ShortOPD: Recovering Pruned LLMs with Short-to-Long On-Policy Distillation

Qingyu Zhang[2,3], **Qianhao Yuan**[2,3], **Hongyu Lin**[2,†], **Yaojie Lu**[2], **Xianpei Han**[2,3], **Le Sun**[2,3], **Xiang Li**[1], **Ming Xu**[1], **Jiarui Li**[1], **Xiuying Zhao**[1]

[1]ByteDance, [2]Chinese Information Processing Laboratory, Institute of Software, Chinese Academy of Sciences, [3]University of Chinese Academy of Sciences

[†]Corresponding author

Abstract

Structured pruning is a hardware-friendly way to compress LLMs, but it is mostly validated on multiple-choice recognition tasks, while the same compressed checkpoints can collapse on the free-form generation that deployment actually requires. Two observations trace this gap. First, greedy PASS@1 nearly vanishes after compression, yet PASS@k recovers substantially under repeated sampling: useful generations are demoted, not erased. Second, the recoverable regime fails mainly through suffix repetition. Recovery should therefore train on the compressed model's own on-policy states with dense token-level supervision, which On-Policy Distillation (OPD) provides by reusing the pre-compression model as a frozen teacher. However, long on-policy rollouts spend early recovery budget on low-information repetitive suffixes, delaying loss descent. To mitigate this waste, we propose **ShortOPD**, a short-to-long OPD schedule that detects teacher-confirmed repetitive suffixes, treats the surviving prefix as each rollout's effective length, and allocates future rollout budgets to the effective lengths the policy can currently use. Across math, code, and open-ended generation, ShortOPD raises the compressed model's score to about 9× its unrecovered value and 1.6–4.4× standard recovery recipes (SFT w/o KD, KD, and SeqKD), and it matches a fixed 8192-token rollout horizon within two points using a quarter of the training time (8.5 vs. 35.9 hours) and 71% fewer rollout tokens. 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.

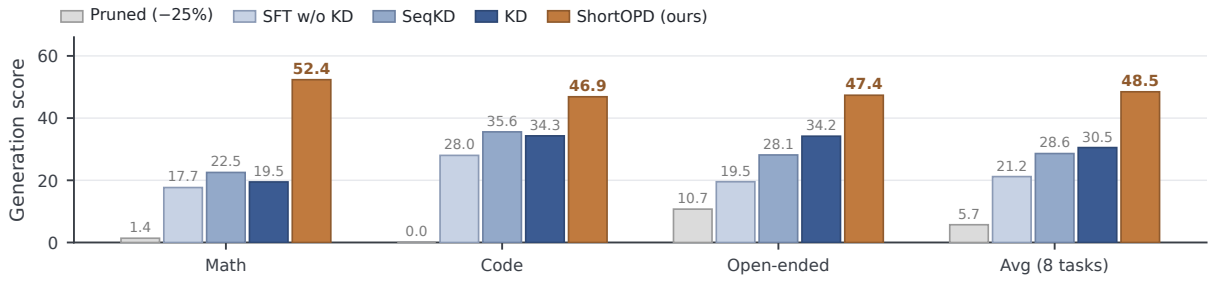


Figure 1 Headline recovery on 25%-pruned Qwen3-4B-Instruct: domain-mean generation scores from Table 3 (open-ended judge scores ×10). ShortOPD restores about two-thirds of the unpruned teacher's average of 75.2.

Table 1 Pruning collapses generation, but correct outputs stay reachable by sampling: best-of-k scores of Qwen3-4B-Instruct with 4 of 36 layers removed, against the number of sampled attempts k. Math/code report PASS@k (%); open-ended tasks report LLM-judge best-of-k scores on a 1–10 scale. Teacher is the unpruned model's greedy score. **Bold**: GSM8K crosses the teacher's greedy score under sampling—correct trajectories are demoted, not erased.

		Pruned (−4 layers), best-of-k							
Domain	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
	MATH500	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

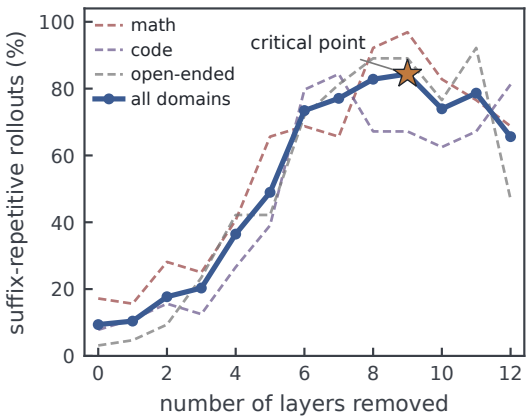
1 Introduction

Serving LLMs at full size is expensive [1, 2], 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 [3–7], with no need for the dedicated kernel or hardware support that unstructured sparsity and low-bit quantization lean on [8–10]. The obstacle is evaluation: most pruning work [3–5, 7, 11–13] reports strong retention on multiple-choice recognition benchmarks such as MMLU and HellaSwag [14, 15], yet the same compressed checkpoints collapse on the free-form generation that deployment actually requires. Removing just 4 of 36 layers from Qwen3-4B-Instruct [2] with Block-Influence depth pruning [5] already devastates greedy generation across math, code, and open-ended tasks (Table 1, PASS@1 versus teacher greedy). This recognition-generation gap is what blocks practical use of structured pruning.

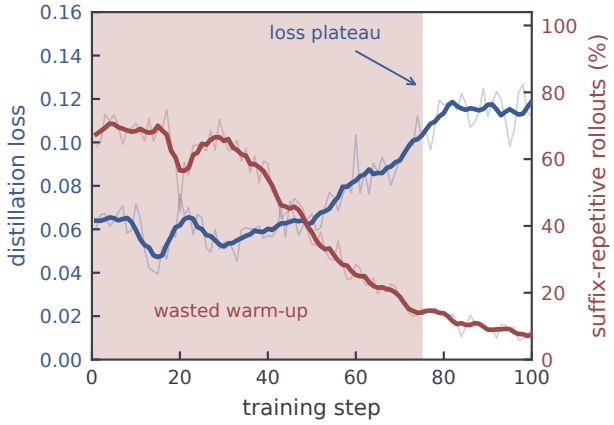
Two observations identify the recovery signal. First, the missing ability is demoted rather than erased. Wen et al. [16] show this for short QA answers; we find the same holds for entire generation trajectories: in Table 1, PASS@k [17] climbs steadily with more sampled attempts on every benchmark, and on GSM8K [18] 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 [5], so generation failure is a chain of local next-token mis-rankings that compound along the student's own decoding path [19, 20]. Sparse rewards [21, 22] and fixed off-policy labels [23] leave these local errors unspecified. Recovery therefore needs dense distributional targets on the states the compressed model actually visits.

OPD [20, 24] 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 early on-policy rollouts are dominated by repetition [25, 26]. Probing BI-pruned Qwen3-4B-Instruct on a fixed 192-prompt math/code/open-ended set, suffix repetition reaches 84% at our 25%-parameter testbed (Figure 2a); with stronger compression, outputs turn incoherent instead of looped (Appendix E), so we view 25% as this model's critical point for recoverable compression and adopt it as our main testbed. These loop tails also carry little measured marginal distillation signal: repeated-suffix tokens have a mean teacher–student generalized Jensen–Shannon divergence (JSD) of 0.0014 versus 0.051 on ordinary tokens (about 35× lower), and a mean teacher NLL of 3×10^{-5} versus 0.68. Together with the conditional gradient analysis in Appendix C, these measurements identify deep repeated suffixes as low-information under the implemented OPD update: they

The experiments in this paper can be reproduced with public implementations of the two building blocks: structured pruning follows https://github.com/icip-cas/ShortX, and on-policy distillation follows https://github.com/VisionOPD/Vision-OPD.



(a) Suffix repetition vs. pruned depth



(b) Repetitive suffixes coincide with low loss

Figure 2 Repetition during early on-policy recovery. **(a)** Suffix-repetition rate versus BI-pruned depth on a fixed 192-prompt probe; the star marks our main 9-layer setting. **(b)** Distillation loss (blue) and suffix-repetition rate (red) during fixed-H=2048 OPD; shading marks the repetition-dominated warm-up. Figure 4(a) overlays ShortOPD.

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. Figure 2(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 while the distillation loss remains at its lowest level, reaching its post-warm-up range only around step 80 as repetition subsides.

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, truncation, and mean effective length. High repetition opens the shrink gate, whose target is an exponential moving average (EMA) of effective length; low repetition and high truncation open the growth gate. A second EMA moves the actual budget toward either target; both 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, reduces tokens spent in repeated suffixes, and improves recovery over fixed short and fixed long horizons (Figure 1).

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.** On Qwen3-4B-Instruct with a 45,447-prompt math/code/instruction corpus, ShortOPD restores about two-thirds of the unpruned teacher's generation score over eight task families, far ahead of SFT w/o KD, KD, SeqKD, and sparse-reward RLVR under matched budgets, and it matches fixed rollout horizons of up to 8192 tokens within two points while generating up to 71% fewer rollout tokens; controlled comparisons confirm that the gain requires *both* on-policy states and dense teacher distributions.
- **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 Related Work

Structured pruning and the recognition-generation gap. Depth pruning is among the most serving-friendly LLM compression families: deleting whole transformer blocks preserves a standard dense architecture, with

layers selected by Block Influence (ShortGPT [5]), merged (LaCo [27]), or removed under other criteria [12, 13, 28]. Width pruning instead removes coupled structures inside layers [3, 4, 11, 29]; Sheared LLaMA [6] learns pruning masks with continued pre-training, and Minitron [7, 30] combines depth and width pruning with distillation-based retraining. One-shot weight sparsification (SparseGPT [8], Wanda [9]) is complementary but needs sparse-kernel support. Across these lines, validation is dominated by recognition-style benchmarks such as MMLU and HellaSwag [14, 15]. Wen et al. [16] show why this is deceptive: pruned models can still score multiple-choice answers yet fail to produce them, the missing answers demoted rather than erased. ShortGPT-gen makes the generative side concrete by routing only *generated* tokens through the full depth, locating the damage along the decoding path [5]. We extend the demotion view from short answers to entire reasoning trajectories and focus on the step these works leave open: after the layers are gone, which training signal best restores generation?

Distillation and on-policy recovery signals. Knowledge distillation [31] trains a student to match a teacher; sequence-level KD [23] trains on fixed teacher-generated responses and is therefore off-policy. A line of work closes the resulting train-inference mismatch by distilling on student states: imitation-learning KD [32], MiniLLM [33] with a reverse KL divergence objective, divergence generalizations [34, 35], and generalized on-policy KD [20]. Recovery retraining after pruning, however, still mostly follows the off-policy recipe: Sheared LLaMA continues pre-training on corpus data [6], and Minitron distills teacher logits on a fixed data blend and validates on recognition suites [7, 30], so whether off-policy logit distillation repairs *generation* is left open; our KD baseline instantiates exactly this recipe for a controlled comparison (Section 4.2). RL with verifiable rewards [21, 22, 36, 37] is on-policy but sparse: it offers no token-level target when a heavily compressed model rarely samples a correct answer. Vision-OPD [24] introduced on-policy distillation for multimodal models; we study the same principle after structural compression, where the privileged teacher is the model's own pre-compression self. All of these methods distill a *healthy* generator; none addresses a damaged student whose rollouts collapse into repetition that the self-teacher then endorses.

3 Method

We first formalize OPD as a recovery recipe for structurally compressed LLMs. The experiments instantiate compression with Block-Influence depth pruning because it is simple, hardware-friendly, and representative of structured pruning, but the recovery objective only requires a compressed student and its pre-compression self. We then present ShortOPD, a short-to-long OPD schedule driven by repetition and truncation feedback; Figure 3 gives the overview and Algorithm 1 the procedure. Finally, we place ShortOPD and its baselines in a single design space.

3.1 Preliminaries: pruning and the OPD base recipe

Representative structured pruning operator. Let $X_{i,t}$ denote the hidden state of token t entering layer i. ShortGPT's Block Influence (BI) of layer i is

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

so a low BI_i marks a layer that barely changes the hidden state. We remove the lowest-BI layers until roughly 25% of parameters are gone, obtaining the compressed student π_θ, and keep the original model π_T as a frozen teacher. The operator is only the experimental object: ShortOPD assumes nothing beyond π_θ being a structurally compressed descendant of π_T.

On-policy self-distillation. Following the two implications of Section 1, the recovery objective should repair the student on the states it actually visits and at token granularity. OPD satisfies both: for each prompt x, the student samples on-policy rollouts $y \sim \pi_\theta(\cdot \mid x)$, the frozen teacher π_T is run on the same trajectories,

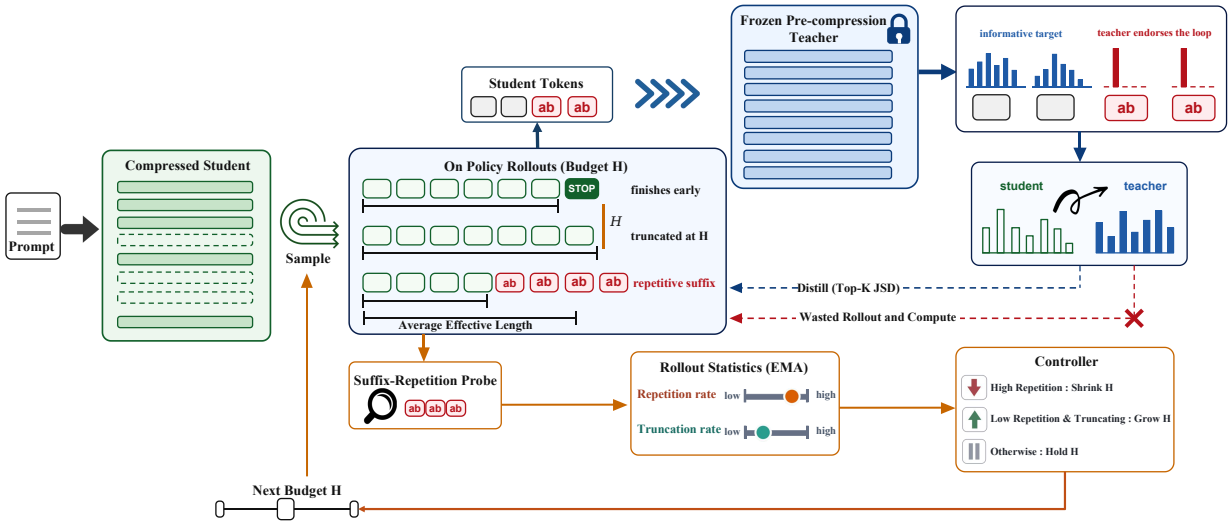


Figure 3 ShortOPD closes a second control loop around OPD. **Distillation loop (blue):** under the current budget H, the compressed student samples on-policy rollouts (finishing early, truncated at H, or collapsing into a repetitive suffix); the frozen pre-compression teacher re-scores the same tokens and returns a dense top-K distribution target at every position. **Budget loop (orange):** a token-only terminal-periodic probe reports severe repetition; its onset, locally refined by the existing OPD loss and teacher NLL, gives effective length. The fraction of non-repetitive rollouts that fill H gives the clean truncation rate. High repetition sets a shrink target from the EMA of mean effective length; low repetition and high clean truncation set a growth target. The next budget is an EMA-smoothed move toward that target.

and the student matches the teacher's next-token distribution at every response position:

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_x \, \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[\frac{1}{|y|} \sum_{t=1}^{|y|} D_\alpha\big(\pi_T(\cdot \mid x, y_{<t}),\, \pi_\theta(\cdot \mid x, y_{<t})\big) \right], \tag{2}$$

where D_α is a generalized Jensen–Shannon divergence; we distill the top-100 logits plus an aggregated tail mass, with clipped importance weighting against the rollout policy. With a large teacher–student capacity gap, forward KL is mode-covering and can force a limited student to spread probability mass across teacher modes, whereas reverse KL is mode-seeking but may reduce diversity [20, 33]. This trade-off is independent of whether the training trajectories are off- or on-policy and remains task-dependent [20]. We therefore use the balanced choice $\alpha = 0.5$, a bounded compromise that avoids extreme log-ratio penalties when the teacher and damaged student differ sharply. Equation (2) is on-policy (the expectation is over student samples), dense (every generated token receives a distributional target), and reward-free, so it applies equally to verifiable math/code prompts and open-ended instruction prompts. Because the teacher is the pre-compression self, OPD pulls the student back toward its own original behavior on the trajectories it actually produces.

3.2 Short-to-Long On-Policy Distillation

Equation (2) implicitly assumes that the student's rollouts are usable training states. Under strong compression this assumption becomes horizon-dependent: a long fixed budget reaches repetitive suffixes before the policy has recovered, while a permanently short budget truncates legitimate long generations later. ShortOPD therefore keeps the global `response_length` as a static padding and context ceiling, but sets a per-step sampling budget $H_t \le H_{\max}$. The budget is not driven by batch-mean loss, which is lowest precisely during the repetitive warm-up (Figure 2b). Instead, ShortOPD observes three statistics at every step: how many rollouts end in a severe periodic loop, how many non-repetitive rollouts exhaust the current budget, and the mean usable prefix before the detected loop. Repetition is the first-level gate: effective length may shorten the budget only when repetition is high, whereas low repetition and high clean truncation permit longer rollouts.

Algorithm 1 ShortOPD

Require: prompts $\mathcal{B}$, 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 rollouts $y_i \sim \pi_\theta(\cdot \mid x_i)$ with budget H_s
  $a_i \leftarrow \text{TerminalLoop}(y_i)$; $r_s \leftarrow B^{-1} \sum_i a_i$
  $q_s \leftarrow B^{-1} \sum_i \mathbb{1}[|y_i| = H_s \ \wedge \ a_i = 0]$
  run π_T on the same trajectories
  update θ with the dense OPD loss over all generated tokens (Eq. 2)
  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

Generation-side feedback. For rollout y_i at step s, the probe examines only its last W valid tokens. Let $a_i = 1$ when the terminal-periodic detector of Appendix B finds a severe loop and $a_i = 0$ otherwise. Let $b_i = 1$ only when the rollout fills the current budget without such a loop, i.e., $|y_i| = H_s$ and $a_i = 0$. The structural loop onset is locally refined using the OPD divergence and teacher NLL to obtain effective length ℓ_i; these loss signals refine the boundary but never determine a_i. We set $\ell_i = |y_i|$ when no severe loop is found. The batch statistics and their EMAs are

$$r_s = \frac{1}{B} \sum_{i=1}^{B} a_i, \qquad q_s = \frac{1}{B} \sum_{i=1}^{B} b_i, \qquad L_s = \frac{1}{B} \sum_{i=1}^{B} \ell_i, \qquad (\bar{r}_s, \bar{q}_s, \bar{L}_s) = \text{EMA}_\beta(r_s, q_s, L_s). \tag{3}$$

The suffix restriction avoids reacting to repeated structure earlier in a valid response, and the EMA suppresses single-batch fluctuations. These statistics reuse tokens and teacher scores already produced by OPD and require no extra forward pass. The effective-length probe is control-only: the current batch still receives the unmodified dense OPD loss of Equation (2) over all generated tokens.

Repetition-gated horizon control. The controller first forms a gated target

$$H_s^* = \begin{cases} \min(H_s, \lambda \bar{L}_s), & \bar{r}_s > \rho_{\text{high}}, \\ \gamma_\uparrow H_s, & \bar{r}_s < \rho_{\text{low}} \ \wedge \ \bar{q}_s > \tau, \\ H_s, & \text{otherwise}, \end{cases} \qquad H_{s+1} = \text{clip}_{[H_{\min}, H_{\max}]}\Big(\big\lfloor \beta_H H_s + (1 - \beta_H) H_s^* \big\rceil_{16} \Big). \tag{4}$$

Here $\lambda > 1$ retains headroom beyond the observed usable prefix, $\gamma_\uparrow > 1$ proposes growth, and β_H smooths both directions before rounding to multiples of 16. The hysteresis band $\rho_{\text{low}} < \rho_{\text{high}}$ prevents oscillation. High repetition has priority even when truncation is also high, and its severity determines the shrink target through $\bar{L}_s$. Growth requires both clean suffixes and evidence that the current horizon is binding. Consequently, short clean responses cannot reduce the budget merely by lowering average length. We initialize at $H_{\max}$ and checkpoint the current budget and all three EMAs across restarts. Algorithm 1 summarizes the loop.

3.3 A design space for recovery signals

ShortOPD is one cell of a space spanned by two axes: trajectory source (on-policy student rollouts vs. off-policy fixed data) and supervision (dense teacher distribution vs. hard labels or sparse reward). Table 2 summarizes where each recovery method sits. Our Qwen3 experiments instantiate SFT w/o KD, SeqKD, KD, and ShortOPD on the same prompt distribution, and add a GSM8K-only RLVR comparison for the sparse on-policy cell:

Table 2 Recovery methods organized by the two design axes. Only ShortOPD trains on the compressed student's own states with a dense token-level signal while steering the rollout budget away from low-information suffixes; it requires neither ground-truth responses nor a reward verifier.

Method	On-policy states	Dense signal	Label-free	Verifier-free
SFT w/o KD	×	×	×	✓
SeqKD	×	×	✓	✓
KD	×	✓	×	✓
RLVR	✓	×	×	×
ShortOPD (ours)	✓	✓	✓	✓

- **SFT w/o KD**: standard supervised finetuning on the corpus's original golden responses. This tests whether simply training the compressed model on the same data is enough.
- **SeqKD** [23]: SFT on fixed responses generated by the same pre-compression teacher π_T. This shares OPD's teacher but removes the on-policy trajectory source.
- **KD** [20, 31]: teacher-forced token-level distillation (also known as word-level KD) that keeps SFT's fixed sequences but replaces hard labels with the teacher's next-token distributions. This shares OPD's dense signal but removes the on-policy trajectory source.
- **RLVR** [21, 38]: on-policy learning with a sparse correctness reward on GSM8K; we instantiate it with both PPO and GRPO. This shares OPD's on-policy sampling but replaces dense teacher distributions with sparse answer rewards.
- **ShortOPD (ours)**: on-policy rollouts with dense self-distillation targets and the repetition-gated budget controller of Section 3.2.

4 Experiments

The experiments evaluate ShortOPD and its OPD foundation along four axes. First, we compare recovered generation quality against practical post-compression recovery baselines across two backbones and eight task families. Second, we analyze the short-to-long horizon controller: its trajectory, its comparison against fixed short and fixed long horizons, and the repetition dynamics that drive it. Third, we isolate the learning signal, separating dense on-policy distillation from off-policy SFT-style recovery and sparse on-policy reward learning. Fourth, we test how recovery depends on the prompt-domain mixture.

4.1 Setup

Backbones and compression. We use Qwen3-4B-Instruct-2507 [2], which has 36 transformer layers. The original model is the teacher π_T, and the compressed student is obtained by deleting the lowest-BI layers until roughly 25% of parameters are removed. BI is computed on PG-19 calibration documents. The teacher is frozen throughout recovery.

Recovery corpus construction. The recovery corpus contains 45,447 prompts across three domains:

- **Math**: GSM8K train [18] (7,473 prompts) and MATH train [39] (7,500), formatted as single-turn problems with a final-answer instruction; MATH training examples are kept disjoint from MATH-500.
- **Code**: 15,000 filtered NVIDIA OpenCodeInstruct [40] prompts (kept only with a positive unit-test signal, deduplicated, and with function names overlapping HumanEval or MBPP test tasks removed) plus 474 unique MBPP [41] non-test tasks.
- **Open-ended**: 15,000 instruction prompts sampled from ShareGPT/Vicuna-style conversations and UltraChat-200K [42, 43].

No HumanEval task or MBPP test task is used for recovery training. All recovery methods share this prompt distribution but consume it differently: OPD and ShortOPD use only the prompt field, sampling on-policy

Table 3 Main generation recovery results on the 25%-pruned student. Math/code are percentages; open-ended tasks are judge scores on a 1–10 scale. Avg normalizes judge scores by ×10. T% reports Avg as a percentage of the corresponding dense teacher Avg. ShortOPD uses a repetition-gated dynamic budget up to 2048 tokens.

		Math		Code		Open-ended				Overall	
Model	Method	GSM8K	MATH	HumanEval	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
	Pruned (−25%, untrained)	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	21.19	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	30.52	40.6
	ShortOPD (ours)	**62.70**	**42.00**	**43.90**	**49.81**	**4.68**	**2.04**	**7.94**	**4.28**	**48.46**	**64.5**

rollouts scored by the frozen teacher; SFT w/o KD trains on the original assistant responses; SeqKD trains on fixed responses generated by the same frozen teacher; and KD keeps SFT's fixed sequences but replaces hard labels with the teacher's next-token distributions. The comparisons therefore vary the recovery signal and trajectory source, not the prompt mixture.

Recovery methods and schedules. All recovery methods start from the same compressed student and see one epoch of the same prompt distribution. KD is a teacher-forced forward-KL baseline, $\mathrm{KL}(\pi_T(\cdot \mid x_{<t}) \| \pi_S(\cdot \mid x_{<t}))$, with temperature 1.0 and no hard-label cross-entropy term; it instantiates, on our recovery corpus, the fixed-context logit-distillation recipe used by prune-then-distill pipelines such as Minitron [7, 30], so the KD-versus-OPD contrast directly tests whether that recipe extends from recognition benchmarks to generation recovery. For OPD, the student samples on-policy responses; the pre-compression teacher is then evaluated on the same trajectories, and the student applies the top-100+tail generalized-JSD distillation loss of Section 3.1. The fixed horizons use a constant rollout budget (2048 or 8192 tokens). ShortOPD uses the same loss and data with the repetition-gated budget rule of Equation (4), initialized at H=2048 so the budget must discover the usable range from the first step. Full hyperparameters are in Appendix A. The nominal one-epoch schedule is 710 steps. All experiments run on a single node with 8× NVIDIA H20 GPUs.

Evaluation. We evaluate free-form generation only, on eight task families in the same three domains:

- **Math**: GSM8K and MATH-500, scored by answer-match accuracy. MATH-500 uses Qwen-style long decoding (a 16k generation cap) with the recommended sampling parameters.
- **Code**: HumanEval and MBPP, scored by execution PASS@1.
- **Open-ended**: Alpaca, QA, Summarization, and MT-Bench [44], judged on a 1–10 scale by GPT-5.5, an independent judge that is neither the teacher nor a Qwen model.

Tables report math/code scores as percentages and judge scores on the 1–10 scale; the Avg column normalizes judge scores by ×10 and averages the eight task families.

4.2 Main generation recovery

Table 3 reports the main results. Structured compression alone nearly destroys generation: the normalized average drops from 75.17 for the dense teacher to 5.71 after pruning. One epoch of SFT w/o KD recovers some ability (21.19 Avg), SeqKD improves on it (28.60), and KD adds a modest increase to 30.52, but all off-policy baselines remain far below on-policy distillation: ShortOPD reaches 48.46 Avg, nearly 9× the untrained score and an 18-point gain over the best off-policy baseline.

The domain-level pattern matters: on-policy recovery is not a math trick, improving code execution, Alpaca, QA, summarization, and MT-Bench simultaneously, and since ShortOPD uses no answer labels or verifiers,

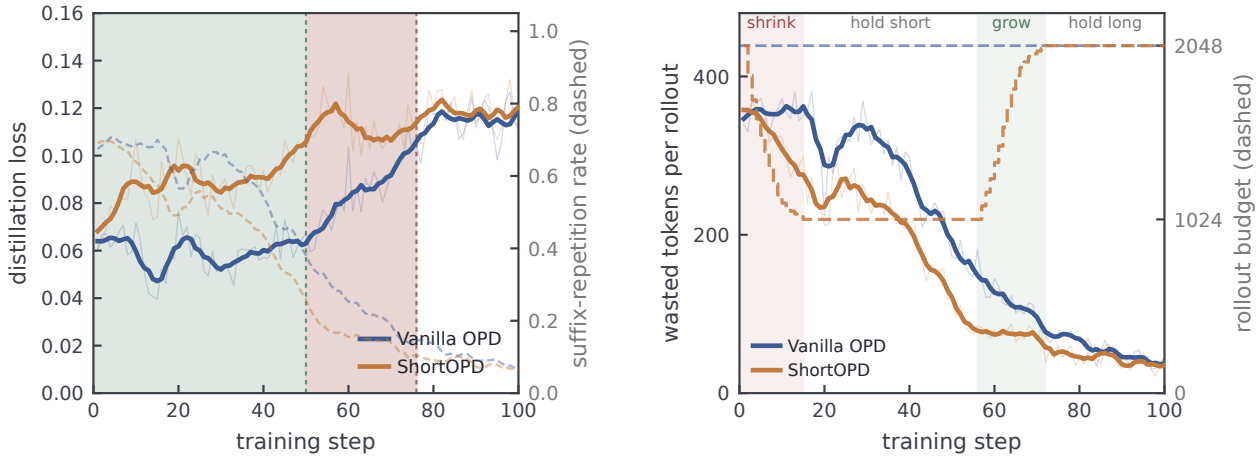


Figure 4 Revisiting the Vanilla OPD warm-up of Figure 2(b). **(a)** Distillation loss (solid, left axis) and suffix-repetition rate (dashed, right axis) for Vanilla OPD and ShortOPD. ShortOPD's loss reaches its plateau within roughly 40–50 steps versus around 80 for Vanilla: green shading marks ShortOPD's warm-up, red the additional Vanilla-only delay. **(b)** Wasted (repetitive-suffix) tokens per rollout (solid, left axis) and rollout budgets (dashed, right axis). ShortOPD shrinks its budget to 1024 during the repetitive warm-up and restores it once clean truncations dominate (shaded actions), removing the wasted mass sooner; its effective length still catches Vanilla by step 100.

the improvement comes entirely from restoring the student distribution toward its own unpruned self on the student's own states.

4.3 ShortOPD: closed-loop horizon control in practice

Closed-loop recovery trajectory. Figure 4 shows that ShortOPD shortens the repetitive warm-up. With Vanilla fixed at H=2048, 55–75% of rollouts repeat and the loss reaches its post-warm-up plateau only after roughly 80 steps. ShortOPD reaches this regime in about 40–50 steps while reducing wasted suffix tokens. Its budget falls from 2048 to 1024 during high repetition and returns to 2048 as clean truncations take over. The controller therefore saves early rollout compute without sacrificing long-generation capacity later in recovery.

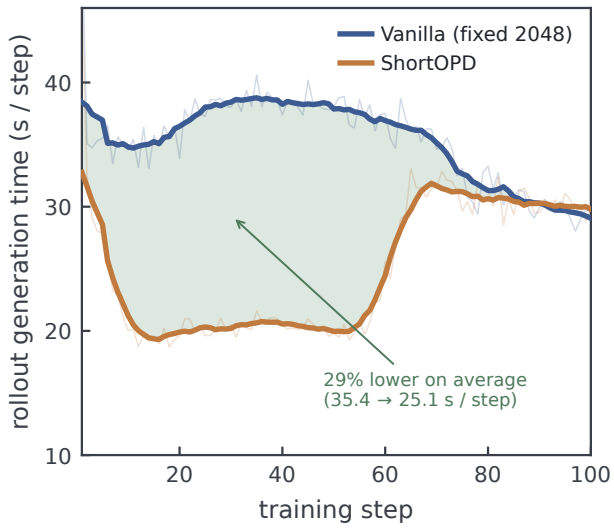
Vanilla versus gated budgets. Figure 5 compares quality and cost across rollout schedules. Relative to Vanilla fixed at 2048, ShortOPD lowers mean generation time over the first 100 steps by 29% (35.4 → 25.1s per step) and end-to-end wall-clock from 11.1 to 8.5 hours. Across fixed 2048, ShortOPD, and fixed 8192, Avg varies only from 48.5 to 50.2, whereas generated rollout tokens range from 250M to 869M. ShortOPD finishes in 8.5 hours with 250M tokens, using 76% less time than fixed 8192 and 24% less than fixed 2048 while remaining within 1.7 Avg points of both. In contrast, fixed 8192 spends 869M tokens and 35.9 hours for only +0.6 Avg over fixed 2048.

4.4 Scaling

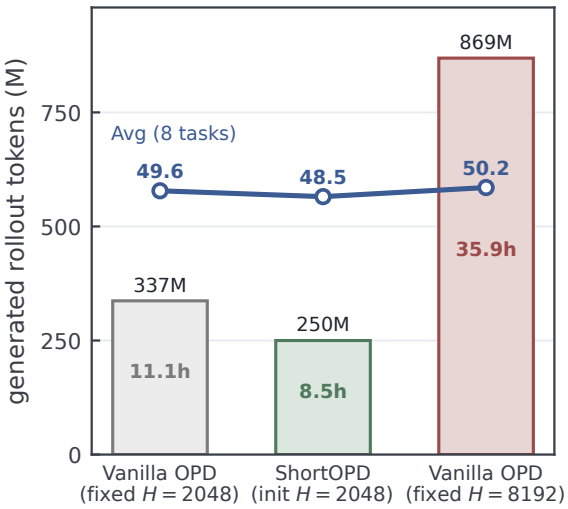
Table 4 tests whether the recovery recipe of ShortOPD continues to improve with additional exposure. Across one, two, and three epochs on the compressed Qwen3-4B-Instruct, gains are broad rather than concentrated in a single aggregate: GSM8K, MATH-500, HumanEval, MBPP, Alpaca, QA, and MT-Bench all improve from the first to the final checkpoint, with the overall Avg rising from 48.46 to 55.41 (73.7% of the teacher). On-policy recovery is therefore scalable: more rollout exposure keeps converting into generation quality, and pushing this scaling further is a natural direction for future work.

4.5 Signal controls: dense versus sparse and off-policy

The ShortOPD-versus-SeqKD and ShortOPD-versus-KD contrasts in Table 3 test whether teacher information is sufficient without on-policy states: with the same frozen teacher, moving its signal off-policy costs 19.9 Avg points (SeqKD), and dense teacher-forced distillation does not close the gap (KD trails ShortOPD by 17.9 points).



(a) Per-step rollout generation time



(b) Quality vs. rollout cost across schedules

Figure 5 What uncontrolled rollout budgets cost. **(a)** Per-step rollout-generation time for Vanilla (fixed 2048) vs. ShortOPD (faint: raw; solid: moving average) over the same first 100 steps as Figure 4; the shaded gap is saved warm-up compute (35.4 → 25.1s per step on average). **(b)** Across three schedules, the eight-task Avg (blue line) varies by less than 2 points while the generated rollout tokens (bars; ShortOPD in green, Vanilla fixed 8192 in red) vary by more than 3×. Wall-clock is annotated inside each bar.

Table 4 Per-domain epoch scaling for ShortOPD on the compressed Qwen3-4B-Instruct. Math/code are percentages; open-ended tasks are judge scores on a 1–10 scale. Avg normalizes judge scores by ×10, and T% is relative to the Instruct teacher Avg.

	Math		Code		Open-ended				Overall	
Checkpoint	GSM8K	MATH	HumanEval	MBPP	Alpaca	QA	Sum	MT-B	Avg	T%
1 epoch / step 710	62.70	42.00	43.90	49.81	4.68	2.04	7.94	4.28	48.46	64.5
2 epochs / step 1420	70.66	50.60	51.83	56.42	4.98	2.20	**8.16**	4.47	53.45	71.1
3 epochs / step 2127	**72.71**	**55.20**	**54.27**	**57.20**	**5.14**	**2.21**	8.15	**4.89**	**55.41**	**73.7**

To isolate the supervision axis, we also run a GSM8K-only comparison between sparse-reward RLVR and ShortOPD from the same compressed Qwen3-4B-Instruct initialization, instantiating RLVR with both PPO and GRPO. All methods use the same GSM8K train prompts, the same batch size, and the same one-epoch step budget; the only intended difference is the learning signal. Figure 6 shows the result: sparse on-policy RLVR barely moves GSM8K accuracy (0.23 with PPO, 1.59 with GRPO), while ShortOPD on the same GSM8K-only prompt set reaches 37.76; the multi-domain ShortOPD run reaches 62.70 and serves as the main-reference point. When the compressed model rarely samples a correct answer, a sparse reward has almost nothing to promote, whereas the dense teacher distribution corrects every token of every rollout regardless of correctness.

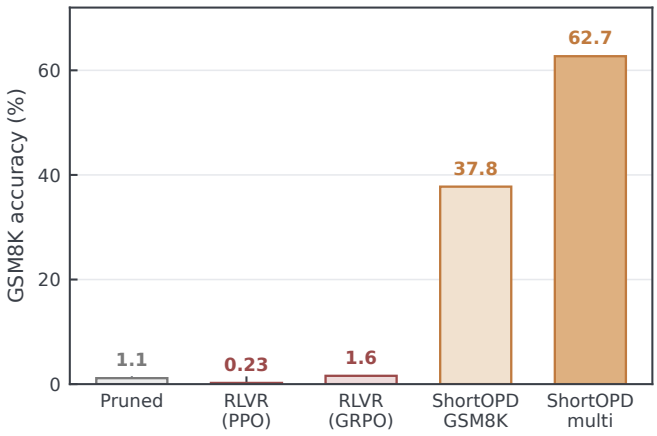


Figure 6 Sparse versus dense supervision on the matched GSM8K-only on-policy comparison. Sparse-reward RLVR (PPO and GRPO) barely moves the compressed student, while ShortOPD on the same prompts restores a large fraction of accuracy; the multi-domain ShortOPD run is shown for reference.

4.6 Recovery-corpus domain ablation

Because ShortOPD is reward-free, the recovery corpus is only a prompt distribution; we test how much its composition matters. Table 5 recovers the same compressed student with leave-one-domain-out recovery corpora. To keep the comparison matched, each ablated corpus keeps the same number of prompts and the same 710 optimizer steps as the full mixture by uniformly resampling from the two remaining domains; only the removed domain changes. This isolates prompt composition from training exposure.

Table 5 Leave-one-domain-out recovery-corpus ablation for ShortOPD on the compressed Qwen3-4B-Instruct. Math-/code are percentages; MATH uses the official MATH-500 protocol. Open-ended tasks are judge scores on a 1–10 scale; Avg normalizes them by ×10. Red marks the domain most directly damaged by the removed slice; the shaded row is the full-mixture reference.

	Math		Code		Open-ended				Overall
Recovery corpus	GSM8K	MATH	HumanEval	MBPP	Alpaca	QA	Sum	MT-B	Avg
Full mixture	62.70	42.00	43.90	49.81	4.68	2.04	7.94	4.28	48.46
w/o code	58.45	33.40	0.61	2.33	4.47	2.15	7.88	3.99	34.96
w/o math	8.04	6.00	28.66	43.19	3.96	1.73	7.19	2.94	30.51
w/o general	54.21	28.80	48.17	50.19	3.01	1.51	5.41	3.15	39.02

The resulting pattern is direct. Removing math nearly eliminates mathematical recovery, dropping GSM8K from 62.70 to 8.04 and official MATH-500 from 42.00 to 6.00. Removing code also lowers math, but collapses execution benchmarks (HumanEval 0.61, MBPP 2.33), showing that code rollouts provide a highly domain-specific repair signal. Removing general instruction data reduces both math and open-ended quality in this run, despite improving the narrow code scores. Thus ShortOPD's dense token-level signal can transfer across related states, but the on-policy state distribution still needs coverage of the capabilities we want to recover.

5 Conclusion

Structured pruning can leave latent capability in the model's search space while failing to promote it during generation, which makes recovery a distributional repair problem rather than ordinary supervised relearning. On-policy self-distillation from the model's own pre-compression self is the right repair signal: on Qwen3-4B-Instruct it substantially outperforms SFT w/o KD, SeqKD, KD, and sparse-reward RLVR on the same recovery prompts. The remaining cost is early rollout quality: long horizons spend many steps on repetitive, low-information suffixes before normal loss descent begins. ShortOPD addresses this with a repetition-gated,

truncation-aware short-to-long budget, matching fixed short and fixed long horizons within two points while generating a fraction of their rollout tokens and concentrating early teacher compute on higher-signal prefixes. These results motivate short-to-long on-policy self-distillation as a default recovery step after structured LLM compression.

6 Limitations

This work instantiates structured compression with BI depth pruning at about 25% removed parameters on the Qwen3-4B families; more pruning methods, model families, compression ratios, and larger models should be tested before claiming universality. Finally, we do not yet establish the boundary where a model has been compressed too aggressively for light post-compression recovery.

References

[1] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024.

[2] Qwen Team. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

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

[9] 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.

[10] 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 llm compression and acceleration. In Proceedings of Machine Learning and Systems (MLSys), 2024.

[11] 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.

[12] 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.

[13] 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.

[14] 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), 2021.

[15] 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.

[16] 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.

[17] 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.

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

[19] 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.

[20] 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.

[21] 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.

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

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

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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, pages 329–339, 2025.

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

[30] 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.

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

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

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

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

[39] 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, 2021.

[40] NVIDIA. Opencodeinstruct. https://huggingface.co/datasets/nvidia/OpenCodeInstruct, 2024.

[41] 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.

[42] 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.

[43] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, 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.

[44] 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.

Appendix

A Training details

Compression. Qwen3-4B-Instruct-2507 has 36 transformer blocks. BI is calibrated on 100 PG-19 documents, and the nine lowest-BI blocks are removed (Table 6); indices are reported before the retained blocks are renumbered at export. The frozen original model serves as the teacher and the compressed student initializes the actor.

Table 6 Pruned configuration.

Original depth	Removed block IDs (zero-based)	Student depth	Param. reduction
36	25–33 (layers 26–34, one-based)	27	∼25%

Main on-policy recovery runs. Training uses the 45,447-prompt math/code/open-ended corpus of Section 4.1 with rollout group 8, train batch 64, maximum prompt and response lengths of 4096 tokens each, rollout temperature 0.8, and learning rate 2×10^{-6}. The objective is the top-100+tail generalized JSD with $\alpha = 0.5$ and importance weights clipped at 2.0; no policy-gradient loss is used. One epoch is 710 steps; the main-table checkpoint is step 709.

Baselines and export. The offline baselines (SFT w/o KD, SeqKD, KD) train on the same prompt distribution with the method-specific fixed responses of Section 4.1, using sequence length 8192, batch 64, learning rate 2×10^{-6}, and one epoch (710 steps); KD uses forward KL at temperature 1.0 with no hard-label cross-entropy term. The GSM8K-only comparison trains the RLVR baselines (PPO and GRPO) and ShortOPD on its 7,473 prompts for one epoch (117 steps) with 8 rollouts per prompt and batch 64; PPO uses GAE with a critic initialized from the compressed student (learning rate 10^{-5}), while GRPO uses group-relative advantages. All runs checkpoint every 50 steps and at the final step, and checkpoints are exported in HF format without optimizer states.

Efficiency runs and controller. The ShortOPD and fixed-horizon efficiency runs reuse the corpus, objective, batch size, rollout count, and learning rate above; they differ only in rollout temperature (0.7) and response ceiling (2048 for ShortOPD and fixed H=2048; 8192 for fixed H=8192). These two $H \leq 2048$ runs also provide the training trajectories shown in Figures 2(b), 4, and 5(a). The controller checkpoints its current budget and all EMA state across restarts. Table 7 lists the detector and controller hyperparameters; symbols follow Algorithm 2.

Table 7 Detector and ShortOPD controller hyperparameters.

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

Because low loss participates in defining effective length, it never opens the shrink gate: only the independent repetition-rate threshold can do so. The probe does not mask the loss; training always applies dense OPD to the tokens actually generated.

B Terminal periodic-loop detector

Algorithm 2 gives the token-only detector used by the controller. It searches only the last W=512 valid 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.

Algorithm 2 Severe terminal periodic-loop detection

Require: valid 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**
  $c_k^{(p)} \leftarrow \mathbb{1}[z_{p+k} = z_k]$ 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^*

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 returned onset is the structural effective-length candidate. We then search locally from that onset for the first 32-token window whose OPD loss and teacher NLL fall below their absolute/relative thresholds; this 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 Conditional OPD gradient at teacher--student agreement

That a smooth divergence has zero gradient at its minimum is immediate; the content of this appendix is the rate at which the implemented per-token gradient vanishes near teacher–student agreement, and that the two implementation details – top-K+tail aggregation and clipped importance weighting – preserve this stationary point. Condition on one sampled prefix $(x, y_{<t})$ and treat that discrete state as fixed during back-propagation, as in the implemented on-policy distillation update. Let p be the frozen teacher distribution and $q = \mathrm{softmax}(z)$ the student distribution. For the canonical forward-KL token loss $\ell_{\mathrm{KL}} = \mathrm{KL}(p\|q)$, the standard softmax-cross-entropy derivative gives

$$\frac{\partial \ell_{\mathrm{KL}}}{\partial z_j} = q_j - p_j, \qquad \nabla_\theta \ell_{\mathrm{KL}} = J_{z,\theta}^\top (q - p). \tag{5}$$

The conditional gradient therefore vanishes as the student distribution approaches the teacher distribution.

Our implementation uses generalized JSD rather than forward KL. For $m = \alpha p + (1-\alpha) q$, the generalized Jensen–Shannon divergence is

$$D_\alpha(p, q) = \alpha \sum_i p_i \log \frac{p_i}{m_i} + (1-\alpha) \sum_i q_i \log \frac{q_i}{m_i}. \tag{6}$$

Because $\partial m_i / \partial q_i = 1 - \alpha$, differentiating both KL terms makes their non-logarithmic terms cancel:

$$\begin{aligned} \frac{\partial D_\alpha}{\partial q_i} &= -\frac{\alpha(1-\alpha) p_i}{m_i} + (1-\alpha)\left(\log \frac{q_i}{m_i} + 1 - \frac{(1-\alpha) q_i}{m_i} \right) \\ &= (1-\alpha) \log \frac{q_i}{m_i}. \end{aligned} \tag{7}$$

The softmax Jacobian is $\partial q_i / \partial z_j = q_i(\mathbb{1}[i=j] - q_j)$. Applying the chain rule therefore gives

$$\begin{aligned} \frac{\partial D_\alpha}{\partial z_j} &= (1-\alpha) \sum_i \log \frac{q_i}{m_i} q_i (\mathbb{1}[i=j] - q_j) \\ &= (1-\alpha) q_j \left[\log \frac{q_j}{m_j} - \sum_i q_i \log \frac{q_i}{m_i} \right]. \end{aligned} \tag{8}$$

At $q = p$, we also have $m = p$, so every logarithm in Equation (8) is zero and the conditional logit gradient is exactly zero. More locally, writing $q = p + \delta$ with $\sum_i \delta_i = 0$ and assuming $p_i > 0$ on the retained support,

$$\log \frac{q_i}{m_i} = \alpha \frac{\delta_i}{p_i} + O(\delta_i^2 / p_i^2), \qquad \frac{\partial D_\alpha}{\partial z_j} = \alpha(1-\alpha)\delta_j + O(\|\delta\|^2 / \min_i p_i), \tag{9}$$

so the logit gradient vanishes linearly as the two distributions agree (at α=0.5, with slope 1/4 of the forward-KL gradient $q_j - p_j = \delta_j$). For model parameters θ, $\nabla_\theta D_\alpha = J_{z,\theta}^\top \nabla_z D_\alpha$; hence the parameter gradient also tends to zero whenever the local logit Jacobian is bounded.

Our implementation evaluates Equation (6) after mapping the vocabulary distribution into the teacher top-100 bins plus one aggregated tail bin. This mapping is differentiable, and the analysis above applies verbatim to the binned distributions, so the gradient vanishes already when the binned distributions agree – a strictly weaker condition than full-vocabulary agreement, since differences inside the tail bin are invisible to the loss. This is the relevant condition for the loop states of Section 1: teacher and student need only agree at this coarse granularity for the update to carry no signal. The clipped importance multiplier is bounded by construction; at exact agreement, both the divergence and its gradient are zero, so multiplying by this weight

also preserves the stationary point. This is a conditional, per-token statement rather than a claim that every small scalar loss implies a small full-objective gradient. Empirically, the low JSD measures teacher–student agreement on loop states, while the separately reported low teacher NLL shows that the teacher itself assigns high probability to continuing the observed loop.

D Evaluation details

GSM8K uses the full test split (1,319 examples) and exact/normalized answer matching. MATH-500 uses boxed-answer extraction and normalization. HumanEval (164 examples) and sanitized MBPP (257 examples) are evaluated with execution PASS@1 in a subprocess sandbox. Alpaca, QA, Summarization, and MT-Bench [44] use fixed EAGLE-style prompt sets and are judged on a 1–10 scale by GPT-5.5, which is neither the teacher nor a Qwen model.

E Repetition versus pruning depth: the full sweep

Figure 2(a) shows the suffix-repetition rate for 0–12 removed layers; Figure 7 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 grows monotonically with removed depth and distinct-2 falls: 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 (Section 4.3 and Appendix A).

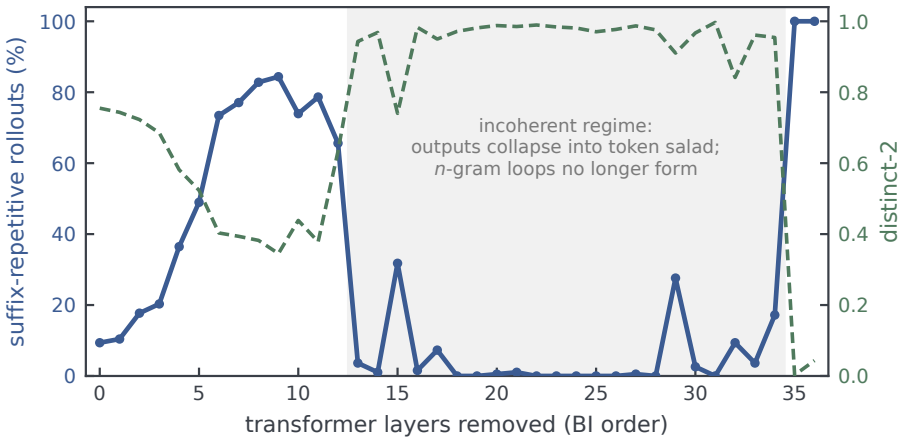


Figure 7 Full pruning sweep on the 192-prompt probe: suffix-repetition rate (left axis) and distinct-2 diversity (right axis) versus the number of BI-pruned layers. Repetition is the characteristic failure mode only in the coherent regime ($k \leq 12$); deeper pruning collapses generation into incoherent text where loops no longer form.

Qualitative collapse regimes. Table 8 shows short excerpts from the same probe. The 25% compression point is still structured enough to produce recognizable prefixes and repetitive suffixes; deeper compression often leaves the repetition metric by entering an incoherent mixed-token regime instead.

F Multiple-choice sanity check

Although the main evaluation targets free-form generation, we also evaluate held-out multiple-choice recovery on ARC-Challenge, HellaSwag, MMLU, and WinoGrande. These benchmarks are not part of the recovery corpus. Table 9 reports Qwen3-4B-Instruct results under candidate log-likelihood scoring over the answer

Table 8 Probe excerpts illustrating the transition from normal generation to repetition and then to incoherent collapse. Green indicates usable content; 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% compressed, 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 ...`

options; SFT-init ShortOPD is ShortOPD initialized from the one-epoch SFT w/o KD checkpoint instead of the raw pruned student, trained with the otherwise identical one-epoch recipe.

The pattern is the mirror image of the main results, and we read it as further evidence that recognition and generation recover along partly independent axes. Candidate log-likelihood scoring is itself an off-policy, teacher-forced task: the model must assign high likelihood to externally written text it did not generate. SFT and KD optimize exactly this objective on fixed sequences, so they recover it best (74.9 and 74.4 Avg) while remaining far behind on generation (Table 3). ShortOPD optimizes the complementary objective: it reshapes the student's distribution on its own rollouts and never trains likelihood on external text, so it dominates generation while its one-epoch MC average sits lower (67.6) – though still 22 points above the untrained pruned model and improving with exposure (71.6 at three epochs). Neither direction of the recognition-generation gap (Section 1) certifies the other. The two signals are complementary rather than conflicting: SFT-init ShortOPD composes them and recovers both axes, matching the dense teacher's MC average (76.8 vs. 76.7).

Table 9 Held-out multiple-choice recovery for the 25%-pruned Qwen3-4B-Instruct, scored by candidate log-likelihood over the answer options. Scores are accuracies in percent.

Method	ARC-C	HellaSwag	MMLU	WinoG.	Avg
Dense teacher	89.93	80.60	70.93	65.19	76.66
Pruned (−25%, untrained)	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
ShortOPD (1 epoch)	79.52	68.89	60.70	61.33	67.61
ShortOPD (3 epochs)	84.47	75.59	64.59	61.72	71.59
SFT-init ShortOPD	**89.93**	**79.67**	**69.99**	**67.64**	**76.81**