changes. Shading marks ShortOPD's warm-up (green) and Vanilla OPD's extra warm-up (pink) in (a), and the shrink (red) and grow (green) phases in (b).

3.5 RQ4: Scope

Table 6: Leave-one-domain-out recovery-corpus ablation (RQ4) for ShortOPD on the compressed Qwen3-4B-Instruct at matched prompt count and steps. Shaded red cells mark the domain most directly damaged by the removed slice. Judge scores are normalized by $\times 10$ in Avg, and T% is relative to the Instruct teacher.

	Math		Code		Open-ended				Overall	
Recovery corpus	GSM8K	MATH	HumanEval	MBPP	Alpaca	QA	Sum	MT-B	Avg	T%
Full mixture	75.06	65.20	57.93	64.20	5.79	3.64	8.21	5.63	61.89	82.3
w/o math	8.04	6.00	28.66	43.19	3.96	1.73	7.19	2.94	30.51	40.6
w/o code	58.45	33.40	0.61	2.33	4.47	2.15	7.88	3.99	34.96	46.5
w/o general	54.21	28.80	48.17	50.19	3.01	1.51	5.41	3.15	39.02	51.9

Changing the recovery data mixture. Table 6 studies leave-one-domain-out corpora for the same pruned Qwen3-4B-Instruct, preserving prompt count and 710 steps by resampling the remaining domains. Removing math drops GSM8K from 75.06 to 8.04 and MATH-500 from 65.20 to 6.00, whereas removing code leaves HumanEval/MBPP at 0.61/2.33 and also lowers math. Removing general instructions lowers every open-ended score and also lowers math and code. Prompt coverage thus shapes the states visited during recovery, including across domains, and is a first-class design choice.

Changing the pruning structure. Table 7 tests whether the advantage of on-policy over off-policy recovery carries over to other pruning structures. Following Minitron (Muralidharan et al., 2024; Sreenivas et al., 2024), we build width-only and hybrid width–depth students at approximately 25% parameter reduction and recover each from identical pruned initializations, comparing ShortOPD with Minitron's own off-policy KD recovery. Pruning lowers generation Avg to 9.56–15.26, and ShortOPD raises it to 50.44–53.94, exceeding Minitron by 7.44–11.10 points. The recovery recipe thus extends beyond layer deletion to width and hybrid pruning.

Table 7: Generation recovery under width-only and hybrid pruning (RQ4). Parentheses give the actual parameter reduction (architectures in Appendix D). Columns and T% follow Table 3. Gray rows are references, and **bold** marks the best recovery per column within each pruned configuration.

		Math		Code		Open-ended				Overall	
Structure	Method	GSM8K	MATH	Human Eval	MBPP	Alpaca	QA	Sum	MT-B	Avg	T%
Qwen3-4B-Instruct											
Dense	Teacher	88.10	89.40	75.00	80.54	6.64	4.38	8.86	6.95	75.17	100.0
Width (25.1%)	Pruned	2.12	5.80	0.61	0.00	1.23	1.07	3.24	1.26	9.56	12.7
	Minitron	47.69	18.80	27.44	37.35	5.44	2.16	8.18	4.08	41.22	54.8
	ShortOPD	**71.04**	**44.40**	**35.98**	**43.97**	**5.70**	**2.34**	**8.19**	**4.59**	**50.44**	**67.1**
Hybrid (25.1%)	Pruned	8.42	4.40	1.22	4.28	2.35	1.14	5.33	1.56	15.26	20.3
	Minitron	52.46	24.20	37.20	45.53	4.85	1.99	7.91	3.58	42.84	57.0
	ShortOPD	**73.62**	**51.20**	**45.12**	**55.25**	**5.58**	**2.17**	**8.11**	**4.77**	**53.94**	**71.8**
Qwen3-8B (no-thinking)											
Dense	Teacher	93.10	81.80	86.59	80.54	6.63	4.26	8.80	6.95	76.05	100.0
Width (25.0%)	Pruned	3.03	4.20	0.00	0.00	1.56	1.11	5.14	1.43	12.45	16.4
	Minitron	59.36	24.00	39.63	45.91	5.30	2.30	8.20	4.17	46.08	60.6
	ShortOPD	**73.16**	**43.20**	**45.73**	**52.92**	**5.79**	**2.44**	**8.24**	**4.86**	**53.52**	**70.4**
Hybrid (25.1%)	Pruned	3.56	3.00	4.27	10.51	1.52	1.09	2.38	1.41	10.67	14.0
	Minitron	60.35	27.20	43.29	50.19	4.72	1.57	8.12	4.04	45.71	60.1
	ShortOPD	**73.09**	**43.40**	**47.56**	**56.03**	**5.45**	**2.09**	**8.47**	**4.85**	**53.59**	**70.5**

4 Related Work

Recovery after structured pruning. Depth and width pruning remove blocks or coupled structures while retaining dense execution (Men et al., 2025; Kim et al., 2024; Gromov et al., 2025; Yuan et al., 2025; Ma et al., 2023; Ashkboos et al., 2024), and Sheared LLaMA, Minitron, and self-data distillation follow pruning with continued training or distillation (Xia et al., 2024; Muralidharan et al., 2024; Sreenivas et al., 2024; Thangarasa et al., 2025). These methods are mostly evaluated on multiple-choice recognition benchmarks, with far less attention to free-form generation. Yet retained recognition scores can conceal substantial generation damage (Wen et al., 2026; He et al., 2026; Wang et al., 2025; Shrestha et al., 2026).

On-policy distillation and rollout control. Unlike sequence-level KD (Kim & Rush, 2016), which fits fixed teacher responses, on-policy distillation supervises student-generated states (Lin et al., 2020; Gu et al., 2024; Agarwal et al., 2024; Ko et al., 2024; Yuan et al., 2026). OPD variants that control how far these states extend rely on signals that break down on pruned students. Prefix OPD follows a preset prefix schedule (Zhang et al., 2026), which cannot shorten when rollouts collapse into loops. Prune-OPD and KAT rely on teacher–student compatibility or KL agreement (Yang et al., 2026; Xin et al., 2026), yet the teacher also assigns high probability to repetitive suffixes, so low divergence cannot distinguish a teacher-endorsed loop from a confident correct continuation. ShortOPD instead gates the budget with a structural repetition signal that the teacher cannot mask.

5 Conclusion

Structured pruning keeps models dense and fast, but it damages free-form generation far more than recognition benchmarks reveal. We trace this damage to a concrete failure mode: pruned students collapse into repetitive suffixes that the teacher endorses, so on-policy distillation wastes its early budget on nearly signal-free tokens. ShortOPD turns this observation into a controller that shrinks the rollout budget while repetition dominates and restores it as the student recovers, at no extra forward cost. Across four Qwen3 model settings, ShortOPD recovers 82.3–84.6% of the teacher's generation quality, outperforms Vanilla OPD by 11.4–18.2 points, and cuts rollout tokens by up to 30%. The recipe transfers to width and hybrid pruning, with corpus coverage as a key design choice. We hope these results steer compressed-model recovery toward what the student actually generates.

AI USE STATEMENT

In this work, we used generative AI tools to assist in writing proofs, namely to check and revise the derivation in Appendix B, and to translate our Chinese drafts into English. No other task requiring disclosure under the ICLR 2027 AI policy involved generative AI. Additionally, we used generative AI tools to draft and revise parts of the paper, write plotting code, and format tables and figures. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 2 defines the objective and horizon update. Appendices D and C give configurations and pseudocode.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations (ICLR)*, 2024.

Wasi Uddin Ahmad, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Vahid Noroozi, Somshubra Majumdar, and Boris Ginsburg. OpenCodeInstruct: A large-scale instruction tuning dataset for code LLMs. *arXiv preprint arXiv:2504.04030*, 2025.

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive structured pruning for large language models. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

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.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

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 (NeurIPS)*, 2015.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. Streamlining redundant layers to compress large language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90% chatgpt quality. `https://lmsys.org/blog/2023-03-30-vicuna/`, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Zenghao Duan, Liang Pang, Zihao Wei, Wenbin Duan, Yuxin Tian, Shicheng Xu, Jingcheng Deng, Zhiyi Yin, and Xueqi Cheng. Circular reasoning: Understanding self-reinforcing loops in large reasoning models. *arXiv preprint arXiv:2601.05693*, 2026.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning (ICML)*, 2023.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, 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: On-policy distillation of large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Shwai He, Guoheng Sun, Haichao Zhang, Yun Fu, and Ang Li. Demystifying when pruning works via representation hierarchies. In *International Conference on Machine Learning (ICML)*, 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021a.

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 *NeurIPS Datasets and Benchmarks*, 2021b.

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

Tatsuya Hiraoka and Kentaro Inui. Repetition neurons: How do language models produce repetitions? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 483–495, 2025. doi: 10.18653/v1/2025.naacl-short.41.

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

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. Shortened llama: Depth pruning for large language models with comparison of retraining methods. *arXiv preprint arXiv:2402.02834*, 2024.

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

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. In *International Conference on Machine Learning (ICML)*, 2024.

Steven Kolawole, Lucio Dery, Jean-François Kagy, Virginia Smith, Graham Neubig, and Ameet Talwalkar. Everybody prune now: Structured pruning of llms with only forward passes. *arXiv preprint arXiv:2402.05406*, 2024.

Eldar Kurtić, Elias Frantar, and Dan Alistarh. ZipLM: Inference-aware structured pruning of language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, et al. Tulu 3: Pushing frontiers in open language model post-training. In *Conference on Language Modeling (COLM)*, 2025.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.

Kevin Lu and Thinking Machines Lab. On-policy distillation. Thinking Machines Lab: Connectionism, 2025. https://thinkingmachines.ai/blog/on-policy-distillation/.

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.

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*, 2025.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Charilaos Pipis, Shivam Garg, Vasilis Kontonis, Vaishnavi Shrivastava, Akshay Krishnamurthy, and Dimitris Papailiopoulos. Wait, wait, wait... why do reasoning models loop? In *International Conference on Machine Learning (ICML)*, 2026.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Safal Shrestha, Anubhav Shrestha, Aadim Nepal, Minwu Kim, and Keith Ross. On the limits of layer pruning for generative reasoning in large language models. *arXiv preprint arXiv:2602.01997*, 2026.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. SLEB: Streamlining LLMs through redundancy verification and elimination of transformer blocks. In *International Conference on Machine Learning (ICML)*, 2024.

Sharath Turuvekere Sreenivas, Saurav Muralidharan, Raviraj Joshi, Marcin Chochowski, Ameya Sunil Mahabaleshwarkar, Gerald Shen, et al. Llm pruning and distillation in practice: The minitron approach. *arXiv preprint arXiv:2408.11796*, 2024.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Vithursan Thangarasa, Ganesh Venkatesh, Mike Lasby, Nish Sinnadurai, and Sean Lie. Self-data distillation for recovering quality in pruned large language models. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 7, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/file/af2d9fb5bcee19ef2dfa70d843520c97-Paper-Conference.pdf.

Keyu Wang, Tian Lyu, Guinan Su, Lu Yin, Marco Canini, Jonas Geiping, and Shiwei Liu. When fewer layers break more chains: Layer pruning harms test-time scaling in LLMs. *arXiv preprint arXiv:2510.22228*, 2025.

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 (ICLR)*, 2020.

Rui Wen, Lu Sun, Jiayang Liu, Zesheng Xu, Tianshuo Cong, and Zheng Li. The benchmark illusion: Pruned llms can pass multiple choice but fail to answer. *arXiv preprint arXiv:2606.17609*, 2026.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. f-divergence minimization for sequence-level knowledge distillation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. In *International Conference on Learning Representations (ICLR)*, 2024.

Haoran Xin, Anhao Zhao, Ying Sun, Jin Li, Xiaoyu Shen, and Hui Xiong. Escaping the KL agreement trap in on-policy distillation, 2026. URL https://arxiv.org/abs/2606.09471.

Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*, 2024.

Yifei Yang, Zouying Cao, and Hai Zhao. Laco: Large language model pruning via layer collapse. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.

Zhicheng Yang, Zhijiang Guo, Yifan Song, Minrui Xu, Yongxin Wang, Yiwei Wang, Xiaodan Liang, and Jing Tang. Prune-OPD: Efficient and reliable on-policy distillation for long-horizon reasoning, 2026. URL https://arxiv.org/abs/2605.07804.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, et al. Dapo: An open-source llm reinforcement learning system at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Qianhao Yuan, Qingyu Zhang, Yanjiang Liu, Jiawei Chen, Yaojie Lu, Hongyu Lin, Jia Zheng, Xianpei Han, and Le Sun. Shortv: Efficient multimodal large language models by freezing visual tokens in ineffective layers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 329–339, 2025.

Qianhao Yuan, Jie Lou, Xing Yu, Hongyu Lin, Le Sun, Xianpei Han, and Yaojie Lu. Vision-opd: Learning to see fine details for multimodal llms via on-policy self-distillation. *arXiv preprint arXiv:2605.18740*, 2026.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Dongxu Zhang, Zhichao Yang, Sepehr Janghorbani, Jun Han, Andrew Ressler II, Qian Qian, Gregory D Lyng, Sanjit Singh Batra, and Robert E. Tillman. Fast and effective on-policy distillation from reasoning prefixes. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 25553–25569, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-acl.1276. URL https://aclanthology.org/2026.findings-acl.1276/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023.

A REPETITION VERSUS PRUNING DEPTH: THE FULL SWEEP

Repetition is a well-documented degeneration mode of neural language models (Holtzman et al., 2020; Welleck et al., 2020; Xu et al., 2022), and training on model-generated prefixes mitigates the resulting exposure bias (Bengio et al., 2015; Ranzato et al., 2016). Figure 1(a) shows the suffix-repetition rate for 0–12 removed layers, and Figure 4 extends the sweep to all 36 layers of Qwen3-4B-Instruct, with the distinct-2 diversity of the generations on the right axis. Three regimes emerge. While the model remains a coherent generator ($k \leq 12$), repetition rises with removed depth, peaking at 84% at $k{=}9$, while distinct-2 falls to its minimum: the model retains enough structure to sustain n-gram loops, and looping is its dominant degeneration mode. Beyond $k \approx 13$, outputs disintegrate into incoherent token sequences: distinct-2 rebounds toward that of random text and n-gram loops no longer form, so the measured repetition rate collapses even though the model is far more damaged. At the extreme ($k \geq 35$), the near-empty stack degenerates into trivial single-token loops and the repetition rate saturates at 100%. The probe, generation settings, and repetition detector are identical across all points. Table 8 shows short excerpts from the same probe: the 25% depth-reduction point is still structured enough to produce recognizable prefixes and repetitive suffixes, whereas deeper compression often leaves the repetition metric by entering an incoherent mixed-token regime instead.

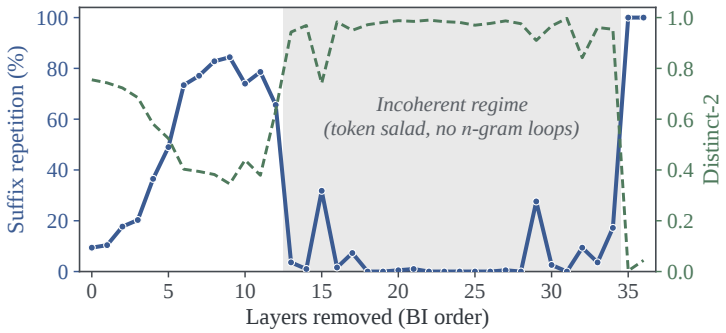


Figure 4: Full pruning sweep on the 192-prompt probe: suffix-repetition rate (left axis) and distinct-2 diversity (right axis) versus the number of BI-pruned layers. Repetition is the characteristic failure mode only in the coherent regime ($k \leq 12$), whereas deeper pruning collapses generation into incoherent text where loops no longer form.

Table 8: Probe excerpts illustrating the transition from normal generation to repetition and then to incoherent collapse. Green indicates usable content, and red indicates repeated or incoherent content.

Regime	Excerpt
Uncompressed	`Let's solve this step by step. We are told: Candice originally had 80 post-it notes... total post-its before placing on cups = post-its used + remaining.`
25% depth-pruned, loop	`Reason step step: step step step step solve step step step ... so.so.so. so.s.so.s ... ).").").")`
Deeper compression, incoherent	`Lonisolths~ "CCCCCCCC..." Spain possessions ... JSON:!QQVV ... @@## ... include_app ...`

Implication for recovery. The 25% depth-pruning setting used throughout the paper removes 9 of the 36 layers and therefore lies inside the coherent regime. Its failures are dominated by repetitive suffixes rather than incoherent text, and the student still produces a usable prefix before it starts looping. This is exactly the regime ShortOPD targets. The loop detector can locate where repetition begins, and the controller can shorten the rollout budget around that point so that on-policy supervision concentrates on the informative prefix. The sweep also shows why a structural repetition signal is reliable at this compression level, since looping remains the dominant degeneration mode for as long as the pruned model stays a coherent generator.

B CONDITIONAL OPD GRADIENT AT TEACHER–STUDENT AGREEMENT

This appendix makes precise why tokens with low per-token OPD loss supply little training signal: the per-token logit gradient is controlled by the square root of that loss. Condition on one sampled prefix $(x, y_{<t})$ and treat it as fixed during back-propagation, as in the OPD update of Section 2.1. Let p be the frozen teacher distribution over the vocabulary, $q = \mathrm{softmax}(z)$ the student distribution with logits z, and $r_i = \log(q_i/p_i)$ the per-token log-ratio. The per-token loss is the reverse KL

$$\ell = \mathrm{KL}(q\|p) = \sum_i q_i r_i = \mathbb{E}_q[r], \qquad \frac{\partial \ell}{\partial q_i} = r_i + 1. \tag{4}$$

The softmax Jacobian is $\partial q_i/\partial z_j = q_i(\mathbf{1}[i=j] - q_j)$, and $\sum_i q_i(\mathbf{1}[i=j] - q_j) = 0$, so the constant term drops out and the chain rule gives

$$\frac{\partial \ell}{\partial z_j} = q_j\,(r_j - \ell). \tag{5}$$

The logit gradient is thus the q-weighted deviation of the log-ratio from its mean, and by the Cauchy–Schwarz inequality

$$\|\nabla_z \ell\|_1 = \mathbb{E}_q\big[\,|r - \mathbb{E}_q r|\,\big] \le \sqrt{\mathrm{Var}_q(r)}. \tag{6}$$

Near agreement, write $q = p + \delta$ with $\sum_i \delta_i = 0$ and $|\delta_i| \le p_i/2$, so that $r_i = \delta_i/p_i + O(\delta_i^2/p_i^2)$. Both the loss and the log-ratio variance then reduce to the χ^2 divergence $\chi^2(q\|p) = \sum_i \delta_i^2/p_i$:

$$\ell = \tfrac{1}{2}\,\chi^2(q\|p)\,\big(1 + o(1)\big), \qquad \mathrm{Var}_q(r) = \chi^2(q\|p)\,\big(1 + o(1)\big), \tag{7}$$

where $o(1) \to 0$ as $\max_i |\delta_i|/p_i \to 0$. Combining with Equation (6),

$$\|\nabla_z \ell\|_1 \le \sqrt{2\ell}\,\big(1 + o(1)\big). \tag{8}$$

A token whose loss is small therefore also has a small logit gradient, shrinking with the square root of the loss and vanishing at exact agreement $q = p$. For model parameters θ, $\nabla_\theta \ell = J_{z,\theta}^\top \nabla_z \ell$, so the same holds for the parameter gradient whenever the local logit Jacobian is bounded. Repeated-suffix tokens, whose mean reverse KL is about $11\times$ lower than that of ordinary tokens (Section 1), therefore have a proportionally smaller gradient bound. This is a conditional, per-token statement rather than a claim that every small scalar loss implies a small full-objective gradient.

Implication for fixed-budget OPD. Equation (1) averages the per-token loss over the full response. Splitting a rollout y_i into an informative prefix of length ℓ_i and a repetitive suffix, the suffix contributes little gradient by Equation (8) but still enters the normalization, so the prefix gradient is scaled by $\ell_i/|y_i|$. Under a fixed budget, early rollouts that loop until the cap therefore dilute the correction on the states where errors originate. By moving the budget toward $\lambda \bar{L}_s$, ShortOPD raises this fraction to roughly $1/\lambda$ while discarding tokens that carry little signal, which concentrates each update on the informative prefix.

C CONTROLLER ALGORITHM AND LOOP DETECTOR

Algorithm 1 applies the OPD update to every generated token and uses terminal repetition, clean truncation, and effective length only to choose the next rollout budget.

C.1 TERMINAL PERIODIC-LOOP DETECTOR

Algorithm 2 gives the token-only detector used by the controller. It searches only the last $W{=}512$ response tokens and tests periods $p \in \{1, \ldots, 10\}$. For a given period, each token is compared with the token p positions earlier. This shifted comparison does not require the response to end on a cycle boundary, so a rollout truncated part-way through its final cycle is still detected. Requiring high agreement at the raw response end prevents repeated material followed by a clean ending from firing the controller.

The lexicographic choice prefers the longest explained terminal tail, then higher periodic agreement, then the shorter period. The detector alone decides whether a rollout is severely repetitive. Its

Algorithm 1 ShortOPD

Require: prompt set $\mathcal{D}$, batch size B, student π_θ, teacher π_T, bounds $H_{\min}, H_{\max}$, thresholds $\rho_{\text{low}}, \rho_{\text{high}}, \tau$, margin λ, growth $\gamma_\uparrow$, EMA decays β, β_H

$H_1 \leftarrow H_{\max}$; initialize EMAs
for training step $s = 1, \ldots, S$ **do**
  sample prompts $\{x_i\}_{i=1}^{B} \subset \mathcal{D}$ and rollouts $y_i \sim \pi_\theta(\cdot \mid x_i)$ with budget H_s
  $a_i \leftarrow \text{TerminalLoop}(y_i)$; $r_s \leftarrow B^{-1} \sum_i a_i$
  $q_s \leftarrow B^{-1} \sum_i \mathbf{1}[|y_i| = H_s \ \wedge \ a_i = 0]$
  run π_T on the same trajectories
  update θ with the dense OPD loss over all generated tokens (Eq. 1)
  compute effective lengths ℓ_i and mean L_s; update $\bar{r}_s, \bar{q}_s, \bar{L}_s$
  if $\bar{r}_s > \rho_{\text{high}}$ **then**
    $H_s^* \leftarrow \min(H_s, \lambda \bar{L}_s)$
  else if $\bar{r}_s < \rho_{\text{low}}$ **and** $\bar{q}_s > \tau$ **then**
    $H_s^* \leftarrow \gamma_\uparrow H_s$
  else
    $H_s^* \leftarrow H_s$
  end if
  $H_{s+1} \leftarrow \text{clip}_{[H_{\min}, H_{\max}]}(\lfloor \beta_H H_s + (1 - \beta_H) H_s^* \rceil_{16})$
end for

Algorithm 2 Severe terminal periodic-loop detection

Require: response tokens y, suffix window W=512, maximum period P=10, terminal anchor A=32, agreement η=0.9, minimum cycles C=3, minimum tail $L_{\min}$=64, severe length L_{sev}=128, severe fraction ϕ=0.30

$z \leftarrow \text{Last}_W(y)$; $n \leftarrow |z|$; $\mathcal{C} \leftarrow \varnothing$
for $p = 1, \ldots, \min(P, n-1)$ **do**
  $c_k^{(p)} \leftarrow \mathbf{1}[z_{p+k} = z_k]$ for $k = 1, \ldots, n-p$
  if $\text{Mean}(\text{Last}_A(c^{(p)})) < \eta$ **then**
    continue ▷ the periodic pattern does not reach the response end
  end if
  $k_p \leftarrow \max\{k : \text{Mean}(\text{Last}_k(c^{(p)})) \geq \eta\}$
  $L_p \leftarrow k_p + p$; $u_p \leftarrow \text{Mean}(\text{Last}_{k_p}(c^{(p)}))$
  if $L_p \geq L_{\min}$ **and** $L_p/p \geq C$ **and** ($L_p \geq L_{\text{sev}}$ **or** $L_p/n \geq \phi$) **then**
    add (L_p, u_p, p) to $\mathcal{C}$
  end if
end for
if $\mathcal{C} = \varnothing$ **then**
  return NOT-SEVERE, $|y|$
end if
$(L^*, u^*, p^*) \leftarrow \arg\max_{(L,u,p) \in \mathcal{C}} (L, u, -p)$
return SEVERE, $|y| - L^*, p^*$

returned onset is the structural effective-length candidate. We then search locally from that onset for the first 32-token window whose mean OPD loss and teacher NLL fall below the thresholds in Table 9, where $\bar{d}_{\text{base}}$ and $\bar{u}_{\text{base}}$ are the mean OPD loss and teacher NLL over the 64 tokens preceding the candidate onset. This search refines the boundary but does not decide whether repetition exists. If no loss-confirmed boundary is found, the structural onset is retained. A rollout contributes to the *clean truncation rate* only if it fills H_s and this detector does not mark it severely repetitive.

C.2 Controller hyperparameters

Table 9 lists the detector and controller hyperparameters, with symbols following Algorithms 1 and 2. The controller checkpoints its current budget and all EMA state across restarts.

Table 9: Detector and ShortOPD controller hyperparameters.

Loop detector		*Horizon controller*	
Suffix window W	512	Rate thresholds $(\rho_{\text{low}}, \rho_{\text{high}}, \tau)$	$(0.20, 0.45, 0.10)$
Candidate periods p	1–10	Statistic / budget EMA decays	0.7 / 0.7
Anchor / agreement (A, η)	$(32, 0.90)$	Margin λ / growth $\gamma_\uparrow$	1.15 / 1.25
Cycles / tail $(C, L_{\min})$	$(3, 64)$	Budget rounding	multiples of 16
Severe-loop tail	≥ 128 or 30% of suffix	Initial budget / bounds (main)	2048 / $[1024, 2048]$
Confirm / baseline window	32 / 64	Initial budget / bounds (8192)	8192 / $[1024, 8192]$
OPD-loss threshold	$\max(0.05,\ 0.25\,\bar{d}_{\text{base}})$		
Teacher-NLL threshold	$\max(0.5,\ 0.25\,\bar{u}_{\text{base}})$		

D Training details

D.1 Recovery corpus

The recovery corpus contains $45{,}446$ prompts across three domains:

- **Math**: GSM8K train (Cobbe et al., 2021) (7,473 prompts) and MATH train (Hendrycks et al., 2021b) (7,500), formatted as single-turn problems with a final-answer instruction, and MATH training examples are kept disjoint from MATH-500.
- **Code**: $15{,}000$ filtered NVIDIA OpenCodeInstruct (Ahmad et al., 2025) prompts (kept only with a positive unit-test signal, deduplicated, and with function names overlapping HumanEval or MBPP test tasks removed) plus 473 unique MBPP (Austin et al., 2021) non-test tasks.
- **Open-ended**: $15{,}000$ instruction prompts sampled from ShareGPT/Vicuna-style conversations and UltraChat-200K (Chiang et al., 2023; Ding et al., 2023).

No HumanEval task or MBPP test task is used for recovery training. All recovery methods share this prompt distribution: Vanilla OPD and ShortOPD use only the prompts, whereas the offline baselines train on the method-specific responses described in Section 3.1.

D.2 Pruning configurations

Let $X_{i,t}$ denote the hidden state of token t entering layer i. The Block Influence (BI) used to rank layers is

$$\mathrm{BI}_i \;=\; 1 - \mathbb{E}_{X,t}\left[\frac{X_{i,t}^{\top} X_{i+1,t}}{\|X_{i,t}\|_2\,\|X_{i+1,t}\|_2}\right], \tag{9}$$

so a low score indicates a small change to the hidden state. BI is calibrated on a held-out corpus, and the nine lowest-BI blocks of the 36-block Qwen3-4B-Instruct-2507 are removed, namely blocks 25–33 (zero-based, layers 26–34 one-based) before the retained blocks are renumbered at export. The Qwen3-4B-Thinking and Qwen3-8B experiments use the same $36 \to 27$ depth reduction, with BI calibrated separately for each backbone. The frozen original model serves as the teacher and the compressed student initializes the actor. Other depth-pruning criteria rank or merge blocks by redundancy (Song et al., 2024; Chen et al., 2025; Yang et al., 2024), and we adopt BI for its simplicity.

For the width/hybrid comparison of Section 3.5, importance scores are calibrated on held-out text and shared by the two exports of each teacher, following Minitron (Muralidharan et al., 2024; Sreenivas et al., 2024) and related structured-pruning methods (Kurtić et al., 2023; Kolawole et al., 2024). Table 10 gives the resulting architectures and exact unique-weight parameter counts. Attention head dimensions and the 32 query/8 key-value head layout are retained.

Table 10: Width-only and hybrid structural-pruning configurations. L, H, and I denote layer count, hidden size, and MLP intermediate size.

Model	Structure	L	H	I	Params	Reduction
Qwen3-4B Instruct	Width	36	2176	8000	3.013B	25.1%
	Hybrid	33	2240	8640	3.013B	25.1%
Qwen3-8B No-thinking	Width	36	3456	10240	6.147B	25.0%
	Hybrid	33	3584	10816	6.138B	25.1%

D.3 Main recovery runs (RQ1)

On-policy runs. Training uses the recovery corpus above, rollout group 8, train batch 64, learning rate 2×10^{-6}, and maximum prompt length 4096. The main schedule is 710 steps.

Offline baselines. SFT w/o KD, SeqKD, and KD train on the same prompt distribution with the method-specific fixed responses of Section 3.1, using sequence length 8192, batch 64, learning rate 2×10^{-6} for 710 steps. KD uses forward KL at temperature 1.0 with no hard-label cross-entropy term. These baselines follow standard LLM distillation practice (Hinton et al., 2015; Wen et al., 2023; Xu et al., 2024), and our on-policy runs follow recent OPD recipes (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025).

D.4 Math+Code comparison (RQ2)

PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), and ShortOPD train on one deterministic shuffle of $29{,}585$ prompts: $7{,}473$ GSM8K, $7{,}500$ MATH, $14{,}139$ verified OpenCodeInstruct, and 473 MBPP train-side examples. APPS is not used. Each code row retains exactly one executable test hidden from the policy prompt, and its reference solution must pass that test in two independent runs of the training sandbox before the row is admitted. PPO and GRPO receive this binary verifier reward, following the reinforcement learning with verifiable rewards recipe of recent reasoning systems (DeepSeek-AI, 2025; Yu et al., 2025; Lambert et al., 2025), whereas ShortOPD consumes only the shared prompt and the frozen teacher's token distributions. Every method uses no-thinking chat templates, batch 64, 8 rollouts per prompt, and 2048-token prompt and response ceilings. PPO uses GAE with a critic initialized from the compressed student (learning rate 10^{-5}), while GRPO uses group-relative advantages.

D.5 Efficiency runs (RQ3)

The Vanilla OPD and ShortOPD efficiency runs use the batch size, rollout count, and learning rate above. The $H \leq 2048$ runs provide the training trajectories shown in Figures 1(b) and 3. Table 5 compares the completed 4B and 8B no-thinking runs with the same 8192-token ceiling within each backbone. The shrink target $\bar{L}_s$ averages effective length over all rollouts, including those that end before the cap, so the budget can still contract under the 8192 ceiling.

E Multiple-choice recognition results

Although the main evaluation targets free-form generation, we also report held-out multiple-choice recognition on ARC-Challenge, HellaSwag, MMLU, and WinoGrande, reported as accuracy (%), none of which is part of the recovery corpus. The three subsections follow the order of the main experiments.

E.1 Depth pruning (RQ1)

Table 11 scores the 25% depth-pruned Qwen3-4B-Instruct by candidate log-likelihood. Pruning lowers the average from 76.66 to 45.65. ShortOPD restores it to 76.81, matching the teacher and exceeding SFT w/o KD (74.85), KD (74.37), and Vanilla OPD (72.02).

Table 11: Multiple-choice recognition for the 25% depth-pruned Qwen3-4B-Instruct (RQ1). Gray rows are references, and **bold** and underline mark the best and second-best recovery results.

Method	ARC-C	HellaSwag	MMLU	WinoG.	Avg
Teacher	89.93	80.60	70.93	65.19	76.66
Pruned	49.49	41.67	41.65	49.80	45.65
SFT w/o KD	89.33	76.89	69.09	64.09	74.85
SeqKD	83.36	72.65	64.19	63.06	70.82
KD	87.71	77.13	67.59	65.04	74.37
Vanilla OPD	85.24	75.69	65.28	61.88	72.02
ShortOPD	**89.93**	**79.67**	**69.99**	**67.64**	**76.81**

E.2 MATH+CODE COMPARISON (RQ2)

Table 12 reports multiple-choice recognition for the checkpoints of Table 4. ShortOPD again ranks first, reaching 66.84/74.90 Avg at 4B/8B versus 48.17/62.30 for PPO and 41.27/59.11 for GRPO.

Table 12: Multiple-choice recognition for the matched Math+Code comparison (RQ2, Table 4). **Bold** marks the best result per column within each block.

Backbone	Method	ARC-C	HellaSwag	MMLU	WinoG.	Avg
Qwen3-4B-Instruct	PPO	54.01	44.39	42.99	51.30	48.17
	GRPO	44.11	33.72	38.40	48.86	41.27
	ShortOPD	**79.10**	**67.81**	**59.75**	**60.69**	**66.84**
Qwen3-8B (no-thinking)	PPO	77.65	55.85	54.46	61.25	62.30
	GRPO	74.57	50.90	51.60	59.35	59.11
	ShortOPD	**88.65**	**79.53**	**67.87**	**63.54**	**74.90**

E.3 WIDTH AND HYBRID PRUNING (RQ4)

For the width and hybrid students of Table 7, we disable thinking and score candidate letters after a fixed assistant `ANSWER:` prefix by mean token log-probability (zero-shot MMLU, maximum length 4096). Table 13 shows that recognition degrades less than generation. The pruned 4B-width and 8B-hybrid models retain 33.71 and 52.11 Avg while reaching only 9.56 and 10.67 generation Avg. ShortOPD exceeds Minitron by 0.73–2.29 points here, smaller than its 7.44–11.10 generation gains, so fixed-option recognition is not a sufficient proxy for generation.

Table 13: Multiple-choice recognition under width-only and hybrid pruning (RQ4). Gray rows are references, and **bold** marks the best recovery result per column within each pruned configuration.

Structure	Method	ARC-C	HellaSwag	MMLU	WinoG.	Avg
Qwen3-4B-Instruct (teacher)		89.93	80.59	70.99	65.59	76.78
Width (25.1%)	Pruned	32.51	25.75	26.92	49.64	33.71
	Minitron	67.15	52.43	48.58	**58.80**	56.74
	ShortOPD	**70.05**	**57.02**	**50.24**	**58.80**	**59.03**
Hybrid (25.1%)	Pruned	55.46	31.23	37.25	51.62	43.89
	Minitron	77.05	60.50	55.74	**58.64**	62.98
	ShortOPD	**78.67**	**65.00**	**57.74**	58.48	**64.97**
Qwen3-8B (no-thinking) (teacher)		90.44	83.23	71.42	66.77	77.97
Width (25.0%)	Pruned	48.81	27.17	35.20	53.12	41.07
	Minitron	75.17	63.65	55.49	57.85	63.04
	ShortOPD	**76.11**	**66.90**	**56.32**	**58.56**	**64.47**
Hybrid (25.1%)	Pruned	65.70	39.19	46.41	57.14	52.11
	Minitron	79.52	63.90	59.29	60.38	65.77
	ShortOPD	**80.29**	**65.26**	**59.53**	**60.93**	**66.50**