EARLY-STOPPING FROZEN TRANSFORMER ENCODERS VIA SPECTRAL RANK

Anonymous authors
Paper under double-blind review

ABSTRACT

Selecting a transformer backbone for use as a frozen encoder requires committing to depth, head count, width, positional encoding scheme and normalization style. Each combination is conventionally evaluated by training the backbone to convergence and probing it on downstream tasks. Recent theory, validated on synthetic data, predicts a two-phase geometric trajectory for transformers when trained on standard next-token prediction. Transformer representations first compress into low-dimensional, factored subspaces. They then re-expand to capture joint correlations and reduce loss. We investigate a downstream consequence the original theory does not address: whether gains in linear probe accuracy on held-out tasks concentrate near the compression-to-expansion transition, and how these gains depend on the probed target. We test this on two structured-sequence domains: chess and network traffic trace-derived tokens from a specification-compliant Open Radio Access Network (O-RAN) testbed generating data on a variety of telecom equipment combinations, and on natural language using published Pythia training trajectories. We train a family of small next-token language models on these domains, varying architectural choices such as depth, normalization style, RoPE layout and RoPE subspace allocation. Using intra-epoch checkpoints we track both the 95%-variance spectral rank of layer-1 representations and the test accuracy of linear probes trained on held-out tasks whose labels the language model does not see. We show that across the position-aware architectural variants tested, with the no-position control behaving as an expected exception, gains for particular probes concentrate near the point of maximal spectral compression. For these probes, continued language-model training contributes marginal, and in some configurations negative, improvement.

1 INTRODUCTION

Selecting a transformer backbone for use as a frozen encoder usually requires an expensive empirical search (Vaswani et al., 2017; Yosinski et al., 2014; Peters et al., 2018). Practitioners vary architecture, positional encoding, normalization, learning schedule, and checkpoint choice. They then evaluate each candidate by training to convergence and probing or fine-tuning on downstream tasks (Yosinski et al., 2014; Peters et al., 2018; Alain & Bengio, 2016). This is especially costly when the target domain is structured but non-linguistic. In such domains, the model may learn useful latent organization long before language-model training has converged. Standard training loss also provides no guarantee that the best next-token predictor is also the best frozen representation (Alain & Bengio, 2016; Belinkov, 2021; Hewitt & Liang, 2019).

Recent theory suggests that next-token-trained transformers pass through a compression-expansion trajectory early in training (Shai et al., 2026). This raises a downstream question not addressed by the original theory: is the compressed phase merely a geometric curiosity, or is it the point at which the representation is most useful for linear readout? We study this question in three domains: packet traces from an Open Radio Access Network (O-RAN) testbed, chess games, and natural language (published Pythia trajectories). We checkpoint densely, track spectral rank with the 95% cumulative explained-variance rank k_{95} at every layer for O-RAN and chess and sample five layers for Pythia, and probe held-out latent targets which the underlying transformer did not see in pretraining. The answer depends on the target. For some targets, natural language in particular, probe gains are

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

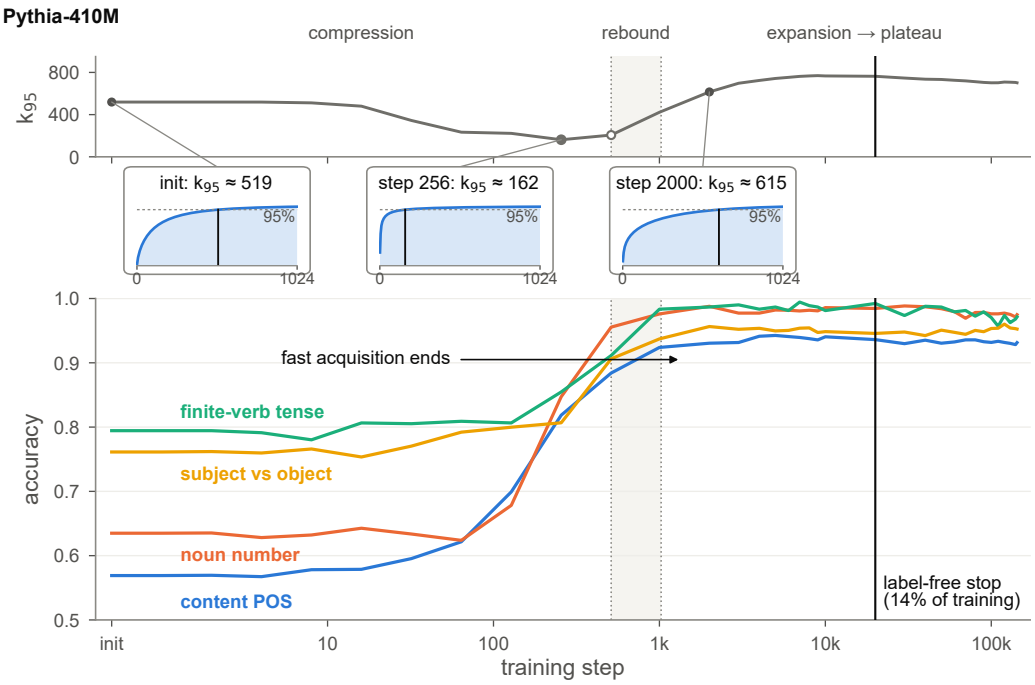


Figure 1: The compression–rebound–expansion trajectory and probe acquisition in one run (Pythia-410M, PolyPythia seed 1, final layer, 35 published checkpoints). **Top:** smoothed k_{95} of final-layer token activations on a fixed, unlabeled cohort. Insets show the cumulative explained-variance curve at initialization, at maximal compression (step 256), and during expansion (step 2,000); the vertical line marks where it crosses 95%. The shaded band spans one doubling after the rebound onset (open marker). **Bottom:** test accuracy of raw-feature linear probes for four syntactic targets that the language model never sees. All four gains are essentially complete within two doublings of the rebound. The label-free rate-to-peak rule (Section 4.4; $q = 0.1$, the setting used throughout) stops at step 20,000, 14% of training, where the four probes are within $+0.5$, $+1.0$, $+2.1$, and -0.7 points of their accuracy at step 143,000. Classic early stopping on held-out loss trains this model to the end.

essentially complete within one to two doublings of the spectral rebound (Figure 1). For others, the main chess concepts in particular, they continue long after it.

Our contributions are:

1. **Probe acquisition is target dependent, and the spectrum marks where fast acquisition ends.** Across O-RAN, chess, and Pythia, targets whose gains are fast finish a median of one to two doublings after the k_{95} rebound (about six for the main chess concepts). For O-RAN fault labels and Pythia syntactic labels, that covers essentially all of the gain. This holds across architectures, seeds, and every measured layer.

2. **A fast probe gain need not reflect newly learned structure.** In some cases the raw probe's gain tracks the expansion of its own input spectrum. This is the case on O-RAN. Probe learning curves should therefore be read together with a scale-invariant probe and the spectrum.

3. **A label-free stopping condition.** A single setting of the rule stops Pythia at 14% of training (classic early stopping on loss: 77–100%) and O-RAN at 35%. On Pythia, it comes within 0.2 points of final probe accuracy on average. The same setting declines to stop 10 of 15 chess runs, which are still learning.

4. **A lower bound for comparing candidates.** Architecture rankings agree with their final order from when acquisition is complete, which in O-RAN is at or shortly after the rebound. This gives a label-free minimum training length before candidate encoders can be compared.

2 BACKGROUND

Factored representations and compression–expansion dynamics. Our work builds on the theoretical framework and empirical findings of Shai et al. (2026). In experiments on synthetic generalized Hidden Markov Models, they find that transformers trained on next-token prediction preferentially learn low-dimensional representations of latent factors in approximately orthogonal subspaces. Such representations preserve predictive information when the factors are conditionally independent, but become lossy when this condition fails. In the latter setting, models nevertheless favor factored representations early in training and, when predictive demands warrant it, subsequently expand into higher dimensions to recover fidelity. Shai et al. interpret these dynamics as evidence for an inductive bias toward factorization. Motivated by their findings, we investigate compression–expansion dynamics through changes in activation dimensionality.

Relation to Information Bottleneck, neural collapse, and grokking. Here, compression and expansion refer to decreases and increases in activation effective dimensionality during training. This geometric notion of compression differs from the mutual-information notion used in the Information Bottleneck framework (Tishby et al., 2000). Our focus also differs from neural collapse (Papyan et al., 2020), which concerns the geometry of class representations and classifiers near the end of supervised training, and grokking (Power et al., 2022; Nanda et al., 2023), which concerns delayed generalization after training performance has saturated. In our experiments, the compression–expansion transition occurs early, within a single training epoch, under a next-token prediction objective without class labels.

Linear probing as a representation diagnostic. Linear probes are a standard diagnostic for what information is linearly accessible from a frozen representation (Alain & Bengio, 2016; Belinkov, 2021; Hewitt & Liang, 2019). Because the readout has limited capacity, probe performance is a defensible proxy (Alain & Bengio, 2016; Hewitt & Liang, 2019) for the quality of the representation rather than for the expressivity of a downstream model. In all three domains, probe targets are latent variables (injected faults, board state, syntax) that the language model did not see in training. A finite probe also measures how *accessible* that information is to the readout it trains: a raw-feature linear probe with a fixed optimization budget can fail on information that a probe on standardized features reads immediately. We therefore use two probes that answer different questions. A probe on standardized features asks whether a target is linearly present at all. A probe on raw features asks whether it is accessible in the representation's native geometry: the geometry seen by any downstream use that consumes frozen features as they are, such as similarity search, clustering, nearest-neighbour or anomaly detection and also the geometry that spectral rank measures. For a reusable frozen domain modeling backbone the second question is usually the operative one, we therefore report both sets of results and treat them as complementary (Section 5.4).

Our criterion is related to, but distinct from, work on compute-efficient training. Data selection, curricula, active learning, and dataset distillation reduce cost by valuing individual examples (Shrivastava et al., 2016; Bengio et al., 2009; Settles, 2009; Sener & Savarese, 2018; Wang et al., 2018). Our signal needs no per-example valuation as any representative unlabeled subset suffices to estimate the spectrum. Validation-loss early stopping (Prechelt, 1998) is the closest analogue, but next-token loss keeps improving long after linear-probe utility has saturated. Learning-rate and schedule search (Smith, 2017; Smith & Topin, 2017; Smith et al., 2018; Sohl-Dickstein, 2024) tunes *how* to train; we ask when a single run's representation is ready, and when candidate runs can first be meaningfully compared.

Open RAN as a structured-sequence domain. O-RAN specifications (ETSI, 2026) define an interoperable cellular base station as a set of disaggregated radio, distributed, and centralized units connected by standardized interfaces (ETSI, 2026). Packet captures on these interfaces yield discrete token sequences with nontrivial structure. This structure includes layered protocol grammar, long-range dependencies between request and response messages, and state-machine-like behavior governing session establishment, handover, and teardown. The downstream tasks of interest include identifying which deployment scenario produced a trace or which equipment perturbation is present. These tasks manifest only through the joint distribution of the sequence. They are, by construction, invisible to per-message conformance testing. Because we control the testbed, we can induce specific perturbations and verify that traces remain specification-conformant. We can then ask whether those perturbations are nevertheless linearly recoverable, and how accessible they are to a plain linear

readout of the encoder's native features. This setting is motivated by practical Open RAN integration challenges observed in the NTIA 5G Challenge, where multi-vendor integration required structured test plans, conformance checks, and detailed debugging of interface-level behavior (Pinson et al., 2025). The practical motivation is diagnosis from passive traces: a self-supervised sequence model pretrained on unlabeled interface traces could serve as a domain foundation model, reused as a frozen encoder for diagnosis, monitoring, and anomaly detection tasks that are not known at pretraining time. Choosing such backbones efficiently is the problem we study. The theoretical value is a structured sequence corpus whose downstream labels correspond to latent system variables rather than surface tokens.

3 Data Generation

3.1 ORAN-Diag: a controlled O-RAN diagnostic corpus

We evaluate on ORAN-Diag. It is a controlled digital-twin benchmark that exposes packet-level and test-harness-level observables at standardized O-RAN interfaces under programmatic fault injection. The benchmark is designed to produce reproducible labeled fault episodes. These episodes span interface, timing, configuration, and resource-pressure failure modes. Because the generating intervention is known by construction, the dataset provides ground-truth scenario labels. It still requires models to infer those labels from packet-derived sequence observations.

The digital twin is built on srsRAN Project in split CU/DU mode, srsRAN 4G UE, an Open5GS 5G core, and ZeroMQ-based RF interconnection (Software Radio Systems, b;a; Open5GS; The ZeroMQ Project). Each episode is executed hermetically in a single container with no state shared across runs. The resulting corpus contains $2{,}000$ scenarios. Each scenario has exactly one design-time condition. This condition is either a nominal baseline or one injected fault. The raw intervention catalog contains the nominal class and faults involving F1-C transport, delayed F1-C availability, F1 setup profile mismatches, UE subscriber mismatch, aggressive protocol timeouts, F1-U transport, F1-U loss, and resource/load pressure through a small GTP-U queue (3GPP, 2026c;b; Stewart et al., 2022; 3GPP, 2026a; Postel, 1981). Transport faults are implemented through deterministic firewall rules. Configuration faults are implemented through edits to CU, DU, or UE configuration files. Dataset generation required approximately 40 Ryzen 9 7900X3D CPU-hours.

Each of the 2,000 scenarios is observed through a fixed suite of 21 harness tests; each test yields a tokenized packet-derived sequence (113-token vocabulary, mean length 268 tokens, 11.3M tokens in total) with parallel timestamps. The primary downstream label maps each scenario's injected condition to one of seven root fault families (nominal, F1-C transport, F1 setup/configuration, subscriber identity, protocol timeout, F1-U transport/impairment, and user-plane resource pressure) and all 21 records of a given scenario share its label. Backbones are trained only with the causal next-token objective over tokens and their timestamps. The scenario distribution, test suite, packet serialization, tokenization, and training objective are detailed in Appendix A.

3.2 Additional domains: chess and natural language

Chess. We train next-move-token language models on Lichess blitz games. The training stream consists of 400,000 games that are disjoint from a fixed evaluation cohort. The models are trained for 12,500 updates at batch size 32, so each game is seen once, with no repeated data. Probe targets are 15 game-state labels read at fixed move boundaries: game outcome, side in check, any check, capture, capture-count parity, average Elo band, engine-evaluation band, material advantage, castling, bishop pair, doubled pawns, a centre-visit XOR, game-length extremes, e4 visited, and a random-hash control. The labels come from the game record, parsed with `python-chess` (Fiekas), and from Stockfish evaluations (The Stockfish developers) at 100,000 nodes per position.

Natural language. For natural language we use the published Pythia training trajectories (Biderman et al., 2023): 160M and 410M parameters with five seeds each, including the PolyPythia seed replicates (van der Wal et al., 2025), and 1B, 1.4B, and 6.9B with one seed each. We read 35 published checkpoints from initialization to step 143,000. Probe targets are word-level syntactic labels from the English Web Treebank (Universal Dependencies): content part of speech (noun, verb, adjective, adverb), noun number, finite-verb tense, subject versus object, and a random-occurrence

control. Each word is represented by the hidden state of its last subword token. Train, development, and test splits are disjoint at the document level.

4 METHODOLOGY

4.1 BACKBONES AND CHECKPOINTS

O-RAN. We pretrain a family of small causal transformer language models (Vaswani et al., 2017) from scratch on the tokenized O-RAN trace corpus, as candidate backbones for an O-RAN domain foundation model. They are next-token predictors, not classifiers: no downstream label is used in training. Unless varied, each model is a decoder-only transformer with $L = 4$, hidden width $d = 128$, $H = 4$ attention heads, feed-forward width $4d$, context length 1024, dropout 0.1, tied embeddings, and AdamW (Loshchilov & Hutter, 2019) with learning rate 3×10^{-4}, weight decay 0.1, and batch size 64. Ten architectures (Table 5, Appendix B) vary positional information (none, learned absolute, sequence RoPE, continuous-time RoPE, and factored sequence–time RoPE (Su et al., 2021)), normalization (LayerNorm or RMSNorm), the sequence/time split of rotary dimensions (8/24, 16/16, 24/8 per head), and depth (3, 4, or 5 layers). Timestamps enter only through RoPE encodings. Each architecture is trained with five seeds for 6,580 updates. Spectra are recorded at every layer on a dense schedule of 477 checkpoints, and probes are fitted at 24 of them.

Chess and natural language. Chess models are decoder-only transformers trained from scratch with the next-token objective, in three architectures (base: $d = 256$, $L = 4$, $H = 4$; deep: $L = 8$; wide: $d = 512$, $H = 8$) with five seeds each (Table 6). For natural language, we analyze the published Pythia trajectories at their 35 released checkpoints and at layers $1, \frac{1}{4}L, \frac{1}{2}L, \frac{3}{4}L, L$.

4.2 SPECTRAL RANK AND THE LABEL-FREE CLOCK

For a checkpoint, let $Z_\ell \in \mathbb{R}^{n \times d}$ be the matrix of token hidden states collected from layer ℓ on a fixed cohort. We center it and compute the eigenvalues $\lambda_1 \geq \cdots \geq \lambda_d \geq 0$ of its empirical covariance. The 95%-variance spectral rank is

$$k_{95} = \min \Big\{ k : \textstyle\sum_{i \leq k} \lambda_i \big/ \sum_{i \leq d} \lambda_i \geq 0.95 \Big\}.$$

At every checkpoint, k_{95} is computed on the same fixed inputs, chosen without reference to any label: 256 token rows for O-RAN, 1,869 games for chess, and about 65k tokens for Pythia. We use a fractional version of k_{95}, which interpolates within the crossing eigenvalue so that small spectral changes register, and smooth it with a causal exponential moving average with a half-life of 5 updates, giving $S(t)$. The *rebound* onset τ is the first strict rise of S after a strict new minimum. A later strict new minimum resets it.

4.3 LINEAR PROBES

At each probed checkpoint we freeze the transformer and train a linear probe. For O-RAN, a scenario is represented by concatenating the final-EOS hidden states of its 21 test records in sorted test order, $\phi(s) = \mathrm{concat}(h_{s,j_1}, \ldots, h_{s,j_{21}}) \in \mathbb{R}^{21d}$, and the probe predicts the seven-way fault family, the configured variant, fault versus nominal, and a random control. Language-model training uses a fixed set of 1,000 scenarios and the probes use the 1,000 split 500/250/250 into fit/validation/test sets fixed across runs. The probe $f(\phi) = W\phi + b$ is trained with AdamW (Loshchilov & Hutter, 2019) for up to 200 epochs with learning rate 10^{-3}, weight decay 10^{-4}, minibatch size 1024, and development-loss early stopping (checks every 5 epochs, patience 50) (Prechelt, 1998). To keep the readout identical across domains, we apply this probe unchanged to every domain, layer, and task, on raw, unstandardized features (the *raw probe*). We also fit the same probe after z-scoring each feature with training-split statistics (the *standardized probe*). We report class-balanced test accuracy, validation-selected: the accuracy at training step t is the test accuracy of whichever probe, among those fitted at checkpoints up to t, has the highest validation accuracy. Checkpoints are not chosen by test score.

4.4 LABEL-FREE STOPPING RULE AND EVALUATION

We use the label-free rule *rate-to-peak* computed causally from $S(t)$. After the rebound onset τ, let $g(t) = (S(t) - S(\tau))/(t - \tau)$ be the average growth rate of the rank since onset, and $G^-(t)$ the largest g observed before t in the current phase. The rule stops at the first observation with

$$g(t) \leq q\, G^-(t) \quad \text{and} \quad t \geq 2\tau,$$

that is, once rank growth has slowed to a fraction q of its peak rate, and no earlier than one doubling after the rebound. The second condition prevents fluctuations of a fraction of a dimension at a near-degenerate minimum from registering as a rebound. Smaller q is more conservative. We use one setting, $q = 0.1$, for every domain, architecture, and layer, and report the full range of q in Appendix E. The baseline is classic early stopping on the language-model loss: we evaluate it every 1% of training (for Pythia, at its published checkpoints) and stop after three consecutive evaluations without improvement.

A layer-run passes if every real (non-control) task at the stop is within 2 percentage points of its value at the reference horizon: 12,500 updates for chess, 6,580 for O-RAN, and 143,000 for Pythia. The *oracle* stop is the earliest probe checkpoint that passes, determined with labels. We measure the time from the rebound to the point where a task comes within 2 points of its final value in training doublings ($\log_2$ of the ratio of steps). To assess when candidates can be compared, we compute the Spearman correlation between architecture rankings (seed-averaged probe accuracy) at intermediate checkpoints and at the end of training.

5 RESULTS AND DISCUSSION

5.1 COMPRESSION IS CONSISTENT ACROSS ARCHITECTURES, AND O-RAN PROBE GAINS SATURATE NEAR RECOVERY

Across the eight RoPE, depth, and normalization variants of O-RAN (40 runs), layer-1 compression is tightly concentrated: k_{95} falls from about 30 at initialization to a minimum of 12.6 ± 0.9 (median 12, range 11–15; 90% of runs in $[12, 14]$) and recovers to about 32 by the end of training. Probe accuracy rises sharply around the rebound and largely saturates by spectral recovery, in all ten architectures (Appendix C, Figure 3): every position-aware variant (except the learned-absolute control) gains at the rebound, the no-position control gains less and later, and the learned-absolute control, whose probe input is high-rank from the start, gains nothing.

5.2 WHICH PROBE GAINS CONCENTRATE NEAR THE TRANSITION DEPENDS ON THE TARGET

Table 1 summarizes, for every task, layer, and run, how much each probe gains over training and when that gain finishes. The three domains separate on two axes (Figure 2).

- **O-RAN:** fast gains that depend on feature scale. The median gain finishes at 1.5% of training, and none of it survives standardization. The standardized probe already reaches 79.5% at initialization, above the raw probe's 64.6% after training.
- **Pythia:** fast gains that are also present after standardization (92% retained).
- **Chess:** slow gains that are also present after standardization. The median gain finishes only after 37.5% of training. The concepts with the largest gains keep improving until 25–50% of training: capture ($+25.8$ points), side in check ($+11.6$), and material advantage ($+7.6$).

5.3 FAST ACQUISITION ENDS NEAR THE SPECTRAL REBOUND

We count training doublings between the label-free rebound and the point where each task comes within 2 points of its final value.

- **O-RAN:** acquisition finishes a median of 1.1 doublings after the rebound (62% of cells within 2 doublings).

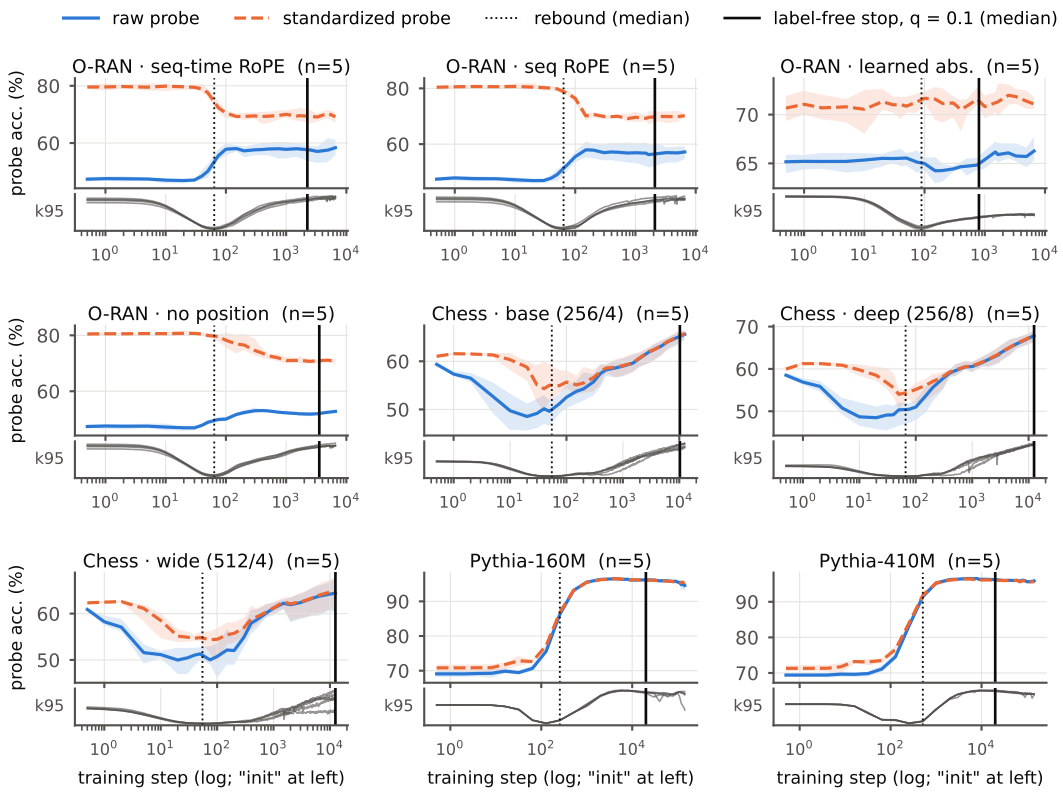


Figure 2: Probe accuracy and k_{95} across domains and architectures (five seeds each; line: seed mean, band: seed range). Blue: raw-feature probe; orange dashed: the same probe on standardized features. Grey strips: smoothed k_{95} per seed at the probed layer (layer 1 for O-RAN, the final layer otherwise). Dotted line: median rebound onset; solid line: median label-free stop ($q = 0.1$, final-layer clock). On O-RAN the raw probe's gain is complete shortly after the rebound, and the standardized probe is above it throughout; the learned-absolute control, whose probe input is high-rank from the start, shows no gain. On Pythia both probes rise together and finish near the rebound. On chess both probes dip during compression and keep improving long after the rebound.

Table 1: Acquisition regimes. Cells are task × layer × run, restricted to raw-probe gains above 2 points; gains are relative to initialization. "Retained" is the share of the summed raw-probe gain that the standardized probe also shows. "Finish" is the fraction of training until the task is within 2 points of its final value (median, interquartile range).

Domain	Cells	Raw gain (pp)	Retained	Finish (median, IQR)
O-RAN	543	14.5	−0.2%	1.5% (1.1–11%)
Pythia	260	24.3	92%	0.7% (0.7–1.4%)
Chess	511	10.5	86%	37.5% (16–75%)

- **Pythia:** two doublings at the final layer (92% within 2) and three across the other sampled layers, on published checkpoints that are spaced about a doubling apart near the rebound (steps 256, 512, 1,000, 2,000).
- **Chess:** acquisition continues for a median of 6.0 doublings after the rebound (only 8% within 2).

The gap is not a horizon artifact: O-RAN and chess both leave about 7 doublings between rebound and horizon. The rebound therefore marks the end of fast acquisition. The separation holds at every measured layer (all blocks for O-RAN and chess, five for Pythia; Appendix F). Pythia's first layer

is the main exception: its worst task converges more slowly, with an oracle stop at 14% of training against 0.7–1.4% for the other layers.

5.4 Fast O-RAN gains reflect readout accessibility

The O-RAN probe reads a concatenation of 21 end-of-sequence states (Section 4.3). In 12 additional runs (3 seeds of each of 4 architectures) we recorded the spectrum of this probe input.

- **RoPE variants (9 runs):** at initialization the probe input is extremely low-rank ($k_{95} \approx 3$). The raw probe's accuracy rises (47% to 53–60%) as that spectrum expands to 4–11 dimensions by step 150. Rank correlation over steps 0–300: 0.43–0.92. Over the same interval, the standardized probe *declines* from about 80% to 68–73%.
- **Learned absolute positions (3 runs):** the probe input is already high-rank at initialization ($k_{95} \approx 36$). The raw probe starts at 64–66% and gains nothing.

On O-RAN what pretraining changes is accessibility: within 1–2% of training, and timed by the spectral rebound, it reorganizes the representation so that an unnormalized linear readout can use it, to a degree that depends on the architecture (e.x., on layer 1, from about 47% at initialization to 58–62% across the RoPE variants). The probe input's spectrum expands from about 3 to 4–11 effective dimensions over the same interval: training improves the conditioning of the representation, which is what a plain frozen-feature readout needs. A standardized probe at initialization, which on O-RAN already exceeds the trained raw probe (Section 5.2), is therefore a free check of how much pretraining can add. Because 88% of raw-probe fits on O-RAN reach the 200-epoch cap, readout optimization is part of what is measured.

5.5 Language-model loss does not mark the end of acquisition

Convergence of the next-token loss is the default stopping signal. On Pythia and O-RAN the loss keeps improving long after probe acquisition has ended, and on chess its rate says nothing about how much probe gain remains (Appendix D).

- **Pythia** When the probes finish (step 1,000, 0.7% of training), a median 19% of the model's total held-out loss improvement is still to come. For Pythia-410M, held-out loss falls from 5.6 to 4.1 nats after that point while no probe improves by more than 2 points. It is still falling 1–2% per doubling at 14% of training, so classic early stopping trains to 77–100%.
- **O-RAN** Training loss is still falling 46% per doubling when the layer-1 probes finish, and 22% per doubling at half of training.
- **Chess** Loss falls 15% per doubling one doubling after the rebound, similar to Pythia's rate when its probes have finished, yet chess probes still gain a further 7.4 points.

Loss convergence therefore does not indicate when the representation is ready. A tuned slowdown threshold on the loss can work within one domain. For example, on Pythia, stopping once held-out loss improves by less than 5% per doubling stops at 3.5–6% of training with at most 0.4 points of probe loss. It does not, however, transfer. On chess the same threshold stops 3 of 15 runs at 7–22% of training on transient loss plateaus, and on O-RAN it stops only 17 of 50 runs, at 52–97% of training. This motivates the utility of a stopping condition not based on loss convergence.

5.6 Label-free stopping

Table 2 compares the rule, at the single setting $q = 0.1$, with classic early stopping. On Pythia it stops at 14% of training against 77–100% for early stopping, at a mean cost of 0.2 points (92% of layer-runs pass the strict test); the larger models behave the same, with Pythia-1B, 1.4B, and 6.9B stopping at 21%, 14%, and 28% of training and mean shortfalls of 0.1–0.4 points. On O-RAN it stops at 35% against 57%, at slightly lower quality (61% against 67% of layer-runs pass; mean shortfall 0.9 against 0.6 points). Pass rates on O-RAN are limited by probe noise rather than by the stopping point: with about 250 test scenarios per run, even stopping at 75% of training passes only 79% of layer-runs, and the rule's mean shortfall is 0.9 points. On chess the rule declines to stop 10 of 15 runs. Of the five it stops, four stop late (52–85% of training, 1–7 points short), and one stops at step

Table 2: Label-free stopping (rate-to-peak, $q = 0.1$, final-layer clock) against classic early stopping on the language-model loss (patience 3). Cells: median stop as a percentage of training / percentage of layer-runs in which every real task is within 2 points of its horizon value. "Oracle": median earliest passing checkpoint, using labels. Pythia early stopping: the 160M and 410M runs with held-out loss.

	Rule ($q = 0.1$)	Early stopping	Oracle
Pythia	14 / 92	77–100 / 100	1.4
O-RAN	35 / 61	57 / 67	7.3
Chess	100 / 85	96 / 78	75

440, 22.7 points short; early stopping instead stops some chess runs at 15–24% of training on loss plateaus. Per architecture, the rule is non-inferior to training to the end at a 5-point margin for all 10 O-RAN architectures, all 3 chess architectures, and Pythia 160M and 410M (one-sided Wilcoxon on per-run mean shortfall, $p = 0.031$ each, the smallest attainable with five seeds; Table 9). The rule is conservative relative to the label-using oracle, which it trails by about three doublings on Pythia and two on O-RAN as it waits until rank growth has slowed, which on Pythia happens several doublings after the probes finish (Figure 1).

5.7 A LOWER BOUND ON TRAINING BEFORE COMPARING CANDIDATES

O-RAN (10 architectures $\times$ 5 seeds): at layer 1, agreement with the final order is 0.19 at initialization, 0.62 at the rebound, and 0.83 one doubling later. At the final layer, eight of the ten architectures end within about 0.5 points of each other, inside the seed spread (1.4 points), so there is no ranking to recover. The rebound is therefore a necessary lower bound, not a sufficient one: comparisons made before it are unreliable in every domain, and where rank keeps expanding, as in chess, they remain unreliable until acquisition ends. This is the same regime in which the stopping rule declines to stop.

6 LIMITATIONS

Stopping. Within a single domain, a fixed training fraction or loss threshold tuned for that domain can naturally match or beat the rule; its advantage is that one setting transfers. At $q = 0.1$ the rule trails the label-using oracle by two to three doublings on O-RAN and Pythia, and it stops one of 15 chess runs at 3.5% of training, during a transient dip in rank in the middle of expansion. Appendix E reports the full range of q. O-RAN's strict pass rates are limited by its small test sets.

Scope. O-RAN has about 250 test scenarios per run, so single-run pass/fail at a 2-point tolerance is noisy. Natural-language results use published Pythia trajectories up to 6.9B parameters and a coarse checkpoint grid near the rebound. All evaluations use linear probes on frozen representations and we do not measure fine-tuning or nonlinear heads.

7 CONCLUSION

Across O-RAN traces, chess, and natural language, the spectral compression-to-expansion transition marks when fast probe acquisition ends. Which targets finish there depends on the target, and a fast gain may reflect readout accessibility rather than newly learned structure, as on O-RAN.

These findings inform three practical decisions: (i) a label-free stopping condition for frozen encoders that waits for spectral growth to slow after the rebound; (ii) a caution for interpreting probe learning curves: pair raw probes with scale-invariant ones and with the spectrum; (iii) a minimum training length before candidate architectures can be meaningfully compared. Convergence of the language-model objective is not the right stopping signal for reusable frozen backbones; the representation's spectrum is a better one.

AI USE STATEMENT

Generative AI (large language model assistants GPT6, OpenAI and Claude Opus 5.5, Anthropic) was used to migrate this text to the ICLR template, and to draft and aid writing. It was used to surface prior and related work. In research execution it helped to implement experimental code, dispatch and monitor training runs, and also helped generate figures. It also proposed analyses which the authors evaluated before adoption. The authors directed the research, made all scientific decisions, checked the code, numbers, and claims, and take full responsibility for the content.

REFERENCES

3GPP. Universal Mobile Telecommunications System (UMTS); LTE; 5G; General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U) (3GPP TS 29.281 version 19.2.0 Release 19). Technical Specification ETSI TS 129 281 V19.2.0, ETSI, January 2026a. URL https://www.etsi.org/deliver/etsi_ts/129200_129299/129281/19.02.00_60/ts_129281v190200p.pdf.

3GPP. 5G; NG-RAN; NG Application Protocol (NGAP) (3GPP TS 38.413 version 19.2.0 Release 19). Technical Specification ETSI TS 138 413 V19.2.0, ETSI, April 2026b. URL https://www.etsi.org/deliver/etsi_ts/138400_138499/138413/19.02.00_60/ts_138413v190200p.pdf.

3GPP. 5G; NG-RAN; F1 Application Protocol (F1AP) (3GPP TS 38.473 version 19.2.0 Release 19). Technical Specification ETSI TS 138 473 V19.2.0, ETSI, April 2026c. URL https://www.etsi.org/deliver/etsi_ts/138400_138499/138473/19.02.00_60/ts_138473v190200p.pdf.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2016. URL https://arxiv.org/abs/1610.01644.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances, 2021. URL https://arxiv.org/abs/2102.12452.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430. PMLR, 2023. URL https://proceedings.mlr.press/v202/biderman23a.html.

ETSI. Publicly Available Specification (PAS); O-RAN Architecture Description (O-RAN.WG1.TS.OAD-R004 - v14.00). Technical Specification ETSI TS 103 982 V14.0.0, European Telecommunications Standards Institute, March 2026. URL https://www.etsi.org/deliver/etsi_ts/103900_103999/103982/14.00.00_60/ts_103982v140000p.pdf.

Niklas Fiekas. python-chess: a chess library for Python. https://pypi.org/project/chess/. URL https://pypi.org/project/chess/. Accessed: 2026-09-26.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks, 2019. URL https://arxiv.org/abs/1909.03368.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability, 2023. URL https://arxiv.org/abs/2301.05217.

Open5GS. Open5GS: Open Source implementation for 5G Core and EPC. https://github.com/open5gs/open5gs. Accessed: 2026-05-07.

Vardan Papyan, X. Y. Han, and David L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40): 24652–24663, September 2020. ISSN 1091-6490. doi: 10.1073/pnas.2015509117. URL http://dx.doi.org/10.1073/pnas.2015509117.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202/.

Margaret H. Pinson, Mark Poletti, Julie E. Kub, Jeremy Glenn, Silas T. Thompson, Angela McCrory, Ashok Misra, Robert Kupsh, Naser Areqat, Shariq Ashfaq, T. Lauriston Hardin, Keith Hartley, Dolores A. Shaffer, and T. K. Woodward. Test Plans, Results, and Lessons Learned About Open RAN Integration During the NTIA 5G Challenges. Special Publication NTIA SP-25-578, National Telecommunications and Information Administration, Institute for Telecommunication Sciences, Boulder, Colorado, May 2025. URL https://its.ntia.gov/publications/download/SP-25-578.pdf.

Jon Postel. Internet Control Message Protocol: DARPA Internet Program Protocol Specification. RFC 792, RFC Editor, September 1981. URL https://www.rfc-editor.org/info/rfc792/.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets, 2022. URL https://arxiv.org/abs/2201.02177.

Lutz Prechelt. Early stopping—but when? In Genevieve B. Orr and Klaus-Robert Müller (eds.), *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, pp. 55–69. Springer, Berlin, Heidelberg, 1998. ISBN 978-3-540-65311-0. doi: 10.1007/3-540-49430-8_3. URL https://doi.org/10.1007/3-540-49430-8_3.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1708.00489.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, Department of Computer Sciences, 2009. URL https://burrsettles.com/pub/settles.activelearning.pdf. Updated January 26, 2010.

Adam Shai, Loren Amdahl-Culleton, Casper L. Christensen, Henry R. Bigelow, Fernando E. Rosas, Alexander B. Boyd, Eric A. Alt, Kyle J. Ray, and Paul M. Riechers. Transformers learn factored representations, 2026. URL https://arxiv.org/abs/2602.02385.

Abhinav Shrivastava, Abhinav Gupta, and Ross B. Girshick. Training region-based object detectors with online hard example mining. *CoRR*, abs/1604.03540, 2016. URL https://arxiv.org/abs/1604.03540.

Leslie N. Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017. URL https://arxiv.org/abs/1506.01186.

Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates, 2017. URL https://arxiv.org/abs/1708.07120.

Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1711.00489.

Software Radio Systems. srsRAN 4G: Open source SDR 4G software suite. https://github.com/srsran/srsRAN_4G, a. Accessed: 2026-05-07.

Software Radio Systems. srsRAN Project: Open source O-RAN 5G CU/DU solution. https://github.com/srsran/srsRAN_Project, b. Accessed: 2026-05-07.

Jascha Sohl-Dickstein. The boundary of neural network trainability is fractal, 2024. URL https://arxiv.org/abs/2402.06184.

Randall Stewart, Michael Tüxen, and Karen Nielsen. Stream Control Transmission Protocol. RFC 9260, Internet Engineering Task Force, June 2022. URL https://www.rfc-editor.org/rfc/rfc9260.pdf.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding, 2021. URL https://arxiv.org/abs/2104.09864.

The Stockfish developers. Stockfish. https://stockfishchess.org/. URL https://stockfishchess.org/. Accessed: 2026-09-26.

The ZeroMQ Project. The ZeroMQ Project. https://github.com/zeromq. Accessed: 2026-05-07.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method, 2000. URL https://arxiv.org/abs/physics/0004057.

Oskar van der Wal, Pietro Lesci, Max Müller-Eberstein, Naomi Saphra, Hailey Schoelkopf, Willem Zuidema, and Stella Biderman. PolyPythias: Stability and outliers across fifty language model pre-training runs. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=bmrYu2Ekdz.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation, 2018. URL https://arxiv.org/abs/1811.10959.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks?, 2014. URL https://arxiv.org/abs/1411.1792.

A ORAN-DIAG DETAILS

Table 3: ORAN-Diag-v1 raw scenario distribution. Each non-nominal scenario contains exactly one injected root-cause condition by construction.

Raw scenario condition	Count	Intervention
NOMINAL_0	400	No injected fault
DELAY_F1C	178	Temporary SCTP gate delay on DU–CU F1-C
DROP_F1C	178	SCTP blackhole on DU–CU F1-C
PLMN_MISMATCH	178	DU PLMN mutation
TAC_MISMATCH	178	DU TAC mutation
UE_IMSI_MISMATCH	178	UE subscriber identity mutation
LOW_TIMEOUTS	178	Tight CU protocol timers
DROP_F1U	178	UDP/2152 blackhole
LOSS_F1U	177	nth-packet UDP/2152 loss
SMALL_GTPU_QUEUE	177	Reduced CU GTP-U queue size

Scenarios, tests, and per-test records. The natural unit of the dataset is a *scenario*, not an individual packet trace. A scenario corresponds to an underlying equipment condition or injected fault. For each scenario $s \in \mathcal{S}$, the harness executes a fixed suite of $m = 21$ tests. We write the resulting per-test records as

$$\{(x_{s,j}, \tau_{s,j}, v_{s,j}, m_{s,j}) : j = 1, \ldots, 21\},$$

where $x_{s,j}$ is the tokenized packet-derived sequence for test j, $\tau_{s,j}$ is the corresponding sequence of packet or token timestamps, $v_{s,j} \in \{\text{PASS}, \text{FAIL}\}$ is the harness observation for that test, and $m_{s,j}$ contains metadata such as the scenario identifier, test identifier, and raw injected condition. The fixed test suite contains 14 prefix tests, 6 composite tests, and one traffic-volume test. The suite spans transport association, F1 setup, UE registration, PDU session establishment, user-plane reachability, throughput, and load/impairment checks.

Table 4: ORAN-Diag-v1 dataset summary.

Item	Value
Scenarios	2,000
Tests per scenario	21
Per-test records	42,000
Raw dataset size	2.7 GB
Overall test-result distribution	24,733 PASS / 17,267 FAIL
Median model-view length	20 lines / 654 bytes
Maximum model-view length	76 lines / 4,437 bytes
Vocabulary size	113 tokens
Test-identity tokens	21
Total token count	11.3M
Mean tokenized sequence length	268.2 tokens
Maximum tokenized sequence length	792 tokens

Packet-derived model views. Each per-test record contains a symbolic sequence derived from packet evidence and harness observations. The parser maps lab IP addresses to semantic roles such as CU, DU, CU-UP, AMF, UPF, and UE; decodes F1AP, NGAP, SCTP, GTP-U, ICMP (3GPP, 2026c;b; Stewart et al., 2022; 3GPP, 2026a; Postel, 1981), and iperf-related traffic; buckets packet lengths; and normalizes high-cardinality packet fields such as sequence numbers, stream identifiers, acknowledgements, TSNs, SSNs, windows, and MAC fields. Each packet-derived event is serialized as `PROTO SRC -> DST LEN_BUCKET INFO`. User-plane bursts are summarized as `[UPLANE_BURST] COUNT_BUCKET DURATION_BUCKET`, preserving coarse volume and duration information without representing every user-plane packet individually. This reduces context length while retaining traffic-intensity and impairment signals. The final tokenized dataset uses a deterministic rule-based tokenizer rather than a learned subword tokenizer. Each sample begins with `[SOS]` and a test-identity token, contains tokenized model-view lines separated by `[EOL]`, and ends with `[EOS]`. Packet timestamps are retained as a parallel sequence and are available to time-aware positional mechanisms.

Scenario-level labels. The downstream labels describe the latent equipment condition or diagnostic state of the scenario, not the class of an individual packet and not the outcome of an isolated trace row. Consequently, all per-test records generated under the same scenario share the same downstream label. The primary label y_s^{hard} maps the raw design-time condition to one of $K_{\mathrm{hard}} = 7$ root fault families and is used only for scenario-level splits and downstream probing of frozen representations.

Self-supervised language-model training. For a tokenized trace $x_{s,j} = (x_{s,j,1}, \ldots, x_{s,j,T})$, the model is trained to predict each next token from the preceding token and timestamp history, $\mathcal{L}_{\mathrm{LM}}(\theta) = -\sum_{s,j,t} \log p_\theta(x_{s,j,t+1} \mid x_{s,j,\le t}, \tau_{s,j,\le t})$. The downstream labels are never provided as model inputs or language-model targets. Their only supervised use is in downstream linear-probe evaluation, including stratifying splits for this task.

B Architectures and compute

Table 5: O-RAN architectures. All architectures are trained under the same checkpointing and evaluation schedule within each seed.

	Architecture	Change from base model	Purpose
1	No position	no positional encoding	position-control baseline
2	Learned absolute	learned absolute position embeddings	standard learned-position control
3	Sequence RoPE	sequence-position RoPE only	sequence order without time
4	Time RoPE	continuous-time RoPE only	timing without sequence RoPE
5	Seq-time RoPE	factored sequence-time RoPE	base architecture
6	Seq-time RoPE + RMSNorm	RMSNorm instead of LayerNorm	normalization axis
7	Seq-time RoPE, 8/24 split	8 sequence dims, 24 time dims per head	time-heavy RoPE allocation
8	Seq-time RoPE, 24/8 split	24 sequence dims, 8 time dims per head	sequence-heavy RoPE allocation
9	Seq-time RoPE, 3 layers	reduced depth	depth axis
10	Seq-time RoPE, 5 layers	increased depth	depth axis

Table 6: Chess architectures. All use RoPE, LayerNorm, AdamW (learning rate 10^{-3}), batch size 32, context up to 256 tokens, and 12,500 updates over 400,000 disjoint games (one pass, no repeated data).

Architecture	Width d	Layers L	Heads H	Seeds
Base	256	4	4	5
Deep	256	8	4	5
Wide	512	4	8	5

Compute. The O-RAN, chess, and Pythia runs, including checkpoint extraction, spectra, and all probe fits, required about 40 GPU-hours on an RTX 4090 and an RTX PRO 6000.

C All ten O-RAN architectures

D Language-model loss and probe accuracy

Pythia held-out loss is the mean next-token cross-entropy on the 22,423 tokens of the UD-EWT development and test sentences used for probing, at each of the 35 published checkpoints (160M and 410M, three seeds each). Chess and O-RAN loss is the per-update training loss, smoothed with an exponential moving average (half-life 50 updates). Chess training is a single pass over disjoint games, so its training loss is held-out. O-RAN training repeats the corpus 20 times. The Spearman correlation between loss and mean probe accuracy across probe checkpoints is high before the rebound in O-RAN and Pythia (medians 0.85 and 0.93) and moderate in chess (0.55). After the rebound it is 1.00 in

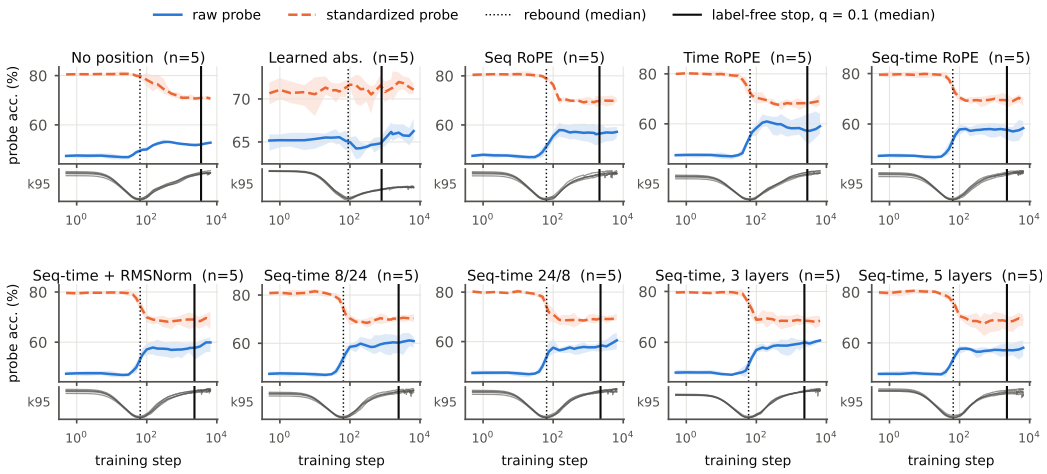


Figure 3: O-RAN, all ten architectures (five seeds each; layer 1; conventions as in Figure 2). Every position-aware variant gains at the rebound and the standardized probe drops at the same point; the no-position control gains less and later; the learned-absolute control, whose probe input is high-rank from initialization, shows no gain. Solid line: median label-free stop ($q = 0.1$, final-layer clock).

chess, where both keep improving, 0.68 on O-RAN, and 0.27 on Pythia. Correlation therefore does not separate the regimes; the rate comparison in Table 7 does.

Table 7: Language-model loss at the end of probe acquisition (medians over runs). "Probes finish" is the label-using oracle stop (every real task within 2 points of its horizon value). Loss rates are relative decreases over the preceding doubling of training. Pythia: held-out next-token loss on the probing text; chess: training loss on a single pass; O-RAN: training loss.

	Probes finish (% of training)	Loss improvement still to come	Loss drop per doubling at finish / at 2τ	Probe gain after 2τ (points)
O-RAN (layer 1)	3.0	25%	46% / 68%	2.5
Pythia (final layer)	0.7	19%	12% / 12%	3.3*
Chess (final layer)	100	–	– / 15%	7.4

*Pythia's 2τ falls between published checkpoints 512 and 1,000, where the probes are still rising.

E STOPPING DETAILS

Table 8: Rate-to-peak rule across q (final-layer clock). Cells: median stop (% of training) / percentage of layer-runs passing the strict test. $q = 0.1$ is the setting used in the main text. The one-doubling guard (Section 4.4) leaves O-RAN and Pythia unchanged at every q shown; without it, at $q = 0.1$ the rule fires at the chess rebound on 5 of 15 runs, and the chess pass rate falls from 85% to 52%.

q	O-RAN	Pythia	Chess
0.5	6.4 / 46	2.1 / 74	2.0 / 6
0.2	16 / 55	4.9 / 80	48 / 20
0.1	**36 / 61**	**14 / 92**	**100 / 85**
0.05	81 / 81	21 / 97	100 / 100
0.02	100 / 99	42 / 99	100 / 100

F PER-LAYER RESULTS

Table 9: Per-architecture non-inferiority at $q = 0.1$: median stop and median classic early-stopping point (% of training), mean and worst per-run mean shortfall (points, every real task at every layer), and the one-sided Wilcoxon p-value for non-inferiority at a 5-point margin (five seeds). Pythia 1B, 1.4B, and 6.9B (one seed each) stop at 21%, 14%, and 28% with mean shortfalls of 0.12, 0.19, and 0.40 points.

Domain	Architecture	Stop	Early stop	Mean	Worst	p
O-RAN	No position	54	63	0.87	1.57	0.031
	Learned abs.	12	48	0.67	1.30	0.031
	Seq RoPE	32	51	0.26	0.60	0.031
	Time RoPE	43	71	0.84	2.05	0.031
	Seq-time RoPE	34	61	0.49	1.43	0.031
	Seq-time + RMSNorm	35	52	1.19	2.19	0.031
	Seq-time 8/24	37	56	0.66	1.39	0.031
	Seq-time 24/8	34	52	1.69	2.37	0.031
	Seq-time, 3 layers	36	65	1.52	3.08	0.031
	Seq-time, 5 layers	34	61	0.79	1.51	0.031
Chess	Base	82	98	1.01	3.34	0.031
	Deep	100	96	0.00	0.00	0.031
	Wide	100	88	0.01	0.05	0.031
Pythia	160M	14	91	0.17	0.35	0.031
	410M	14	100	0.21	0.40	0.031

Table 10: Per-layer acquisition (raw-probe gains > 2 points). "Retained" is the share of the raw gain also present under the standardized probe; "Finish" is the median fraction of training until the task is within 2 points of its final value.

Domain	Layers	Cells	Retained	Finish
O-RAN	1–5 (per layer)	15–140	-1.1% to 0.9%	1.5%
Pythia	5 relative depths	52	88–94%	0.7%
Chess	1–8 (per layer)	32–124	71–92%	16–44%

Table 11: Per-layer stopping (rate-to-peak rule, $q = 0.1$, each layer's own clock). Cells: median stop (% of training) / pass rate (%) / median oracle stop (%). Pythia layers are grouped by relative depth across sizes; chess layers 5–8 exist only in the deep architecture.

Domain	Layer	Stop / pass / oracle
O-RAN	1 / 2 / 3	48 / 73 / 3.0 61 / 61 / 4.6 60 / 74 / 10
	4 / 5	37 / 52 / 38 34 / 100 / 4.6
Pythia	first / $\frac{1}{4}$	98 / 100 / 14 21 / 92 / 1.4
	$\frac{1}{2}$ / $\frac{3}{4}$ / final	2.1 / 100 / 1.4 2.8 / 100 / 1.4 14 / 100 / 0.7
Chess	1 / 2–4	100 / 100 / 25 98–100 / 67–93 / 75
	5–8	83–100 / 20–100 / 75–100