Mechanistic Interpretability of Loop Control-Flow Generation

Anonymous Authors[1]

Abstract

Large language models are increasingly used for code generation, but the mechanisms by which they represent and produce program control flow remain poorly understood. We investigate how Gemma-2 2B predicts Python loop control-flow keywords, focusing on the syntactically related tokens `break` and `continue`. We combine Direct Logit Attribution (DLA) with activation patching to nominate and causally test attention heads involved in these predictions. For `break`, we find a localized, distributed circuit: jointly patching the top five DLA-ranked heads recovers 25.4% (95% CI $[15.3, 34.1]$) of the clean–corrupted logit-difference gap across 100 random-paired prompts. For `continue`, the analogous intervention produces only a small metric movement, driven by changes in the competing `break` logit rather than suppression of `continue` itself, with gap recovery statistically indistinguishable from zero. The DLA-ranked heads are thus causally involved but do not function as direct keyword-selecting components. Two syntactically similar control-flow predictions therefore rely on qualitatively different internal mechanisms, illustrating both the utility and the limitations of DLA-guided activation patching for circuit discovery in code-trained models.[1]

1. Introduction

Transformer-based language models (Vaswani et al., 2017) are increasingly used to generate production code (Chen et al., 2021; Jiang et al., 2024), yet the internal mechanisms by which they do so remain poorly understood, a significant gap given the practical importance of reliable code generation.

A growing body of mechanistic interpretability work has identified attention-head circuits underlying specific behaviours in language models, including indirect object identification (Wang et al., 2023), numerical comparison (Hanna et al., 2023), and induction-style in-context copying (Olsson et al., 2022). Recent work has begun to apply similar techniques to code models, including sparse-autoencoder-based analyses that identify directions associated with code correctness (Tahimic & Cheng, 2025). And head-level circuit analysis on syntactic code-completion tasks (Miller et al., 2025).

In this work, we study a narrowly defined control-flow prediction problem: how Gemma-2 2B selects between the Python loop keywords `break` and `continue` at a prediction position inside a loop body. We use Direct Logit Attribution (DLA) to nominate candidate attention heads and activation patching to test their causal role. For `break`, we find evidence of a localised but distributed contributor circuit consisting of five attention heads spanning layers 8–24, whose joint intervention recovers roughly one quarter of the clean–corrupted logit-difference gap. In contrast, applying the analogous procedure to `continue` produces a metric movement driven primarily by changes in the competing `break` logit rather than suppression of the `continue` logit, indicating that the DLA-flagged heads, although strongly aligned with the `continue` unembedding direction,do not function as `continue`-selecting components under intervention. This asymmetry between two superficially analogous control-flow keywords in the same model is the central finding of the paper.

2. Setup and Background

We conduct all experiments on Gemma-2 2B, a dense 26-layer Transformer language model developed by Google DeepMind (Gemma Team, 2024). The model has 8 attention heads per layer, giving 208 attention heads in total. We use Gemma-2 2B because it is small enough for head-level mechanistic analysis while remaining a modern code-capable language model with a documented architecture. Architectural details are summarised in Appendix A.

We study next-token prediction inside Python loop bodies, focusing on the control-flow keywords `break` and `continue`. These tokens are syntactically similar but have

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]Code: `https://anonymous.4open.science/r/loop-code-mech-interp-0015/`

different semantic effects: `break` exits the loop, while `continue` skips to the next iteration. We construct paired prompt sets per target: 100 positives where the keyword is a syntactically valid completion with substantial logit mass (though not necessarily top-1), and 50 matched-context negatives (25 no-loop, 25 alternative-control-flow) used to enforce keyword specificity in the DLA ranking. The patching analysis of Section 4 uses a logit-difference metric and does not require argmax dominance on either prompt. Table 1 summarises the setup.

Table 1. Experimental setup. Each target has 100 positive prompts and 50 negatives: 25 no-loop prompts and 25 alternative-control-flow prompts.

Target	Pos.	Neg.	Metric
`break`	100	50	$\Delta(\ell_{\texttt{break}} - \ell_{\texttt{continue}})$
`continue`	100	50	$\Delta(\ell_{\texttt{continue}} - \ell_{\texttt{break}})$

We use two sequential methods, implemented with TransformerLens (Nanda & Bloom, 2022), Hugging Face Transformers (Wolf et al., 2020), and PyTorch (Paszke et al., 2017). First, Direct Logit Attribution (DLA), a head-level extension of the logit lens (nostalgebraist, 2020; Belrose et al., 2023), ranks attention heads by how strongly their outputs project onto the target token's unembedding direction at the prediction position. This narrows the search from all 208 heads to a small set of candidates for each target token.

Second, activation patching (Meng et al., 2022; Heimersheim & Nanda, 2024) tests whether these DLA-nominated heads are causally involved. For each candidate head, we replace its output in a clean run with the corresponding activation from a corrupted run, then measure the resulting change in the target-token logit. This separates candidate discovery from causal validation: DLA identifies heads that appear important by direct projection, while patching tests whether intervening on those heads actually changes the prediction.

3. DLA Nominates Candidate Heads

Direct Logit Attribution (DLA) estimates how much each attention head directly contributes to a target-token logit. Following the residual-stream view of Transformer circuits (Elhage et al., 2021), attention heads write additive output vectors into the residual stream, and the unembedding maps the final residual representation to vocabulary logits. For a target token y, we compute the DLA score for head h in layer ℓ by taking the head's output after its output projection, selecting the vector at the prediction position, and taking a dot product with the target token's unembedding direction:

$$\mathrm{DLA}_{\ell,h}(y) = \left(z_{\ell,h} W^{O}_{\ell,h}\right)^{\top} W_U[:, y].$$

Here, $z_{\ell,h} W^{O}_{\ell,h}$ is the head's residual-stream write vector and $W_U[:, y]$ is the unembedding direction for token y. Because DLA measures only direct alignment with the output logit direction, it ignores nonlinear head–head interactions and indirect contributions routed through later layers. We therefore treat DLA as a candidate-discovery method rather than as evidence of causal necessity.

For each target token, we compute this score at the final prediction position for every prompt in the corresponding clean prompt set, then average scores across prompts to obtain a single DLA ranking over all heads. To avoid surfacing heads whose alignment is generic rather than keyword-specific, we additionally compute DLA scores against a panel of alternative tokens (`continue`, `return`, `pass`, `raise`, `print`) and retain heads with high positive contribution to the target relative to the maximum across alternatives. The combination of cross-prompt consistency on positives, suppression on negatives, and specificity against alternatives defines the candidate set used in the activation-patching experiments of Section 4. Table 2 reports the highest-ranked heads under this procedure for `break` and `continue`.

Table 2. Heads selected for patching, ranked by Pos μ, excluding L18H0 (loop-control) and L0H0 (attention-sink).

Target	Heads patched
`break`	L21H6, L24H3, L17H7, L15H6, L8H4
`continue`	L23H0, L21H0, L7H4

We retain the patched set from the original five-head DLA ranking for direct comparability; the new ranking is reported in Section 3.

3.1. `break` candidate heads

For `break`, DLA nominates L21H6, L18H0, L24H3, L17H7, and L16H0, spread across middle and late layers with heterogeneous specificity against alternative tokens. Notably, L18H0 projects more strongly onto `continue` than onto `break` (specificity -4.52 against the `break` panel), and L17H7's specificity is near zero despite a strong absolute projection. This profile of multiple contributors with mixed specificity mirrors distributed circuits in prior work (Wang et al., 2023).

We do not interpret these DLA scores as causal evidence. Instead, they define the candidate set for the activation-patching experiments in Section 4. In particular, the presence of several high-scoring heads motivates testing both individual heads and joint interventions over the top-ranked set.

3.2. `continue` candidate heads

For `continue`, DLA also identifies a compact set of candidate heads, including L18H0, L23H0, L21H6, L21H0, and

L7H4. Two of these candidates: L18H0 and L21H6 also appear in the `break` ranking, and the cross-target picture is more striking than the rankings alone suggest: the same head L18H0 has Pos $\mu = +5.71 \pm 2.01$ on the `continue` unembedding direction and Pos $\mu = +2.21 \pm 0.97$ on the `break` direction across the n=100 positive sets, with specificity scores of $+4.86$ and -4.52 respectively. This pattern is inconsistent with a pure keyword-selecting interpretation and instead suggests that DLA is partly surfacing heads that encode loop-control context in general, with the same residual-stream content projecting differently onto different keyword directions at the unembedding.

As with `break`, we treat these heads as candidates rather than confirmed causal components. The overlap between the two DLA rankings, together with the cross-target sign behaviour of L18H0, sets up the key causal question tested in Section 4: whether DLA-ranked heads behave like direct keyword selectors when intervened on, or whether their causal role differs across the two control-flow tokens.

4. Activation Patching Reveals an Asymmetry

Activation patching (Meng et al., 2022; Heimersheim & Nanda, 2024; Zhang & Nanda, 2024) tests whether the DLA-nominated heads are causally involved in the target-token prediction. For each intervention, we run the clean prompt while replacing selected head outputs at the prediction position with the corresponding activations from a paired corrupted run, then measure the change in the logit-difference metric $\Delta\ell_{\text{diff}}$ between the target keyword and its main competitor (Section 3). A negative change indicates that the patched heads were supporting the target keyword over its competitor in the clean context, while a near-zero change under conditions where the metric is well-defined indicates causal involvement that fails to register as a clean keyword-selection effect. All patching numbers are aggregated across 100 random-paired (clean, corrupted) prompts (seed-fixed); we report bootstrap 95% percentile CIs throughout.

Table 3 summarises the main patching results.

4.1. `break`: a distributed contributor circuit

We pair the clean prompt `base` with a matched-context corrupted prompt `c_base`. On the logit-difference metric $\ell_{\texttt{break}} - \ell_{\texttt{continue}}$, the clean prompt scores -1.55 ($\ell_{\texttt{break}} = +41.00$, $\ell_{\texttt{continue}} = +42.55$) and the corrupted prompt scores -9.01 ($+28.10$, $+37.11$), giving a clean–corrupted gap of $+7.46$. Both prompts assign higher logit to `continue` than to `break`; the contrast that drives the patching analysis is that `base` is markedly less `continue`-biased than `c_base`, not that `base` elicits `break` as the argmax-top-1 prediction. We patch corrupted activations into the clean run at the prediction position and report the change in this difference, $\Delta\ell_{\text{diff}}$, following the protocol of Heimersheim & Nanda (2024) and the logit-difference metric advocated by Zhang & Nanda (2024).[2]

Table 3. Activation patching effects for DLA-nominated heads on the logit-difference metric, mean and bootstrap 95% CI across n=100 random-paired prompts. Negative values indicate the patch reduced the clean prompt's preference for the target keyword. Mean clean–corrupted gap: $+10.12$ $[+8.01, +12.27]$ in both directions.

Target	Intervention	$\Delta\ell_{\text{diff}}$
`break`	L21H6	-0.97 $[-1.20, -0.73]$
`break`	L24H3	-0.42 $[-0.55, -0.28]$
`break`	L17H7	-1.38 $[-1.72, -1.03]$
`break`	L15H6	$+0.18$ $[+0.05, +0.31]$
`break`	L8H4	-0.11 $[-0.18, -0.05]$
`break`	L21H6 + L24H3	-1.40 $[-1.73, -1.06]$
`break`	Top-5 heads	-2.71 $[-3.34, -2.05]$
`continue`	L23H0	-0.61 $[-0.86, -0.35]$
`continue`	Top-3 heads	-0.80 $[-1.09, -0.49]$
`break`	L18H0 (cross)	-2.40 $[-2.96, -1.81]$

Patching the five candidate heads individually reveals substantial heterogeneity. L24H3 and L17H7 produce reliable same-signed effects on the metric ($\Delta\ell_{\text{diff}} = -0.42$ and -1.38, recovering 9.9% and 13.7% of the gap); L21H6's individual effect is smaller and its 95% CI on % recovered crosses zero (5.1% $[-3.5, +10.1]$). L17H7 is the largest individual contributor, despite a near-zero DLA specificity score and a lower Pos μ than L21H6. The remaining two heads behave differently: L15H6 produces a small wrong-signed effect ($+0.18$ $[+0.05, +0.31]$), and L8H4 sits at the per-head noise floor (-0.11 $[-0.18, -0.05]$). This profile,some heads carrying the bulk of the effect, others negligible or weakly antagonistic,mirrors the distributed structure observed in IOI (Wang et al., 2023) and numerical comparison (Hanna et al., 2023).

Jointly patching all five top-DLA heads produces $\Delta\ell_{\text{diff}} = -2.71$ $[-3.34, -2.05]$, recovering 25.4% $[15.3, 34.1]$ of the clean–corrupted gap. We interpret this as evidence that the DLA ranking surfaces a genuine causal contributor set, even though only a subset of the patched heads are individually responsible for the effect. The remaining $\sim$75% is presumably carried by heads outside the top five and by indirect contributions through MLPs and downstream attention layers, neither of which we attempt to localise here. A more systematic search would extend this analysis along the lines of automated circuit discovery (Conmy et al., 2023); we leave this to future work.

Pairwise (L21H6+L24H3:-1.40) and full-set (-2.71) joint

[2] As a sanity check, patching all 208 heads jointly reproduces the corrupted-prompt logits within 0.01, confirming that the patching pipeline is wired correctly.

patches are near-additive (sums of individual effects: -1.39 and -2.70), consistent with the patched heads writing approximately independent contributions into the residual stream.

4.2. `continue`: causal involvement without selector behavior

We mirror the procedure of Section 4.1 with targets reversed: clean prompt `c_base` and corrupted prompt `base`, with the logit-difference metric flipped to $\ell_{\texttt{continue}} - \ell_{\texttt{break}}$. The clean prompt scores $+9.01$ ($\ell_{\texttt{continue}} = +37.11$, $\ell_{\texttt{break}} = +28.10$) and the corrupted prompt scores $+1.55$ ($+42.55$, $+41.00$), giving a clean–corrupted gap of $+7.46$,numerically identical to the break gap. Patching, hook locations, and the logit-difference metric are otherwise unchanged (Heimersheim & Nanda, 2024; Zhang & Nanda, 2024).

The result diverges sharply from the `break` pattern. Patching L23H0 yields $\Delta\ell_{\text{diff}} = -0.61$ $[-0.86, -0.35]$ and jointly patching the top three heads yields -0.80 $[-1.09, -0.49]$. Both effects are driven by changes in the `break` logit rather than suppression of the `continue` logit (top-3: $\Delta\ell_{\texttt{continue}} = -0.06$ $[-0.32, +0.21]$, $\Delta\ell_{\texttt{break}} = +0.74$ $[+0.62, +0.87]$), recovering 2.2% $[-22.0, +16.3]$ of the gap. Under a naive selector reading, patching corrupted activations should *decrease* the metric by a comparable fraction to the `break` case. The absence of a coherent, gap-scale intervention effect falsifies the simple selector hypothesis for the `continue` candidate set. Distinguishing among candidate mechanisms,e.g. direct suppression (McDougall et al., 2023), propagation through `hook_z` (Goldowsky-Dill et al., 2023), or self-repair (McGrath et al., 2023),would require path patching, which is out of scope.

Patching L18H0 alone in the cross-target direction (continue-context $\rightarrow$ break-context) produces $\Delta\ell_{\text{diff}} = -2.40$ $[-2.96, -1.81]$, recovering 19.8% $[5.1, 31.7]$ of the `break` gap: comparable in magnitude to the entire top-5 `break` circuit (25.4%). A single head DLA-flagged as continue-biased thus accounts for most of the apparent break-circuit effect, suggesting the patching response is mediated largely by disruption of a shared loop-control signal rather than by selective suppression of break-specific computation. Combined with the cross-target DLA profile of L18H0 reported in Section 3, this rules out a direct keyword-selector reading and is consistent with L18H0 encoding shared loop-control context.

The asymmetry between Sections 4.1 and 4.2 is the central methodological finding: an identical DLA-then-patch procedure surfaces a distributed contributor circuit for `break` but no coherent selector effect for `continue`, despite both targets having strongly DLA-ranked candidate heads.

5. Discussion and Conclusion

We applied DLA-guided activation patching to two syntactically similar control-flow targets in Gemma-2 2B and obtained qualitatively different results. The procedure surfaces a structured contributor set for `break` but no coherent intervention effect for `continue`, even though both targets have strongly DLA-ranked candidate heads. Two observations follow: one methodological, about how DLA scores relate to intervention behaviour; one substantive, about why analogous targets can respond so differently to the same pipeline.

5.1. Attribution finds where to look, not what the circuit does

Our results illustrate a known limitation of DLA: keyword-specificity scores do not predict intervention magnitude. The largest individual `break` contributor (L17H7, 13.7%) had near-zero DLA specificity, while L18H0, which DLA flagged as continue-leaning with specificity -4.52 against `break`, recovered 19.8% of the `break` gap under cross-target patching. This is consistent with prior work showing that DLA captures only direct projection onto the unembedding and can be undone by downstream erasure and indirect routing (Janiak et al., 2023; Zhang & Nanda, 2024). One practical diagnostic worth naming: cross-target patching distinguishes context-encoding heads from keyword-selectors, since a head that recovers gap on the wrong target is encoding shared context rather than selection.

5.2. Similar control-flow tokens can rely on different mechanisms

The `break` and `continue` targets respond differently to the same pipeline, but the asymmetry is in intervention behaviour rather than head encoding: loop-control heads exist for both targets and shift keyword logits when patched cross-target. Several mechanisms could explain the gap in intervention success. Self-repair, where downstream components compensate for ablated heads, has been shown to be imperfect and noisy across full training distributions (Rushing & Nanda, 2024; McGrath et al., 2023), and could mask the effect of patching continue-related heads. The clean `continue` logit is also near saturation in our prompts ($\ell_{\texttt{continue}} = 42.55$ versus $\ell_{\texttt{break}} = 41.00$ on the break-clean prompt), limiting how far intervention can move the metric. Continue selection may additionally be more distributed across many small contributors. Disambiguating among these explanations would require path patching (Goldowsky-Dill et al., 2023), which is out of scope here. Limitations include the use of a single model and a two-keyword target set; replication on other code-capable models (Miller et al., 2025) and extension via automated circuit discovery (Conmy et al., 2023) are natural next steps.

References

Belrose, N., Furman, Z., Smith, L., Halawi, D., Ostrovsky, I., McKinney, L., Biderman, S., and Steinhardt, J. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023. URL https://arxiv.org/abs/2303.08112.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL https://arxiv.org/abs/2107.03374.

Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., and Garriga-Alonso, A. Towards automated circuit discovery for mechanistic interpretability. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2304.14997.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Gemma Team. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. URL https://arxiv.org/abs/2408.00118.

Goldowsky-Dill, N., MacLeod, C., Sato, L., and Arora, A. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*, 2023. URL https://arxiv.org/abs/2304.05969.

Hanna, M., Liu, O., and Variengien, A. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.00586.

Heimersheim, S. and Nanda, N. How to use and interpret activation patching. *arXiv preprint arXiv:2404.15255*, 2024. URL https://arxiv.org/abs/2404.15255.

Janiak, J., Rager, C., Dao, J., and Lau, Y.-T. An adversarial example for direct logit attribution: Memory management in GELU-4L. *arXiv preprint arXiv:2310.07325*, 2023. URL https://arxiv.org/abs/2310.07325.

Jiang, J., Wang, F., Shen, J., Kim, S., and Kim, S. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*, 2024. URL https://arxiv.org/abs/2406.00515.

McDougall, C., Conmy, A., Rushing, C., McGrath, T., and Nanda, N. Copy suppression: Comprehensively understanding an attention head. *arXiv preprint arXiv:2310.04625*, 2023. URL https://arxiv.org/abs/2310.04625.

McGrath, T., Rahtz, M., Kramár, J., Mikulik, V., and Legg, S. The hydra effect: Emergent self-repair in language model computations. *arXiv preprint arXiv:2307.15771*, 2023. URL https://arxiv.org/abs/2307.15771.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2202.05262.

Miller, S., Rai, D., and Yao, Z. Mechanistic understanding of language models in syntactic code completion. *arXiv preprint arXiv:2502.18499*, 2025. URL https://arxiv.org/abs/2502.18499.

Nanda, N. and Bloom, J. Transformerlens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

nostalgebraist. interpreting GPT: the logit lens. LessWrong, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS 2017 Workshop on Autodiff*, 2017. URL https://openreview.net/forum?id=BJJsrmfCZ.

Rushing, C. and Nanda, N. Explorations of self-repair in language models. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.15390.

Tahimic, K. and Cheng, C. Mechanistic interpretability of code correctness in LLMs via sparse autoencoders. *arXiv preprint arXiv:2510.02917*, 2025. URL https://arxiv.org/abs/2510.02917.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.03762.

Wang, K. R., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2211.00593.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

Zhang, F. and Nanda, N. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.16042.

A. Model Details

Table 4. Gemma-2 2B architecture hyperparameters.

Hyperparameter	**Value**
Number of Layers	26
Attention Heads per Layer	8
Model Dimension	2304
Head Dimension	256
MLP Dimension	9216
Vocabulary Size	256,000
Maximum Context Window	8,192 tokens
Key-Value Heads (GQA)	4
Positional Embeddings	Rotary (RoPE)
Normalisation	RMSNorm (pre-norm)
Logit Soft-Cap	30.0
Precision	float16