

Linear Layer Repair: Residual-Controlled Layer Profiling for Fault-Tolerant Transformer Inference

Jiya Sheetal Rathi

Illinois Institute of Technology
Chicago, Illinois, USA
jrathi@hawk.illinoistech.edu

André Bauer

University of Chicago
Booth School of Business
Chicago, Illinois, USA
andre.bauer@bauer-research.net

Abstract

Deciding which transformer layers can be cheaply replaced - when a node fails mid-computation, or when a block is pruned - requires knowing what each layer actually computes. A straightforward approach is to fit a linear map between consecutive hidden states and read off its R^2 . In our analysis, these fits look excellent, exceeding R^2 of 0.97. However this number is largely an artifact of the residual stream: an untrained network scores just as well. The informative quantity is instead how linearly predictable a layer’s update $h_{L+1} - h_L$ is. Measured this way, and across six models from three architecture families, layers fall into a clear U-shaped pattern: those at the beginning and end of the stack perform large but almost perfectly linear transformations, while those in the middle perform small nonlinear ones. Given these insights, we present **Linear Layer Repair**, a policy that reads a layer’s profile and prescribes one of three responses when it goes missing: (i) skip it, (ii) swap in a fitted linear map, or (iii) declare it unrecoverable. Using WikiText-103, we test Linear Layer Repair by removing layers and measuring the damage. The results show that the linear map beats plain skipping at all 16 LLaMA-3.2-1B layers and 26 of 28 in Qwen2-1.5B; replacing a single middle layer costs under half a point of perplexity against a baseline of 15; even four consecutive layers cost only ~ 2.5 points; and at an early layer, where skipping raises perplexity above 10^5 , the patch reduces it more than a thousand-fold. Task-level evaluations show corresponding recovery on prediction-sensitive behavior.

1 Introduction

What does an individual transformer layer compute, and how much of that computation can a cheap substitute reproduce? The question has an immediate practical form: when a large language model is served across several devices and one fails, the layers it held disappear mid-forward-pass with no time

to retrain around the gap, and a system that wants to keep running needs a small, fixed substitute computed in advance. But not every layer admits one. Answering which do requires a measurement of layer computation that is meaningful in the first place — and we show the conventional one is not.

To answer this, we first have to understand the task of a layer. One line of recent work (Razhigaev et al., 2024) approaches this by asking whether consecutive layers are related linearly: if $h_{L+1} \approx Wh_L + b$ fits well, then an expensive block might be swapped for a single matrix multiply. Across the decoder models we study, the held-out R^2 -score never drops below 0.97. However, that number is misleading, and the reason is the residual stream. A transformer block does not replace its input; it computes $h_{L+1} = h_L + \Delta(h_L)$ and adds the result to an input that flows through untouched. In the middle of the network Δ is small next to h_L , so a map that barely differs from the identity will score a high R^2 no matter what the block actually computed. To demonstrate this, we ran the same fitting procedure on randomly initialized controls spanning the decoder-only and encoder-only models. These networks have learned nothing, and they still reach full-state R^2 between 0.95 and 0.99 (Section 3). As this measure cannot distinguish between a trained and randomly initialized model, we investigated how linearly predictable the update $\Delta_L = h_{L+1} - h_L$ is on its own.

Over six models from three architecture families, the update’s linear predictability traces a U down the depth of the network. The layers at either end make large changes that are almost perfectly linear—LLaMA-3.2-1B’s layer 1 is 99.6% linearly predictable—while the layers in between make small changes that resist linear description. Untrained networks show no such structure. In contrast, despite their architectural differences, all trained models converge to similar Δ - R^2 values in their middle layers.

Building on this observation, we propose **Linear Layer Repair** (Section 4): a policy that maps a layer’s measured profile to a repair action. The U-shaped profile naturally categorizes layers into three: layers whose output is dominated by the residual are simply skipped; layers whose update is large but linear are replaced by a fitted map; layers where neither condition holds are flagged as unrecoverable without their true weights. We validate the policy by removing layers or replacing them with the fitted maps in the running model and measuring perplexity and zero-shot task accuracy (Section 5).

Our contributions are:

1. A residual-controlled measurement of layer computation, and the finding that it follows a U-shaped, learned profile consistent across encoder-only, decoder-only, and encoder-decoder models (Section 3).
2. **Linear Layer Repair**, a policy that predicts, per layer, which of three repair actions preserves model function, validated by direct replacement (Sections 4, 5).
3. A characterization of the practical envelope: the fitted maps are effectively full-rank, a matched-budget MLP does not improve on them, and they transfer across text domains and lengths but not reliably across early layers into code (Section 6).
4. A meta-observation of independent interest: the conventional inter-layer R^2 is reproduced by untrained networks, and we specify the controls (identity baseline, Δ -target, random initialization) that a valid analysis should report.
5. An open-source pipeline that fits exact ridge maps for every layer pair, every regularization strength, and both targets from a single forward pass, making the study reproducible on a single consumer GPU.

2 Related Work

Linearity of transformer layers. Recent work reports that transformations between consecutive decoder layers are close to linear, and observes that the effect is substantially attributable to the residual stream, with linearity scores dropping once the residual component is removed (Razhigaev et al.,

2024). We extend this in three ways. First, we add a random-initialization control, which shows that the uncontrolled metric fails to distinguish trained from untrained networks at all. Second, we characterize the depth structure of the residual-controlled signal, which is U-shaped and learned rather than uniform, and show that it holds across three architecture families. Third, we connect the measurement to function by replacing layers and evaluating the resulting model. Yang et al. (2023) show ReLU transformers admit local linear decompositions, a complementary formal perspective.

Learned maps between representations. The Tuned Lens (Belrose et al., 2023) fits affine maps from intermediate states to the final layer as interpretability instruments. Our maps are fitted between adjacent layers and used as functional substitutes, but share the premise that a learned affine map can capture a meaningful portion of what a stack computes.

Layer redundancy and pruning. LayerDrop (Fan et al., 2019) applies structured layer dropout to enable depth reduction. ShortGPT (Men et al., 2024) and Gromov et al. (2024) identify redundancy in deeper LLM layers and prune whole blocks by inter-layer similarity, typically followed by healing or fine-tuning. Our setting differs in both objective and constraints: these methods ask which layers can be *permanently deleted* from a model one controls and may retrain, whereas we ask which layers can be *transiently replaced* by a pre-computed drop-in substitute with no retraining—the operating regime of a node failure. Our baseline is therefore the skipped layer itself, and our contribution is the mapping from a layer’s measured profile to the repair action that preserves function.

Linearization in dynamical systems. That non-linear dynamics admit local linear descriptions is classical—Koopman operator theory (Brunton et al., 2021) and Kalman filtering (Kalman, 1960)—and motivates the linear-approximation hypothesis, but does not predict which layers are approximable, nor at what functional cost.

3 Measuring Layer Computation

3.1 Setup

We study six pretrained models: Qwen2-1.5B (Yang et al., 2024) and LLaMA-3.2-1B (Grattafiori et al., 2024) (decoder-only); BERT-base (Devlin

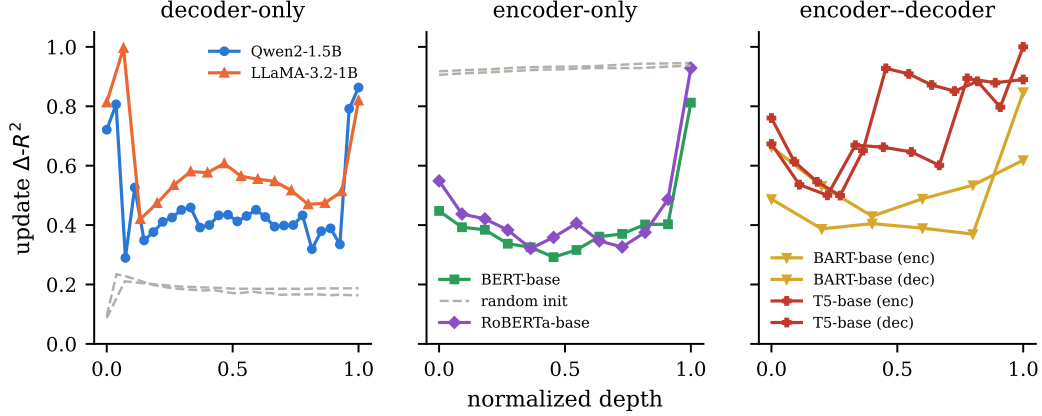


Figure 1: Update linearity (ΔR^2) across normalised depth for all six models, grouped by architecture family. The U-shape—high at the boundaries, low in the middle—appears in every model. Dashed grey curves show the available randomly initialised controls; their structureless profiles support the interpretation that the U-shape is learned. (T5-base’s final two encoder layers are excluded; see text.)

et al., 2019) and RoBERTa-base (Liu et al., 2019) (encoder-only); and BART-base (Lewis et al., 2019) and T5-base (Raffel et al., 2020) (encoder-decoder, both stacks analysed). For each pair of consecutive hidden states we fit a ridge regression $h_{L+1} \approx Wh_L + b$, selecting the regularisation strength ($\lambda \in \{10^{-2}, \dots, 10^3\}$) on a held-out validation split of $\sim 30k$ tokens. Fitting data are 6,000 OpenWebText documents tokenised with each model’s own tokeniser, truncated to 512 tokens, with padding masked out. To make the study tractable on a single GPU, we accumulate per-layer sufficient statistics $\Sigma xx^\top, \Sigma xy^\top$ in one forward pass over roughly 3M tokens of OpenWebText (Gokaslan and Cohen, 2019); the exact centred ridge solution for any regularisation strength, and for the update target below, is recovered from these statistics. We report held-out R^2 throughout. To quantify stability, every map is refit three times on disjoint thirds of the fitting data; the standard deviation of held-out R^2 across refits never exceeds 0.002 for any layer of any model (per-layer values accompany the released pipeline), so we omit error bars, which would be smaller than plot markers.

3.2 Measures

We compute, per layer:

- **Full-state R^2** : predicting h_{L+1} from h_L . This is the conventional metric.
- **Identity R^2** : the fit of $h_{L+1} \approx h_L$, i.e., the *residual share*—how much of the output is the input, carried forward.

- **ΔR^2** : predicting the update $\Delta_L = h_{L+1} - h_L$. This isolates the computation the layer performs from what it passes through, and is our primary measurement.

3.3 The conventional metric is uninformative

Averaged over non-boundary layers, full-state $R^2 = 0.977$ (Qwen2) and 0.984 (LLaMA), with encoder and encoder-decoder models similar (0.84–0.96). Two controls show this number says little about learned computation. The identity baseline alone reaches 0.955–0.962 in the decoders: most of the fit is the residual. And on randomly initialized controls spanning the decoder-only and encoder-only models, full-state R^2 still climbs with depth to 0.95–0.99 despite the absence of any training. A metric reproduced by untrained networks cannot serve as evidence of learned linear structure. We therefore base our analysis on ΔR^2 .

3.4 A learned, U-shaped profile

Figure 1 plots ΔR^2 against normalized depth. Three things stand out, and all three hold across every model we tested.

The first is that linear structure does not vanish once the residual is controlled for—it just shrinks to a reasonable size. Through the middle of each network, ΔR^2 sits between 0.35 and 0.53 for the decoders, around 0.36 for the encoders, and 0.40–0.46 in the encoder-decoder stacks. So a real fraction of each update is linear, but nothing like the near-unity that full-state R^2 advertised.

The second is the shape. At the boundaries the curve jumps: ΔR^2 reaches 0.72 and 0.86 at

Qwen2’s first and last layers, 0.81 and 0.82 for LLaMA, 0.45 and 0.81 for BERT, 0.55 and 0.93 for RoBERTa. LLaMA’s layer 1 is the cleanest case we saw: its update is 99.6% linearly predictable, even though the identity baseline captures almost none of its output. Read together, the two facts say that boundary layers apply large but nearly linear transformations—most naturally understood as basis changes, into and out of the embedding geometry—while the interior layers apply small and genuinely nonlinear refinements. The Δ and identity curves are, more or less, reflections of each other.

The third is that none of this appears before training. In the randomly initialized controls the U-shape is simply gone (we return to why in Section 3.5), and trained models that start from very different architectural priors nonetheless converge onto the same mid-stack band. Training, then, is doing two things at once: carving the depth structure, and pulling otherwise dissimilar models toward a shared regime.

3.5 A note on the random-initialization control

The role of the random-init control differs between architectures, and this is itself informative. In decoders, untrained Δ - R^2 is flat and low (≈ 0.17): random blocks make small updates dominated by residual noise. In encoders, untrained Δ - R^2 is high (≈ 0.93) but equally *flat*: random encoder blocks make large updates that are near-linear projections of their input. In both cases the untrained profile is *structureless*—the U-shape is absent—whereas the trained profile is structured. The learned signature is therefore the *shape*, not the level. We verified that update magnitudes $\|\Delta\|/\|h\|$ are comparable between trained and untrained networks, ruling out a magnitude artifact. We report random-init controls for four models spanning both encoder-only and decoder-only families.¹

4 Linear Layer Repair

The profile from Section 3.4 decides what to do with a missing layer L based on how much of the residual passes through (identity R^2) and how linear its own update is (Δ - R^2). More precisely, the three actions are:

- **Skip** (high identity R^2). If the layer’s output is dominated by the residual, the layer can be omitted and its input passed through unchanged. This is the case for the bulk of mid-stack layers, where the update is small.
- **Linear patch** (low identity R^2 , high Δ - R^2). If the layer performs a large but nearly linear transformation, we substitute its fitted map $Wh_L + b$. This is the case at the stack boundaries, where skipping is catastrophic but the transformation is recoverable.
- **Unrecoverable** (low identity R^2 , low Δ - R^2). If the layer performs a large *and* substantially nonlinear transformation, neither substitute suffices; the layer must be replicated or the node restarted. Identifying these layers is operationally valuable: they are the layers to place on reliable hardware or replicate across nodes.

In our decoder experiments the branches separate cleanly without tuning: layers with identity $R^2 > 0.9$ (the mid-stack) fall to SKIP, and among the remainder, Δ - $R^2 \gtrsim 0.7$ marks the patchable boundary layers. We treat these as descriptive of an observed bimodal separation rather than calibrated constants, and note that the policy is deliberately a *joint* criterion: because absolute Δ - R^2 levels are architecture-dependent (Section 3.5), neither quantity alone suffices, but the identity share anchors the decision across architectures. Quantitatively, the profile predicts repair outcomes: across all 44 layers of both decoders, Δ - R^2 correlates with the log-perplexity recovery of the patch over skipping at Spearman $\rho = 0.65$ ($p < 10^{-5}$), and $\rho = 0.59$ ($p = 0.045$) when restricted to the twelve low-identity layers where patching is consequential.

Maps used for substitution are drop-in by construction: the centred ridge solution of Section 3 is algebraically identical to an uncentred affine map $h \mapsto hW + b$ with $b = \bar{y} - \bar{x}W$, so deployment applies W and b directly to raw hidden states. “Native representation space” means only that, unlike some prior analyses, we apply no ℓ_2 -normalisation or whitening to inputs before fitting. Because the policy is derived entirely from quantities measured in Section 3, no additional fitting is required at repair time beyond storing the per-layer maps.

¹T5-base’s final two encoder layers produced numerically unstable statistics under fp16 extraction, consistent with T5’s known large activation magnitudes, and are excluded from all summaries.

5 Validating the Policy

We replace layers in the running model, substituting each layer L either by the identity (SKIP) or its fitted map (LINEAR), and measure perplexity on the WikiText-103 test set (Merity et al., 2016). Encoder-decoder models are excluded from replacement: T5 threads relative position bias between blocks, which a drop-in patch would break, and our replacement experiments cover single-stack models only. Both contribute to the representational analysis only.

5.1 Single-layer replacement

Figure 2 shows three regimes, each predicted by the policy. *Mid-stack* substitution is inexpensive: Qwen2 stays within 15.6–17.5 perplexity against an intact baseline of 15.2; LLaMA, with fewer and thus more load-bearing layers, costs 20.4–27.3 against an intact baseline of 19.0. *Early* layers are catastrophic to skip but recoverable by patching: skipping Qwen2’s layer 0 gives perplexity 1.2×10^5 , the patch recovers it to 85.7; skipping LLaMA’s layer 1 gives 1584, the patch gives 26.0, consistent with that layer’s $\Delta-R^2 = 0.996$. *Late* layers are intermediate (Qwen2’s final layer: $37.4 \rightarrow 23.7$ under patching). The linear map outperforms skipping at 26/28 Qwen2 layers and 16/16 LLaMA layers. The exceptions are instructive: Qwen2’s layer 1 satisfies the patch criterion only marginally ($\Delta-R^2 = 0.81$, against boundary-layer values of 0.86–1.00 elsewhere) while its identity share ($R^2 = 0.02$) offers no fallback, and there the patch (169) is no better than skipping (147) — the edge case that motivates treating the thresholds of Section 4 as descriptive rather than calibrated, and the layer to which we return in Section 5.3. The second exception, layer 4, is immaterial: patch (17.48) and skip (17.26) differ by 0.2 perplexity against an intact baseline of 15.2, and the policy prescribes SKIP there in any case.

5.2 Multi-layer replacement (node failure)

Because a failed node can host several consecutive layers, we replace runs of $k \in \{1, 2, 3, 4\}$ adjacent layers (Figure 3). Qwen2 tolerates four consecutive mid-stack layers at 17.7 perplexity (+2.5 over intact) versus 19.3 for skipping. LLaMA degrades faster (33.2 at $k=4$) but benefits far more from substitution than skipping (95.3), and in the worst-placed window the patches convert a collapse into a functioning model ($2.5 \times 10^4 \rightarrow 298$). The sub-

Strategy		Boundary layers			
		L0	L1	L26	L27
Qwen2	skip	1.3×10^5	126.7*	17.2	33.4
	linear (full)	78.7	158.7	15.8	21.4
	neighbour reuse	3.4×10^6	6.7×10^5	17.6	46.3
	mixture $k=4$	68.3	84.8	15.7	20.5
		L0	L1	L14	L15
LLaMA	skip	2.5×10^4	1421.1	36.5	55.1
	linear (full)	34.4	24.3	23.7	30.9
	neighbour reuse	5.7×10^4	2.3×10^4	34.8	49.1
	mixture $k=4$	35.3	23.9	23.2	30.0

Table 1: Perplexity under alternative substitutes at the boundary layers (intact: Qwen2 ≈ 15.2 , LLaMA ≈ 19.0). *best prior option at Qwen2’s Layer 1. Reduced-rank rows omitted for space: no rank below full matched the full map on LLaMA; on Qwen2, $r=512$ improved on the full map at layer 0 only (65.5 vs. 78.7), a model-specific regularisation effect.

stitute for a window is the composition of the k per-layer maps, not a single map fitted from h_L to h_{L+k} ; since each is fitted on activations from the intact model, these are conservative lower bounds relative to sequential refitting.

5.3 Repairing the hard layers

For the boundary layers of both decoders, we compare four substitutes head-to-head: the full linear map, a *directly fitted* reduced-rank map (reduced-rank regression, not post-hoc truncation), reuse of the adjacent layer’s map, and a *mixture of linear maps*— $k=4$ ridge maps fitted to a k -means partition of the layer’s input space, with tokens routed to their nearest centroid’s map at repair time.² Table 1 reports the results.

The two models tell one story with different endings, and the difference is itself the confirmation. On Qwen2, the mixture matches or improves on the single map at every boundary layer despite a $4 \times$ data disadvantage—most dramatically at Layer 1, the one layer the policy flags as unrecoverable because the single map underperforms skipping. There the mixture reaches 84.8 perplexity against 126.7 for skipping: the layer’s resistance to a single linear description is substantially *piecewise*-linear structure, recoverable by routing among a few region-specific maps. Together with this layer’s domain-sensitive behavior on code (Section 6), a consistent picture emerges of early layers applying

²Mixture and reduced-rank candidates were fitted on a 1,500-document subsample (the full map uses all 6,000), and all strategies are compared on an identical held-out set; absolute values therefore differ slightly from Figure 2.

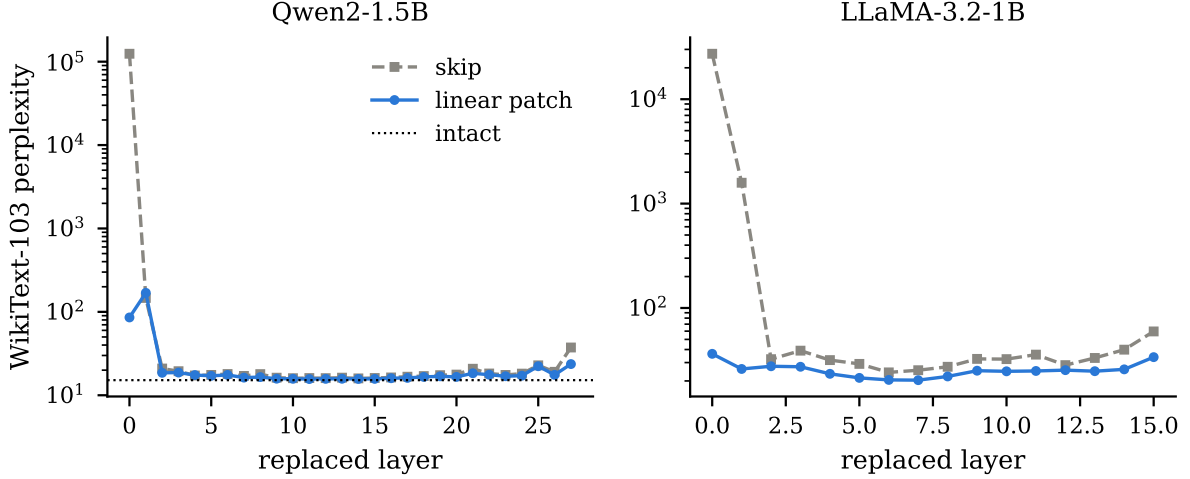


Figure 2: WikiText-103 perplexity when each layer is skipped (dashed) or replaced by its fitted linear map (solid), for the two decoder models. Dotted line: intact model. The linear map beats skipping at nearly every layer; the cost is small mid-stack and a rescue at the boundaries.

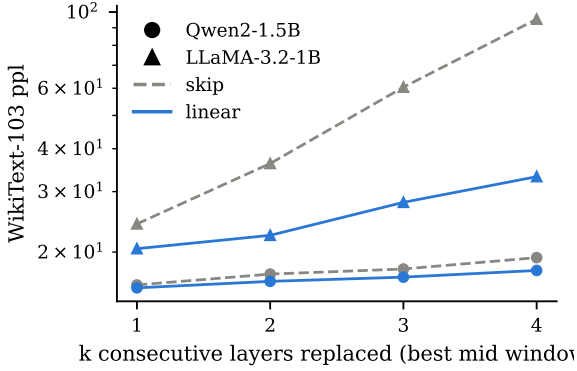


Figure 3: Replacing k consecutive layers (best-placed mid-stack window). Even four consecutive layers are recoverable; the linear map (solid) stays far below skipping (dashed) as k grows.

input-conditional, locally linear transformations. On LLaMA, by contrast, the mixture merely ties the single map everywhere—and this is what the profile predicts: LLaMA’s boundary updates are already almost perfectly linear ($\Delta R^2 = 0.996$ at layer 1), leaving no residual for piecewise structure to capture. The mixture’s gains appear exactly where, and only where, the single map’s ΔR^2 leaves room.

Two further points. Reusing an adjacent layer’s map fails catastrophically at the boundaries of *both* models (perplexity 10^4 – 10^6): the maps are genuinely layer-specific, justifying per-layer fitting. And the *unrecoverable* branch of the policy softens to *unrecoverable by a single map*: where a boundary layer’s update is only partly linear, a small

routed mixture extends the repairable set at $k \times$ the storage.

5.4 Task-level evaluation

Perplexity need not track task behavior, so we evaluate four zero-shot benchmarks via the LM Evaluation Harness (Gao et al., 2021)—ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), and PIQA (Bisk et al., 2019)—under INTACT, SKIP, and LINEAR (Table 2).³ Margin-based multiple-choice tasks (ARC-Easy, HellaSwag, PIQA) are within noise of intact under all conditions. LAMBADA, which requires exact next-word prediction and is the closest analogue of perplexity, is discriminative: substituting a mid-stack pair in LLaMA recovers accuracy from 0.132 (skip) to 0.398 (linear) against 0.626 intact, and its best single layer recovers $0.388 \rightarrow 0.568$. One negative case is instructive: at Qwen2’s layer 25, the late-stack layer with the lowest ΔR^2 , the patch scores below skipping on LAMBADA (0.534 vs. 0.590)—where the map captures little of the update, it can mislead prediction-sensitive behaviour, exactly as the policy anticipates.

5.5 The policy transfers to encoders

We repeat the single- and multi-layer experiments on BERT-base and RoBERTa-base, using masked-token accuracy in place of perplexity (intact ≈ 0.55 and 0.59 respectively). The results validate not

³500 examples per task; binomial standard error ≈ 2.2 points, so differences below ~ 4 –5 points are within noise.

Model	Condition	ARC-E	HSwag	LAMB.	PIQA
Qwen2	intact	.630	.476	.648	.752
	best skip	.666	.468	.590	.742
	best linear	.684	.468	.602	.754
	worst skip	.626	.448	.590	.744
	worst linear	.626	.458	.534	.742
	pair skip	.638	.466	.552	.750
	pair linear	.674	.458	.584	.754
LLaMA	intact	.642	.448	.626	.758
	best skip	.586	.424	.388	.742
	best linear	.652	.446	.568	.754
	worst skip	.546	.414	.404	.696
	worst linear	.580	.426	.500	.708
	pair skip	.472	.376	.132	.662
	pair linear	.506	.390	.398	.696

Table 2: Zero-shot accuracy under layer substitution (500 examples/task, ± 2.2 pt binomial SE). Multiple-choice tasks (ARC-Easy, HellaSwag, PIQA) are robust to intervention; LAMBADA, which requires exact next-word prediction, is discriminative, and the linear map consistently recovers accuracy that skipping loses. “best”/“worst” are the mid-stack layers with the highest/lowest $\Delta-R^2$; “pair” replaces two adjacent mid-stack layers.

merely the linear map but the policy’s *branch assignments*.

Where the policy prescribes patching, patching wins decisively. RoBERTa’s layer 0 is strongly load-bearing (skipping it collapses accuracy to 0.101) with a linear update: the patch restores accuracy to 0.571, and in multi-layer windows containing it the patch turns total failure into partial function ($0.025 \rightarrow 0.387$ at $k=2$; $0.006 \rightarrow 0.196$ at $k=3$).

Where the policy prescribes skipping, forcing a patch can hurt. BERT’s layer 0, unlike RoBERTa’s, is barely load-bearing (skip 0.523 vs. intact 0.55), and its mid-stack layers exhibit the skip-branch profile (layers 5–8: identity $R^2 = 0.795$ – 0.812 , $\Delta-R^2 = 0.29$ – 0.37); composing patches across these windows underperforms skipping itself (0.286 for the patch vs. 0.386 for skipping, four layers from layer 5, and at seven of nine four-layer windows). The same asymmetry appears within RoBERTa at its deepest low- $\Delta-R^2$ windows. Single-layer substitution remains within noise of intact throughout both models’ mid-stacks, confirming those layers are substitutable by omission alone.

The contrast between the two encoders is itself instructive: their layer-0 roles differ, and the policy handles both without special-casing because it reads each layer’s measured profile rather than assuming a fixed role for position. Across all four

models, the pattern is consistent: the fitted map is a rescue mechanism for large-but-linear layers, not a universal improvement, and applying it where the profile does not call for it is the one reliable way to do worse than doing nothing.

6 The Practical Envelope

The maps are effectively full-rank. Truncating W by SVD degrades the model sharply: at rank 512 (a third of d in Qwen2, a quarter in LLaMA) perplexity is already 35–269, and at rank 128, $\Delta-R^2$ falls below 0.4 with perplexity above 10^3 . Whatever linear structure the layers possess is spread across nearly all directions of the representation space. In absolute terms the constraint is mild: a full map is 4.7MB (Qwen2) or 8.4MB (LLaMA) in fp16, negligible against the weights it replaces.

Nonlinear substitutes do not help at matched budget. A two-layer GELU MLP trained on the same pairs does not exceed the closed-form ridge solution at any probed depth (Qwen2 layer 21: 0.895 vs. 0.940; LLaMA layer 12: 0.912 vs. 0.966). We note the budget asymmetry (the ridge solution is exact over ~ 3 M tokens; the MLP saw 10^5 tokens for 5 epochs), and claim only that a modestly-trained nonlinear substitute offers no advantage over the linear one.

Maps transfer across domains, less so at early layers. Maps fitted on OpenWebText transfer to WikiText-103 and to sequences of twice the fitting length with no qualitative change. On code the mid-stack profile is preserved but early-layer behavior is unstable and model-dependent (Qwen2’s layer-1 patch is worse than skipping on code, LLaMA’s is better), so we claim only that early-layer substitutes are less domain-portable.

7 Limitations

Our functional experiments cover two decoder models at 1–1.5B parameters; the representational profile covers six models but the substitution policy is validated on decoders (and, where available, encoders via masked-token accuracy). Multi-layer maps are fitted on intact-model activations, so multi-layer results are lower bounds relative to sequential refitting. The MLP control is budget-limited and does not bound nonlinear substitutes in general. Random-init controls cover four of six models, and the policy’s branch thresholds, while

cleanly separated in our models, are descriptive rather than calibrated; a systematic sensitivity analysis across architectures and training stages is future work. Task evaluations use 500 examples per task. Finally, our maps are token-wise and therefore cannot capture attention’s cross-token computation, part of what the mid-stack “nonlinear residue” comprises; disentangling the two is future work.

8 Conclusion

We set out to find which transformer layers can be cheaply replaced when one goes missing, and the answer turned out to depend on a measurement the usual one obscures. By asking how linearly predictable a layer’s *update* is, rather than its output, we recovered a learned U-shaped profile that holds across encoder-only, decoder-only, and encoder-decoder models alike: big near-linear transformations at the two ends of the stack, small nonlinear ones in the middle. That profile is enough to say, layer by layer, what a cheap substitute can and cannot fix. Linear Layer Repair makes the decision explicit—skip, patch, or give up—and our layer removal experiments show that these profile-based decisions predict functional outcomes. The detour through the residual control also left us with a cautionary point worth stating on its own: the standard inter-layer linearity score is reproduced by networks that have learned nothing, and any analysis leaning on it should report the controls we describe here. Two extensions are immediate: sequential refitting of multi-layer patches, and scaling the mixture repair of Section 5.3 across models and cluster counts to resolve the *unrecoverable* designation into a complexity ladder. Our pipeline is released so both halves of this work can be checked and pushed further.

References

- Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. 2023. [Eliciting latent predictions from transformers with the tuned lens](#). *Preprint*, arXiv:2303.08112.
- Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. [Piqa: Reasoning about physical commonsense in natural language](#). *Preprint*, arXiv:1911.11641.
- Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. 2021. [Modern koopman theory for dynamical systems](#). *Preprint*, arXiv:2102.12086.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. [Think you have solved question answering? try arc, the ai2 reasoning challenge](#). *Preprint*, arXiv:1803.05457.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [Bert: Pre-training of deep bidirectional transformers for language understanding](#). *Preprint*, arXiv:1810.04805.
- Angela Fan, Edouard Grave, and Armand Joulin. 2019. [Reducing transformer depth on demand with structured dropout](#). *Preprint*, arXiv:1909.11556.
- Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, and Niklas Muennighoff. 2021. [A framework for few-shot language model evaluation](#).
- Aaron Gokaslan and Vanya Cohen. 2019. Open-webtext corpus. <http://Skylion007.github.io/OpenWebTextCorpus>.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Gloriosi, and Daniel A. Roberts. 2024. [The unreasonable ineffectiveness of the deeper layers](#). *Preprint*, arXiv:2403.17887.
- Rudolph Emil Kalman. 1960. [A new approach to linear filtering and prediction problems](#). *Journal of Basic Engineering*, 82(1):35.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. [Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension](#). *Preprint*, arXiv:1910.13461.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [Roberta: A robustly optimized bert pretraining approach](#). *Preprint*, arXiv:1907.11692.
- Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2024. [Shortgpt: Layers in large language models are more redundant than you expect](#). *Preprint*, arXiv:2403.03853.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. [Pointer sentinel mixture models](#). *Preprint*, arXiv:1609.07843.

- Denis Paperno, Germán Kruszewski, Angeliki Lazari-
dou, Ngoc Quan Pham, Raffaella Bernardi, Sandro
Pezzele, Marco Baroni, Gemma Boleda, and Raquel
Fernández. 2016. [The LAMBADA dataset: Word
prediction requiring a broad discourse context](#). In
*Proceedings of the 54th Annual Meeting of the As-
sociation for Computational Linguistics (Volume 1:
Long Papers)*, pages 1525–1534, Berlin, Germany.
Association for Computational Linguistics.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine
Lee, Sharan Narang, Michael Matena, Yanqi Zhou,
Wei Li, and Peter J. Liu. 2020. [Exploring the limits
of transfer learning with a unified text-to-text trans-
former](#). *Preprint*, arXiv:1910.10683.
- Anton Razhigaev, Matvey Mikhalechuk, Elizaveta
Goncharova, Nikolai Gerasimenko, Ivan Oseledets,
Denis Dimitrov, and Andrey Kuznetsov. 2024.
[Your transformer is secretly linear](#). *Preprint*,
arXiv:2405.12250.
- An Yang, Baosong Yang, Binyuan Hui, Bo Zheng,
Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan
Li, Dayiheng Liu, Fei Huang, Guanting Dong, Hao-
ran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian
Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and
43 others. 2024. [Qwen2 technical report](#). *Preprint*,
arXiv:2407.10671.
- Sen Yang, Shujian Huang, Wei Zou, Jianbing Zhang,
Xinyu Dai, and Jiajun Chen. 2023. [Local interpre-
tation of transformer based on linear decomposition](#).
In *Proceedings of the 61st Annual Meeting of the
Association for Computational Linguistics (Volume 1:
Long Papers)*, pages 10270–10287, Toronto, Canada.
Association for Computational Linguistics.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali
Farhadi, and Yejin Choi. 2019. [Hellaswag: Can
a machine really finish your sentence?](#) *Preprint*,
arXiv:1905.07830.