One Model to Loop Them All

Elastic Latent Recurrence for Language Models

Paper: Faisal Ladhak, Shantanu Acharya, Santiago Akle Serano,
Simeng Sun, Boris Ginsburg, Krishna C Puvvada

arXiv:2610.12582v1 8 October 2026
Independent reading discussion 15 October 2026

Preserve the useful one-loop endpoint; spend more compute when needed.

Elastic depth needs a competitive starting point

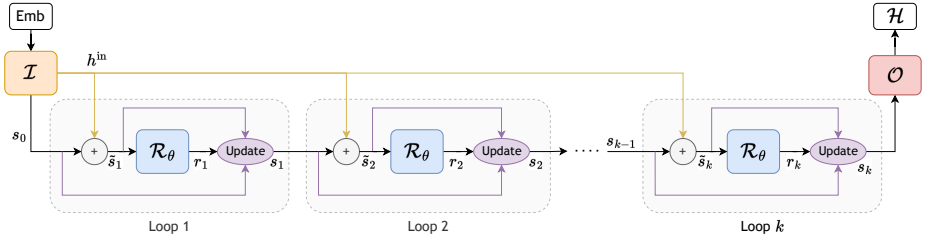
The deployment question

Can one checkpoint serve a fast tier and a stronger tier while retaining the fast tier's accuracy?

- Shared layers let the model increase effective depth without storing another set of weights.
- The one-loop tax: a looped model can underperform its non-recurrent counterpart at its cheapest setting.
- ELR combines a learned update, shallow/deep gradient mixing, and explicit loop conditioning.

Paper, Secs. 1–2, PDF pp. 1–4. Scope: loop counts within the trained range.

Only the middle block repeats



$$D_{\text{eff}}(k) = L_{\text{in}} + kL_{\text{rec}} + L_{\text{out}}$$

For the hybrid models, 14/24/14 gives **52, 76, 100, 124** executed sublayers at $k = 1, 2, 3, 4$.

Paper, Fig. 1, PDF p. 2; Eq. (1), p. 3; Appendix A.1, p. 18. Each attention, GDP, or FF sublayer counts once.

Adaptive delta learns how much to revise each state

$$\tilde{s}_i = s_{i-1} + h^{\text{in}}, \qquad r_i = \mathcal{R}_\theta(\tilde{s}_i), \qquad s_0 = h^{\text{in}}$$

A gated step toward the candidate state

$$s_i = s_{i-1} + \beta_i \odot (r_i - s_{i-1}), \qquad \beta_i = \sigma(W_\beta \tilde{s}_i + b_\beta) \odot \text{softplus}(\gamma)$$

- A token-dependent, channel-wise step; $W_\beta \in \mathbb{R}^{d \times d}$ and $b_\beta, \gamma \in \mathbb{R}^d$.
- Zero initialization gives $\beta_i = \frac{1}{2} \log 2 \approx 0.347$.
- Gate adds $d^2 + 2d$ parameters, shared across loops. Learned steps need not remain below one.

Paper, Sec. 3.1, Eq. (2), PDF p. 4; Appendix A, p. 18.

Mix shallow and deep gradients within one update

Draw a common $k \sim U\{1, \ldots, K\}$ for each microbatch. A rotating fraction f of data-parallel ranks uses one loop instead.

$$p_1 = \frac{1}{K} + \left(1 - \frac{1}{K}\right) f, \qquad \bar{k} = f + (1 - f)\frac{K+1}{2}$$

Main runs: $K = 4$, $f = 0.10$

$(p_1, p_2, p_3, p_4) = (0.325, 0.225, 0.225, 0.225)$; $\bar{k} = 2.35$.

- Example: when $k = 4$, 10 of 100 ranks run one loop and 90 run four.
- Average their gradients before the optimizer update; backpropagate through every executed loop.

Paper, Sec. 3.2, Eq. (3), PDF pp. 4–5; Appendix A.3, p. 19. Rank example is explanatory.

Condition shared layers on progress and budget

$$\tau_i = i/k, \quad \kappa = k/K, \quad c_i = \mathrm{MLP}([\phi(\tau_i); \phi(\kappa)])$$
$$\tilde{x} = (1 + \lambda(c_i)) \odot \mathrm{RMSNorm}(x) + \mu(c_i)$$
$$\hat{x} = \tilde{x} + A\big(g(c_i) \odot B\tilde{x}\big)$$
$$y = x + (1 + \omega(c_i)) \odot f(\hat{x})$$

- Modulate normalized input, add a gated low-rank correction, and scale the residual branch.
- $A \in \mathbb{R}^{d \times r}$, $B \in \mathbb{R}^{r \times d}$; parameters are layer-specific and shared across loops.
- Input block is unconditioned; output block uses $(1, k/K)$.

Paper, Sec. 3.3, Eq. (4), PDF p. 5. ϕ: sinusoidal embedding. f: one model sublayer.

Zero initialization preserves the trained endpoint

At the switch to mixed depth, add zero-initialized conditioning heads:

$$s_i = s_{i-1} + (\pi_i + g_{\text{exit}}(c_i)) \odot \beta_i \odot (r_i - s_{i-1})$$

$$\pi_1 = 1, \qquad \pi_{i>1} = 0.$$

1. Layer modulation initially makes no change to the original sublayers.
2. Loop one reproduces the existing ELR checkpoint's update.
3. Later loops have zero exit factors: $s_i = s_{i-1}$.

Initial function is the same for every budget

In exact arithmetic, every k starts from the one-loop checkpoint's output. Later loops learn useful corrections during continuation.

Paper, Sec. 3.3, PDF p. 5. The starting checkpoint already has the adaptive-delta gate.

Late mixed-depth training limits added compute

If mixed depth covers fraction ρ of training tokens,

$$\frac{C_{\mathrm{ELR}}}{C_{\mathrm{base}}} \approx (1-\rho) + \rho \frac{D_{\mathrm{eff}}(\bar{k})}{D_{\mathrm{eff}}(1)}.$$

Configuration	$\bar{k}$	ρ	Estimated cost
4B / 30B matched runs	2.35	0.25	1.16×
GPT-0.5B late ablation	2.70	0.25	1.20×
Koi-4B pretraining	2.35	0.50	1.31×

Keep optimizer state and the learning-rate schedule. These are layer-execution estimates, not measured training wall time.

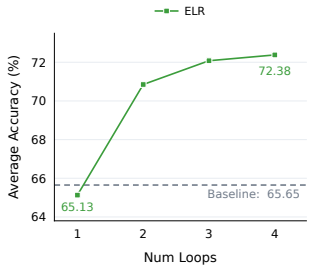
Paper, Eq. (5), PDF p. 6; Sec. 6.5, p. 13; Appendix A.4, pp. 19–20.

Three experiments answer different questions

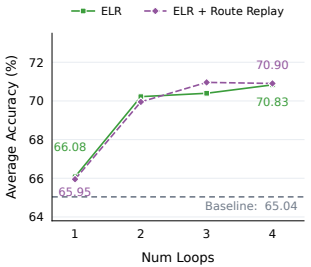
Experiment	Controlled comparison	What it establishes
4B / 30B-A3B	Matched backbone, data and optimizer; 1T / 300B tokens; no conditioning	One-loop parity and gains from depth; about 16% more estimated training compute
GPT-0.5B	300B tokens; validation loss; usually $K = 5$, no input re-injection	Effects of updates, sampling, delayed training and conditioning
Koi-4B	Full recipe plus post-training; external reference models	End-to-end usefulness; no matched non-recurrent training control

Paper, Secs. 4–6, PDF pp. 6–14; Appendix A, pp. 18–20.

One-loop averages stay close; more loops add accuracy



(a) GDP-Hybrid-4B



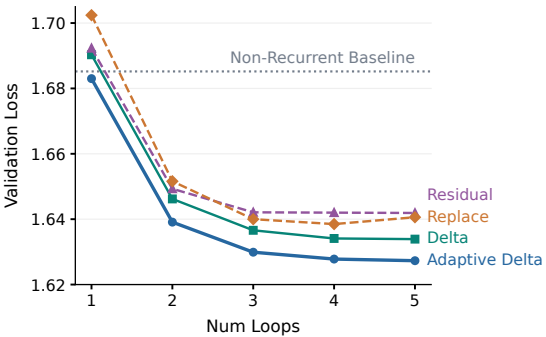
(b) GDP-Hybrid-30B-A3B

4B: 65.65 baseline, 65.13 $\rightarrow$ 72.38 ELR. 30B: 65.04 baseline, 66.08 $\rightarrow$ 70.83 ELR.

The second loop supplies 79% / 87% of the one-to-four-loop gain. Individual tasks can regress: 4B HumanEval is 66.5 at $k = 2$, 64.0 at $k = 4$.

Paper, Fig. 2, PDF p. 7; Tables 8–9, pp. 21–22. Ten-task averages; changes are percentage points.

Input-dependent step sizes give the best ablation curve



- Replace: $s_i = r_i$.
- Residual: $s_i = s_{i-1} + r_i$.
- Delta: a learned, input-independent channel-wise step.
- Adaptive delta is best at every tested depth; improves loss by 0.006–0.007 over delta.

Paper, Fig. 4a / Sec. 6.1, PDF p. 11. GPT-0.5B, 300B tokens, no input re-injection; lower loss is better.

Depth marginals alone do not determine the result

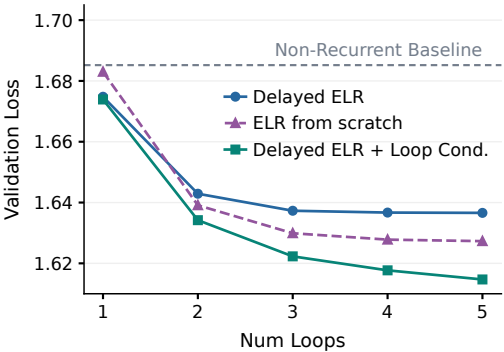
Same marginal distribution: $p_1 = 0.32$, $p_{2,3,4,5} = 0.17$.

Sampling	Depths/update	Step time	Loss at 1	Loss at 5
Synchronized, weighted	1	1.00×	1.6871	1.6316
15% rank override	≤ 2	1.00×	1.6830	1.6273
Independent ranks	≤ 5	1.36×	1.6774	1.6194

- Non-recurrent baseline loss: **1.6852**.
- Mixing more depths in an update improves the observed loss.
- Independent ranks wait for the deepest rank; overrides retain the synchronized step time.

Paper, Table 3 / Sec. 6.3, PDF p. 12. GPT-0.5B; lower loss is better.

Conditioning strengthens a late-trained loop range



- Add conditioning after 75% of tokens.
- One-loop loss: 1.6748 $\rightarrow$ 1.6739.
- Five-loop loss falls by about 0.022.
- Five-loop result: 1.6147 vs. 1.6273 for unconditioned training from scratch.

Conditioning adds 31.6M parameters (+6.34%). The comparison changes both the schedule and the model.

Paper, Fig. 6 / Sec. 6.6, PDF pp. 13–14. No conditioned-from-scratch control is reported.

Route replay mainly accelerates training

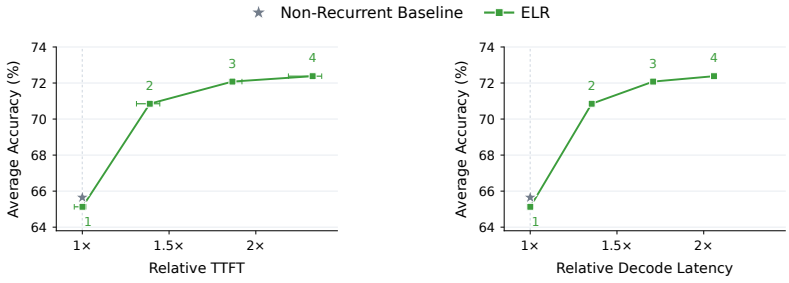
Reuse first-loop expert assignments *and weights*; experts still process each loop's updated representation.

Relative to dynamic routing	Inference-only replay	Replay training
Four-loop validation loss	+0.0055	+0.0038
Mixed-depth update time	—	−11.7%
Fixed four-loop update time	—	−19.6%
8K inference forward time	−1.2%	−1.2%

Router calls fall from 53 to 23 at $k = 4$. Expert work and all-to-all communication remain. Four-loop average score: 70.83 dynamic vs. 70.90 replay.

Paper, Table 1 / Sec. 4.3, PDF p. 8; Appendix E, pp. 25–26. The 11.7% saving applies to the mixed-depth phase.

Extra depth has a measurable latency price



(a) Time to First Token

(b) Decode Latency

At $k = 4$: 2.33× TTFT and 2.06× per-token decode latency.

At $k = 3$: 96% of the 4B accuracy gain at 1.87× TTFT.

Paper, Fig. 7, PDF p. 25. H100 80GB, BF16, batch 1, 1,024 input + 128 fixed output tokens; vLLM 0.20.2, eager prefill, CUDA-graph decode. Bars measure timing uncertainty.

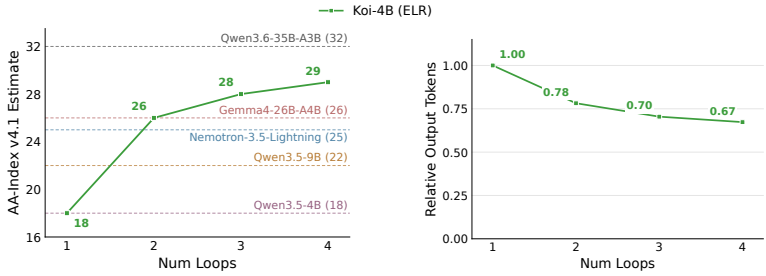
Koi carries mixed depth through post-training

Stage	Budget	Loop schedule
Pretraining	7T tokens at 8K context	First half $k = 1$; second half mixed + conditioning
Context extension	50B tokens at 256K	Uniform $k = 1 \ldots 4$, 10% override
SFT	Text-only data, four epochs	Same mixed-depth schedule
RLVR	200 async GRPO steps	$4 : 1 : 1 : 1$ prompt allocation across depths

Conditioning adds 155.6M parameters (3.94% of the backbone). Koi has no matched non-recurrent training control.

Paper, Sec. 5.1, PDF p. 9; Appendix A.4 / Table 6, pp. 19–20.

Koi improves scores while producing fewer tokens



(a) AA-Index v4.1 Estimate

(b) Relative Output Tokens

Unrounded index: **17.52, 25.57, 27.93, 29.23**.

Authors' harness estimates, not official AA results. CritPt is unrun and contributes zero; reference models are evaluated in the same harness.

Paper, Fig. 3, PDF p. 9; Tables 10–11, pp. 22–23. Token ratios are geometric means across eight benchmarks.

Shorter outputs partly offset per-token compute

$$\bar{r}_k = \left(\prod_{b=1}^{8} \frac{T_b(k)}{T_b(1)} \right)^{1/8}$$

$T_b(k)$ is mean completion tokens per rollout, summed over all requests and turns, including reasoning and capped requests.

Approximate decode-cost arithmetic

$$k = 2: \quad 1.36 \times 0.7821 \approx 1.06 \qquad k = 4: \quad 2.06 \times 0.6732 \approx 1.39$$

This combines fixed-workload GDP-Hybrid-4B token latency with Koi's benchmark length ratios. It is an estimate of decode work, not measured end-to-end task latency.

Paper, Sec. 5.2, PDF p. 10; Eq. (6) / Appendix C.1, p. 23; Appendix D, pp. 24–25.

One KV cache changes which history each loop reads

During decoding, token t at loop i attends to

Earlier tokens' **final-loop** keys and values, plus token t's own loop-i keys and values. The current slot is overwritten each loop.

Three of six attention layers are recurrent:

$$\frac{M_{\text{per-loop}}(k)}{M_1} = \frac{3+3k}{6}, \qquad \frac{M_{\text{last-loop}}(k)}{M_1} = 1.$$

- At $k = 4$, 128K context, BF16: **7.5 GiB** $\rightarrow$ **3.0 GiB** attention KV per request.
- GDP keeps per-loop state, below 25 MiB per request at $k = 4$.
- No retraining; the cache policy changes the attention computation.

Paper, Secs. 3.4 / 5.3, PDF pp. 6, 10–11. Memory ratio derived from the reported layer layout.

Most depth gains survive KV sharing, with losses

Benchmark	One loop	Four: per	Four: last	Δ pp
Terminal-Bench 2.1	12.1	28.0	27.4	−0.6
Humanity's Last Exam	5.9	11.9	10.5	−1.4
AA-LCR	51.3	68.3	65.3	−3.0
LiveCodeBench v6	50.5	71.9	70.2	−1.6
MATH-500	88.8	97.2	97.0	−0.2
AA-Omniscience non-halluc.	86.1	85.9	88.2	+2.3

Differences use unrounded scores. Across depths, seven of 27 paired comparisons reach uncorrected $|z| \geq 2$; uncertainty is within a single run.

Paper, Table 2, PDF p. 10; Appendix C.2, pp. 23–24. Scores are percentages; deltas are percentage points.

GDPval exposes a consequential cache-policy failure

Failures among 220 tasks	$k = 2$	$k = 3$	$k = 4$
Per-loop KV	1	6	12
Last-loop KV	13	29	52

- Most additional failures are timeouts; agent turns rise 17–19%.
- Timed-out tasks receive no Elo judgment, so the scored task sets differ.
- The paper reports no AA-Index composite for last-loop KV.

Practical implication

Memory savings need task-completion and wall-time evaluation alongside benchmark scores.

Paper, Appendix C.2, PDF p. 24.

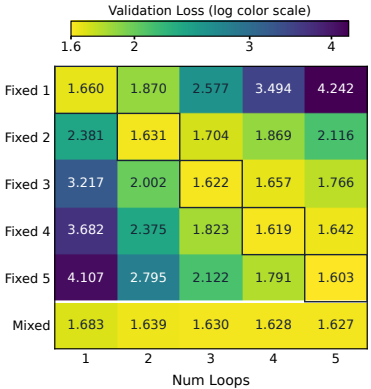
ELR makes a useful compute knob within one model

1. **Protect the endpoint.** Adaptive updates and within-update depth mixing preserve near-baseline one-loop averages.
2. **Learn extra depth incrementally.** Zero-initialized conditioning enables a function-preserving transition; the late recipe improves deeper loss.
3. **Account for each cost separately.** More loops increase compute; route replay saves training work; KV sharing saves memory with task-dependent costs.

Open directions: later-loop training exposure, conditioned-from-scratch controls, adaptive stopping, and post-trained recurrent MoE.

Paper, Sec. 8, PDF p. 15, and limitations from Secs. 4–6 / Appendix C.2. Main discussion ends here; backup slides follow.

Backup: fixed depth does not yield elastic inference



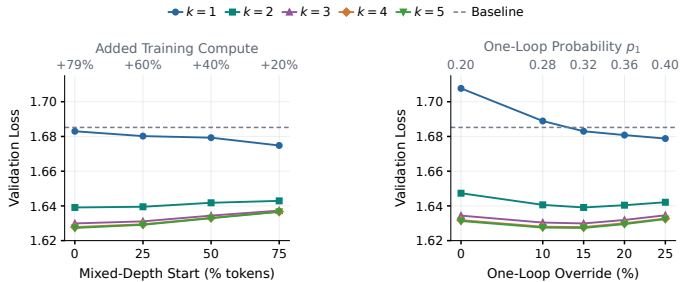
Rows: separate training runs.
Columns: evaluation depth.

Each fixed-depth model works best at its own training depth.

Mixed depth gives a usable curve in one checkpoint, with some loss relative to each specialist.

Paper, Fig. 4b / Sec. 6.2, PDF pp. 11–12. GPT-0.5B validation loss; logarithmic color scale.

Backup: training schedules trade quality for cost



(a) Delayed Mixed-Depth Training

(b) One-Loop Override

Each x-axis point is a separately trained model. Late training reduces overhead; larger override fractions favor the shallow endpoint over deeper loops.

Paper, Fig. 5 / Secs. 6.4–6.5, PDF p. 13. GPT-0.5B; estimated compute is specific to $K = 5$, partition 8/14/8.

Backup: aggregate route stability hides token drift

First versus fourth loop, dynamic model	Value
Aggregate expert-frequency cosine	0.978
First-loop experts retained per token	53.17%
Routed-output cosine: replay / shifted control	0.767 / 0.227
Full-MoE cosine: replay / shifted control	0.918 / 0.718

256 held-out 2K-token windows, averaged over ten recurrent MoE layers. Output comparisons hold the fourth-loop input fixed; shifted routes preserve usage counts but break token correspondence.

Paper, Appendix E / Table 13, PDF pp. 25–26. Retention 95% document-bootstrap interval: [52.37, 53.89]%.

Backup: entry and exit conditioning complete the rule

$$z_i = \begin{cases} \tilde{s}_i, & i = 1, \\ (1 + \lambda_e(c_i)) \odot \mathrm{RMSNorm}(\tilde{s}_i) + \mu_e(c_i) + \eta(c_i) \odot h^{\mathrm{in}}, & i \geq 2. \end{cases}$$

$$\begin{aligned} r_i &= \mathcal{R}_\theta(z_i; c_i), \\ \beta_i &= \sigma(W_\beta \tilde{s}_i + b_\beta) \odot \mathrm{softplus}(\gamma), \\ s_i &= s_{i-1} + (\pi_i + g_{\mathrm{exit}}(c_i)) \odot \beta_i \odot (r_i - s_{i-1}). \end{aligned}$$

- First loop receives $z_1 = 2h^{\mathrm{in}}$ in the hybrid models.
- The adaptive gate uses the unmodulated $\tilde{s}_i$.
- Zero heads make later exit factors vanish; $\pi_1 = 1$, $\pi_{i>1} = 0$.

Paper, Sec. 3.3, PDF p. 5. GPT-0.5B default ablations omit input re-injection.

Backup: input re-injection is not essential at 0.5B

Recurrent input	Loss at $k = 1$	Loss at $k = 5$
Non-recurrent baseline	1.6852	—
s_{i-1}	1.6830	1.6273
$s_{i-1} + h^{\text{in}}$	1.6858	1.6325

- Re-injection slightly worsens this configuration's loss.
- The paper reports unchanged ordering after repeating sampling, fixed/mixed, override and delayed-training ablations.
- Transfer of the state-update ranking is expected by the authors, not demonstrated in this appendix.

Paper, Appendix F.1 / Table 14, PDF p. 26.

Reference and reading materials

Faisal Ladhak, Shantanu Acharya, Santiago Akle Serano, Simeng Sun, Boris Ginsburg, and Krishna C Puvvada.

One Model to Loop Them All: Elastic Latent Recurrence for Language Models.

arXiv:2610.12582v1, 8 October 2026.
https://arxiv.org/abs/2610.12582v1

All empirical figures are from this version of the paper. Selected tables are transcribed from the cited source tables. Explanatory examples and memory/cost arithmetic are identified where used.

Companion: Chinese reading notes, editable Beamer source, and local paper PDF in the reading archive.