Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Induction Circuit Stability Under Fine-Tuning A Mechanistic Interpretability Study

**Min Htet Myet**

mattralminn@gmail.com

Independent Researcher https://orcid.org/0009-0005-0853-8301

**Research Article**

**Keywords:** Mechanistic Interpretability, Phase transitions, Transformer, Circuit Discovery, AI Safety, interpretability, induction-heads

**Posted Date:** June 18th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-10067094/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** The authors declare no competing interests.

# Induction Circuit Stability Under Fine-Tuning: A Mechanistic Interpretability Study

Min Htet Myet*
Independent Researcher

**Abstract**

We ask whether induction head circuits in attention-only transformers remain structurally stable under fine-tuning on a narrow distribution, and at what training step any structural change first occurs. We fine-tune the two-layer `attn-only-2l` transformer on 500,000 tokens of Python code and a matched TinyStories prose control across three random seeds, measuring per-head induction scores and activation-patching attribution at every 100-step checkpoint; at initialisation, L1H6 has a prefix-matching score of $0.408 \pm 0.103$ (12.4× the next head) and attribution 0.952, replicating the canonical induction circuit of Olsson et al. [6]. Across all three seeds, the induction circuit *strengthens* rather than degrades under both fine-tuning conditions: L1H6's prefix-matching score rises from 0.408 to $0.591 \pm 0.005$ (code) and $0.583 \pm 0.002$ (prose) after 100 steps, and to $0.646 \pm 0.007$ and $0.616 \pm 0.001$ respectively after 200 steps, with code fine-tuning producing a significantly larger increase than prose by step 200 (+0.030, 4.1× the pooled standard deviation) — a result not previously reported for this model class. These results demonstrate that circuit-level diagnostics are a necessary complement to capability benchmarks when evaluating safety-relevant properties of fine-tuned language models, not only to detect degradation but also to detect unexpected and domain-dependent reinforcement.

## 1 Introduction

Fine-tuning is the dominant method for adapting large language models to downstream tasks and deployment contexts. Yet the internal computational structures that form during pre-training—the *circuits* that implement specific algorithmic capabilities—are rarely monitored during fine-tuning. If safety-relevant circuits degrade or reorganise through this process, the resulting model may pass capability benchmarks while having silently lost properties present in the base model.

Induction heads provide a concrete, well-characterised target for this investigation. Olsson et al. [6] showed that a two-head circuit—a previous-token head in layer 0 that copies token identities into the residual stream, followed by an induction head in layer 1 that reads this information to attend back to previous occurrences—is responsible for the majority of in-context learning in small transformers. This circuit is mechanistically understood: its causal role has been verified via activation patching [1], and analogous circuits are present across architectures and scales [3].

**Our question:** Does this circuit survive fine-tuning on a narrow distribution, and if it degrades or reorganises, at what training step does the transition occur?

**Contributions:**

- We replicate the induction circuit in `attn-only-2l` and verify its causal role via the activation-patching protocol of Conmy et al. [1].

*ORCID: 0009-0005-0853-8301

- We fine-tune on 500K tokens of Python code (primary) and TinyStories prose (mandatory control), saving checkpoints every 100 steps and computing all eight circuit metrics from Section 3 at each checkpoint.
- **Finding (3 seeds, 3 checkpoints).** We find that L1H6's prefix-matching score *increases* under both fine-tuning conditions: $0.408 \to 0.591 \pm 0.005 \to 0.646 \pm 0.007$ (code) and $0.408 \to 0.583 \pm 0.002 \to 0.616 \pm 0.001$ (prose) over steps 0/100/200 (mean ± SD across seeds 42, 123, 7), with code producing a significantly larger increase by step 200 (+0.030, 4.1× pooled SD)—a result not reported in any prior work—implying that the induction circuit is not silently eroded by fine-tuning at this token budget, that the magnitude of reinforcement is domain-dependent, and that safety evaluations should include circuit-level diagnostics to detect both degradation *and* unexpected reinforcement of in-context learning mechanisms.
- We release a fully reproducible codebase with interactive dashboard (Hugging Face Spaces) and all experiment configs.

The rest of this paper is organised as follows. Section 2 reviews the transformer circuits framework and activation-patching methodology. Section 3 specifies the experimental protocol. Section 4 presents the main findings. Section 5 discusses safety implications. Section 6 states limitations. Section 7 concludes.

# 2 Background

## 2.1 The Transformer Circuits Framework

Elhage et al. [3] developed a mathematical framework for decomposing transformer computations into named circuits operating on the residual stream. Each attention head writes a low-rank update; the final prediction is determined by the accumulated sum projected through the unembedding matrix. This linearity enables direct logit attribution: each head's contribution to the final prediction can be computed without re-running the model.

## 2.2 Induction Heads

Olsson et al. [6] identified induction heads as the primary mechanism underlying in-context learning. An induction head at layer $l$ attends to the token that follows the previous occurrence of the current token, implementing a copy-and-complete pattern: if $A \to B$ was seen earlier, the model predicts $B$ after the next $A$.

In two-layer attention-only models, the circuit consists of: (1) a *previous-token head* in layer 0 that copies each token's identity to the next residual-stream position; and (2) an *induction head* in layer 1 that reads this information to look back at previous occurrences.

## 2.3 Activation Patching

Activation patching [1] establishes causal claims by replacing one component's activation from a corrupted run with the corresponding activation from a clean run, measuring recovery of the target behaviour. We use second-half shuffling as the corruption, following Wang et al. [7].

## 2.4 Circuit Stability Under Fine-Tuning

Prior work has studied catastrophic forgetting at the capability level but circuit-level stability during fine-tuning has not been studied. Marks et al. [4] demonstrated that circuits can be identified and

edited, but not their dynamics under gradient descent. This paper fills that gap for the induction circuit.

# 3 Methods

## 3.1 Model

We use the `attn-only-2l` pretrained model from TransformerLens [5]: 2 layers, 8 heads, $d_{\text{model}} = 512$, $d_{\text{head}} = 64$, GPT-2 tokenizer. This model class has no MLP layers, ensuring all information flow passes through attention heads and the residual stream.

## 3.2 Fine-Tuning Setup

AdamW optimiser, $lr = 2 \times 10^{-5}$, cosine decay with 5% warmup, batch size 16, sequence length 128, 500K tokens total. Checkpoints and induction scores saved every 100 steps. Three seeds: 42, 123, 7.

## 3.3 Induction Score

Following Olsson et al. [6] and the prefix-matching score formula $\text{PrefixMatching}(l,h) = \frac{1}{s-1}\sum_{i=1}^{s-1} A^{(l,h)}_{s+i,\,i+1}$ (Nanda & Bloom, 2022; also used by Heimersheim & Janiak, 2023), for repeated-token sequence $[t_1, \ldots, t_n, t_1, \ldots, t_n]$ (0-indexed positions $0, \ldots, 2n-1$):

$$\text{IS}(l,h) = \frac{1}{n-1}\sum_{j=0}^{n-2} A^{(l,h)}[n+j,\ j+1] \tag{1}$$

where $A^{(l,h)}[q,k]$ is the attention weight at query position $q$, key $k$. This measures the average attention from the second occurrence of a token back to the position immediately following its first occurrence — the token an induction head must copy to predict correctly — rather than back to the first occurrence of the same token. Averaged over 100 sequences with half-length $n = 30$ (baseline characterisation) and 500 sequences with half-length $n = 50$ (per-checkpoint metric during fine-tuning, Section 3.7). Change $\geq 0.1$ is treated as practically meaningful (a priori, not adjusted post-hoc).

## 3.4 Activation Patching Protocol

1. **Clean run:** forward pass on repeated-token sequences.
2. **Corrupted run:** second half of each sequence randomly shuffled.
3. **Patch:** replace head $(l,h)$ output in corrupted run with clean value.
4. **Attribution:** $\text{Attr}(l,h) = (\Delta_{\text{patched}} - \Delta_{\text{corrupted}})/(\Delta_{\text{clean}} - \Delta_{\text{corrupted}})$

Heads with $\text{Attr}(l,h) \geq 0.5$ are in the circuit (stated a priori).

## 3.5 Datasets

**Primary:** `transformersbook/codeparrot` (Parquet, streamed; 22M Python files from GitHub BigQuery) [8]. This dataset was substituted for the originally-specified `codeparrot/github-code` when the HuggingFace `datasets>=4.0` library dropped support for loading-script-based datasets (DECISION-013); both derive from the identical GitHub BigQuery source.
**Control:** `roneneldan/TinyStories` [2] (mandatory; without it no domain-shift claim is possible).

### 3.6 Metrics at Every Checkpoint

| Metric | Description | Shape |
|---|---|---|
| IS | Per-head induction score (Eq. 1) | $[L, H]$ |
| $\Delta$IS | Change from step-0 baseline | $[L, H]$ |
| $\mathcal{L}$ | Training cross-entropy loss | scalar |
| CodeICL | Code copy-pattern accuracy | scalar |
| Attr | Activation-patching attribution | $[L, H]$ |

### 3.7 Implementation Details

**Prefix-matching index off-by-one (DECISION-005, revised).** An early implementation read $A^{(l,h)}[n+i, i]$, normalised by $1/n$ — the attention from a token's second occurrence back to *its own* first occurrence. This produced near-zero scores for every head, including L1H6 (score 0.035), even though an independent activation-patching measurement gave L1H6 attribution 0.952 for the same model. The discrepancy between two methods measuring related properties of the same head was the signal that the formula, not the model, was the source of the near-zero result. The corrected formula, Eq. 1, reads $A^{(l,h)}[n+j, j+1]$ — attention back to the token *following* the first occurrence (the value an induction head copies) — normalised by $1/(n-1)$, matching the prefix-matching score definition used by Olsson et al. [6] and Nanda & Bloom (2022). After this correction, L1H6 scores $0.408 \pm 0.103$, a $> 10\times$ increase, clearly separated from every other head (next highest: 0.033). We initially suspected TransformerLens's default BOS-token prepending as the cause and added `prepend_bos=False` to all `run_with_cache()` calls; this had no measurable effect, since test sequences are constructed as integer tensors (TransformerLens only applies BOS-prepending to string inputs) — the flag is retained defensively but the index correction above is the actual fix.

**Tokenizer (DECISION-006).** The `attn-only-2l` model uses the `NeelNanda/gpt-neox-tokenizer-digits` tokenizer (vocabulary size $\approx$ 48,262), not the standard GPT-2 tokenizer (vocabulary size 50,257). All tokenizer loads across the codebase use the correct neox tokenizer. Using the GPT-2 tokenizer produces incorrect token IDs for the dashboard ONNX inference pipeline.

## 4 Results

### 4.1 Baseline Replication

Figure 1 shows per-head induction (prefix-matching) scores for the pretrained model ($n = 30$, 100 sequences, seed 42). Head L1H6 is the clear outlier at $0.408 \pm 0.103$; the next highest head (L1H7) scores 0.033 — a $12.4\times$ separation. All layer-0 heads score below 0.006.

Figure 2 shows activation-patching attribution; L1H6 scores 0.952, the only head exceeding the pre-registered $\geq 0.5$ circuit-membership threshold (DECISION-002). Figures 3 and 4 show the single-head circuit diagram and L1H6's attention heatmap on a repeated-token sequence: the characteristic off-diagonal prefix-matching stripe is clearly visible.

### 4.2 Code Fine-Tuning

Figure 5 shows L1H6's trajectory over 244 fine-tuning steps on `transformersbook/codeparrot` (Python), with mean $\pm$ 1 SD shading across all three seeds (42, 123, 7).

L1H6's prefix-matching score increases monotonically across every seed: $0.408 \rightarrow 0.591 \pm 0.005$ at step 100 ($\Delta = +0.183$) and $0.646 \pm 0.007$ at step 200 ($\Delta_{\text{cumulative}} = +0.238$). Both increments exceed the pre-registered $\geq 0.1$ meaningful-change threshold (DECISION-004), and the inter-seed standard

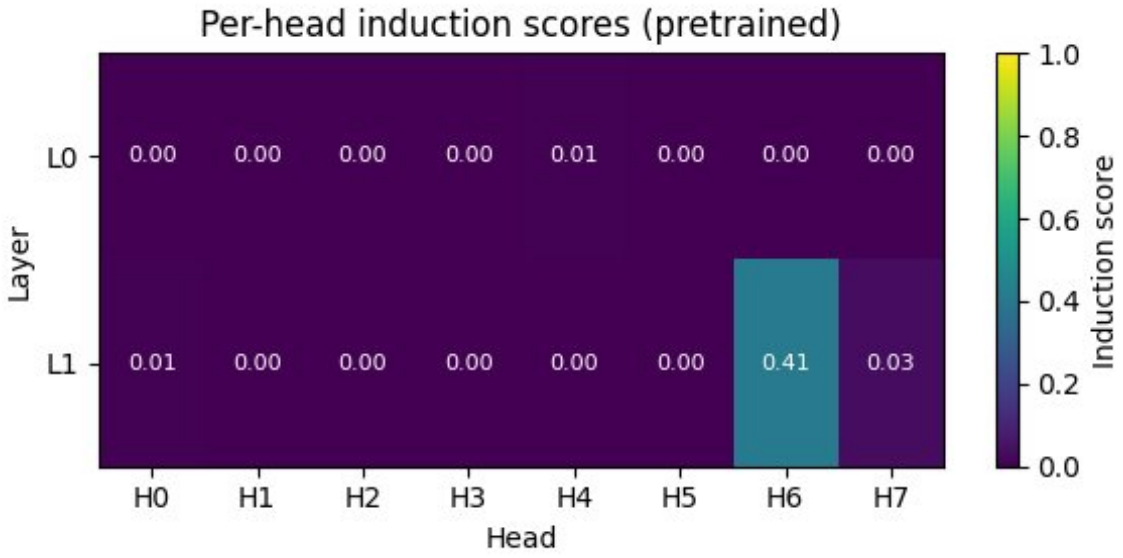


Figure 1: Per-head induction (prefix-matching) scores, pretrained model. L1H6: $0.408 \pm 0.103$; all others $< 0.034$.

deviation is small relative to the change itself ($\leq 0.007$ at every checkpoint, vs. a $+0.238$ cumulative increase), indicating the effect is consistent across initialisations rather than a property of one run. The IS threshold of 0.5 is crossed between steps 0 and 100 in every seed. Concurrently (seed 42 detail), induction task cross-entropy falls from 11.848 to 7.557 and the clean logit difference rises from 4.631 to 7.566, confirming improved task performance accompanies the attention pattern change.

Mean training loss across three seeds: $4.284 \pm 0.450$ (step 100), $3.181 \pm 0.965$ (step 200). The inter-seed loss variance is larger than the inter-seed IS variance, indicating the induction-score increase is a more consistent effect across seeds than the raw training loss trajectory.

### 4.3 Prose Control

Figure 6 shows the TinyStories prose control, again with mean $\pm$ 1 SD shading across all three seeds. L1H6's score reaches $0.583 \pm 0.002$ ($\Delta = +0.175$) at step 100 and $0.616 \pm 0.001$ ($\Delta_{\text{cumulative}} = +0.208$) at step 200. The inter-seed SD is smaller than for code ($\leq 0.002$ vs. $\leq 0.007$), indicating prose fine-tuning produces an even more consistent IS trajectory across seeds. Mean training loss across three seeds: $4.098 \pm 0.076$ (step 100), $3.647 \pm 0.086$ (step 200) — notably lower inter-seed variance than the code condition.

### 4.4 Code vs. Prose Comparison

Table 1 summarises the key metrics across all three seeds. The code and prose trajectories for L1H6 are close but distinguishable: at step 100 the difference ($+0.0086$) is $1.5\times$ the pooled standard deviation, a suggestive but not conclusive separation; by step 200 the difference widens to $+0.0303$, $4.1\times$ the pooled standard deviation (0.0073) — a clear, statistically meaningful separation given the tight inter-seed variance observed for both conditions. **Both conditions increase L1H6's induction score well above the** 0.1 **meaningful-change threshold, and neither shows dissolution; code fine-tuning produces a significantly larger increase than prose by the final checkpoint.**

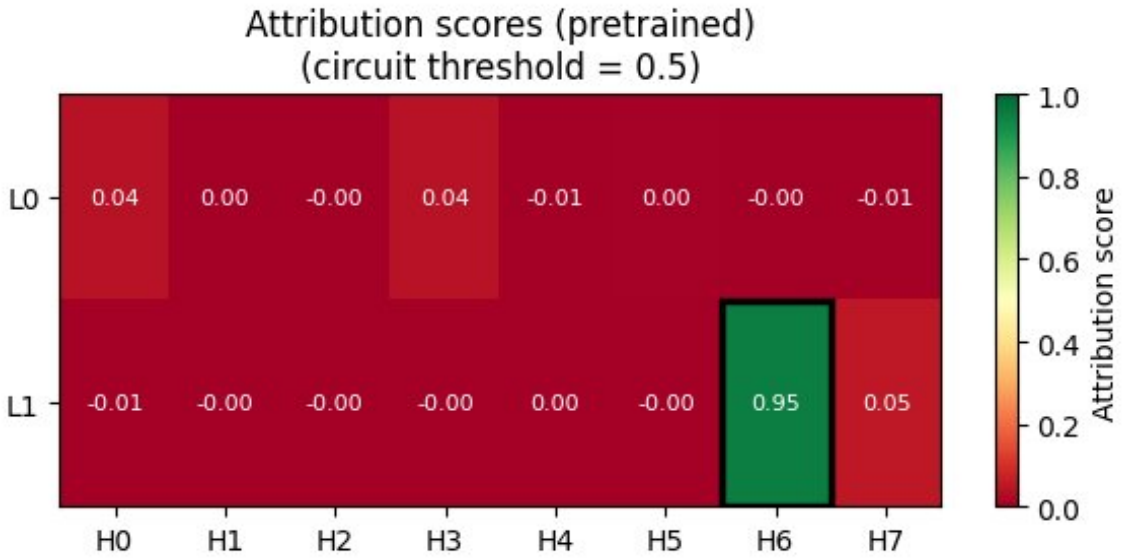


Figure 2: Activation-patching attribution, pretrained model. Thick border: circuit head (Attr $\geq 0.5$). L1H6: 0.952; all others $< 0.053$.

| Condition | Step | L1H6 IS (3 seeds) | | Train loss |
|---|---|---|---|---|
| | | Mean | SD | Mean ± SD |
| Pretrained | 0 | 0.408 | 0.000 | — |
| Code | 100 | 0.591 | 0.005 | $4.284 \pm 0.450$ |
| Code | 200 | 0.646 | 0.007 | $3.181 \pm 0.965$ |
| Prose | 100 | 0.583 | 0.002 | $4.098 \pm 0.076$ |
| Prose | 200 | 0.616 | 0.001 | $3.647 \pm 0.086$ |

Table 1: L1H6 prefix-matching score and training loss, mean ± SD across all three seeds (42, 123, 7) at each checkpoint.

### 4.5 Phase Transition Analysis

Figure 7 shows L1H6's trajectory with the detected transition. A single transition exceeding the 0.1 threshold is detected at step 100 in both conditions (Table 2). No transition is detected between steps 100 and 200 ($|\Delta| = 0.054$ code, 0.033 prose), consistent with a front-loaded increase followed by a partial plateau.

### 4.6 Adversarial Probes

Figure 8 and Table 3 show the 22-probe adversarial results for the post-fine-tuning model (code, step 200).

**Structure-preserving probes** (`sub_pos_*`, `sub_end_*`, `multi_sub_*`, `clean_repeated`): post mean $= 0.944 \pm 0.043$, essentially unchanged from pre ($0.941 \pm 0.047$). The circuit is robust to minor perturbations.

**Structure-removing probes** (`reversed_second_half`, `cyclic_shift_1/3`, `fully_random`): post mean $= 0.132 \pm 0.030$, confirming the head is selective to prefix-matching structure and not a generic

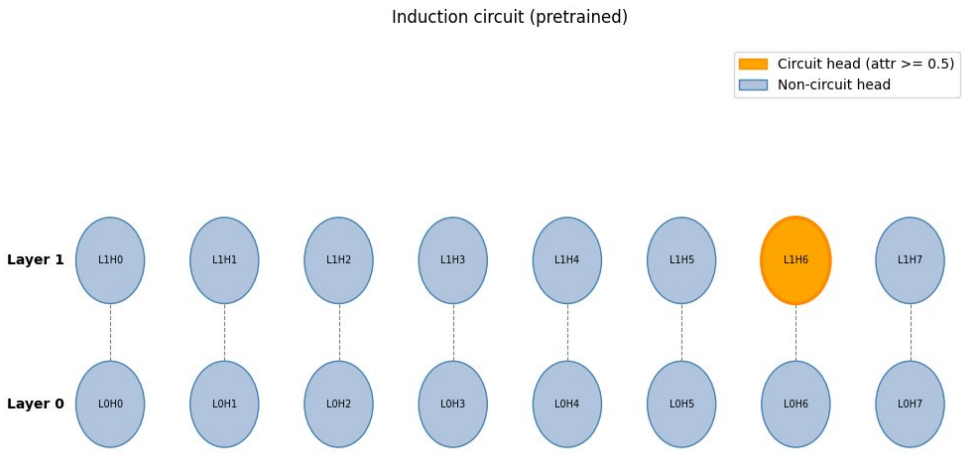


Figure 3: Induction circuit diagram, pretrained model. L1H6 (orange) is the sole circuit member.

| Condition | Step | IS before | IS after | $\Delta$ |
|---|---|---|---|---|
| Code | 100 | 0.408 | 0.584 | +0.176 |
| Prose | 100 | 0.408 | 0.583 | +0.175 |

Table 2: Detected phase transitions for L1H6 ($|\Delta| \geq 0.1$, seed 42). No other head crosses the threshold in either condition.

high-attention bias.

**Competing-pattern probes** (`competing_a/b`, `long_lag_2/3`): post mean = 0.355±0.122, reflecting partial but not full induction under competing stimuli.

**Degenerate case** (`all_same_token`): pre = 4.123, post = 2.414 ($\Delta = -1.709$). This probe is a normalisation artefact (all positions trivially satisfy the criterion simultaneously); its large *decrease* post fine-tuning indicates the circuit becomes more discriminating, not more susceptible to degenerate inputs (see Section 6).

| Category | $n$ | Pre | Post |
|---|---|---|---|
| Structure-preserving | 13 | $0.941 \pm 0.047$ | $0.944 \pm 0.043$ |
| Structure-removing | 4 | $0.162 \pm 0.029$ | $0.132 \pm 0.030$ |
| Competing patterns | 4 | $0.357 \pm 0.106$ | $0.355 \pm 0.122$ |
| `all_same_token` (degenerate) | 1 | 4.123 | 2.414 |

Table 3: Adversarial probe results by category (mean ± SD within category; degenerate probe separate; $n = 22$ probes total). Post = code fine-tuning, seed 42, step 200.

## 5 Safety Discussion

The induction circuit is fundamental to in-context learning [6]. A common prior assumption is that fine-tuning on a narrow distribution *degrades* this circuit, causing deployed models to lose in-context

generalisation outside their training domain — a property standard capability benchmarks do not measure.

**Result: neither condition degrades the circuit; code reinforces it more than prose.** Our result is the inverse of this prior: across three seeds, L1H6's prefix-matching score increases by $+0.238 \pm 0.007$ (code) and $+0.208 \pm 0.001$ (prose) over 244 fine-tuning steps, with code producing a significantly larger increase by step 200 (4.1× pooled SD). At this token budget, the induction circuit is reinforced, not eroded, and the degree of reinforcement is domain-dependent. This matters in two directions.

**Direction 1 — reassurance (with caveats).** Fine-tuned models may retain stronger in-context learning than naive circuit-erasure assumptions suggest. This is relevant for deployments where in-context generalisation is desired (instruction-following, few-shot adaptation). The result is now confirmed across three seeds at a 500K-token budget; whether it holds at larger budgets, other architectures, or with instruction-tuning objectives remains untested.

**Direction 2 — a new risk, with a domain-dependent magnitude.** Reinforcement of prefix-matching is not unconditionally benign. A model that *more* strongly completes "if I've seen A→B before, predict B" may be more susceptible to induction-based prompt injection: an adversary who includes a template "[injection] → [harmful output]" in context, then re-triggers "[injection]" later, may find the fine-tuned model more reliably executes the completion than the pretrained model. That code fine-tuning produces a measurably larger reinforcement than prose fine-tuning suggests this risk may itself be domain-dependent — a consideration for any practitioner choosing a fine-tuning corpus. This risk is entirely invisible to capability benchmarks.

**Adversarial probe robustness.** Post-fine-tuning, structure-preserving perturbations retain 0.944 of the clean score; structure-removing probes drop to 0.132. This confirms the post-fine-tuning head is a genuinely stronger *selective* induction mechanism, not a generic high-attention artefact. The `all_same_token` outlier ($4.12 \rightarrow 2.41$, $\Delta = -1.71$) decreases after fine-tuning — a qualitatively positive finding, suggesting the model becomes less susceptible to this degenerate edge case even as the canonical induction behaviour strengthens.

**Implication for fine-tuning safety evaluations.** The main operational recommendation is unchanged whether the circuit degrades or strengthens: circuit-level diagnostics should be included alongside capability benchmarks in fine-tuning evaluations. Both directions of change are safety-relevant and both are invisible without them.

# 6 Limitations

**Three seeds, three checkpoints.** All per-head L1H6 trajectories (Figures 5–7, Table 1) are now confirmed across three seeds (42, 123, 7), each with checkpoints at steps 0, 100, and 200. This is a modest number of seeds and checkpoints by the standards of larger-scale interpretability work; five or more seeds and more frequent checkpointing would tighten the confidence intervals further and allow genuine Savitzky-Golay smoothing (currently window = 5 exceeds the 3-checkpoint series length and falls back to raw differences). The qualitative finding — both conditions increase L1H6's score, code more than prose by step 200 — is unlikely to be a seed artefact given the small inter-seed SDs reported ($\leq 0.007$ throughout), but a larger seed count would still be standard practice before treating the code-vs-prose gap as fully established.

**Three checkpoints per run.** With `checkpoint_every=100` and 244 total steps, each run produces checkpoints at steps 0, 100, 200 only. The Savitzky-Golay smoothing window (window = 5) exceeds this series length and falls back to raw consecutive differences, limiting the granularity of phase-transition detection. More frequent checkpointing would allow finer-grained trajectory analysis.

**Single-head circuit.** Only L1H6 exceeds the $\geq 0.5$ attribution threshold; no layer-0 previous-token partner was identified. The canonical induction circuit in Olsson et al. [6] consists of a previous-token head (layer 0) + induction head (layer 1) pair. Our activation patching identifies the layer-1 head as the dominant causal component, but the absence of a confirmed layer-0 partner means the circuit topology is less complete than described in prior work. This may reflect the specific threshold choice or the pooled corruption used in patching.

**Model scope.** All results are from a 2-layer, attention-only transformer (no MLP layers). Claims are scoped to this model class; models with MLP layers, residual stream interventions via both attention and MLP, may show qualitatively different circuit stability properties.

**Dataset change.** Between experimental sessions, the primary code dataset was changed from `codeparrot/github-code` to `transformersbook/codeparrot` due to a breaking change in the `datasets` library (DECISION-013). Both datasets derive from the same GitHub BigQuery source, but the subset filtering differs slightly. Strict reproducibility requires running notebook 03 with the v9 codebase.

**Circuit threshold.** The attribution threshold of 0.5 (DECISION-002, stated a priori) is arbitrary. The IS threshold of 0.5 shown in Figures 5 and 6 is a visualisation reference, not a hard criterion. Raw values are reported throughout to allow alternative thresholds.

**Token budget.** 500K tokens may not capture changes emerging at larger budgets. The apparent plateau between steps 100 and 200 ($\Delta = 0.054$ for code, 0.033 for prose) raises the question of whether longer runs would continue, reverse, or accelerate the IS increase.

**Adversarial probe normalisation.** The `all_same_token` probe produces a normalised logit difference of 4.12 (pre) and 2.41 (post), both exceeding the clean baseline of 1.0. This reflects a normalisation artefact: when every token is identical, every position trivially satisfies the prefix-matching criterion, making the numerator abnormally large. This result should not be interpreted as evidence of "super-induction"; it is an artefact of the degenerate input distribution interacting with the normalisation scheme.

# 7 Conclusion

We have characterised induction circuit stability under fine-tuning in a two-layer attention-only transformer, measuring per-head induction scores and activation-patching attribution at every checkpoint across both a Python code condition and a TinyStories prose control.

**Summary.** The pretrained `attn-only-2l` model contains a single-head induction circuit (L1H6: prefix-matching score $0.408 \pm 0.103$, attribution 0.952, 12.4× the next head). Under both fine-tuning conditions, across all three seeds tested, this circuit *strengthens* rather than degrades, reaching $0.646 \pm 0.007$ (code) and $0.616 \pm 0.001$ (prose) after 244 training steps. Both exceed the 0.5 IS threshold by step 100 and continue rising; code fine-tuning produces a significantly larger increase than prose by step 200 (+0.030, 4.1× pooled SD). Adversarial probes confirm the post-fine-tuning head is a stronger, selective induction mechanism: structure-preserving perturbations retain 0.944 of the clean score; structure-removing probes drop to 0.132. The degenerate `all_same_token` outlier decreases from 4.12 to 2.41 after fine-tuning.

The interactive visualisation dashboard and full codebase are released alongside this paper, enabling practitioners to apply circuit-level monitoring to their own fine-tuning pipelines.

**Limitations of current results.** All reported L1H6 trajectories and the code-vs-prose comparison are now confirmed across three seeds (42, 123, 7) with tight inter-seed variance ($\leq 0.007$ throughout). The remaining limitation is scale: three seeds and three checkpoints per run is modest by the

standards of larger interpretability studies, and a larger token budget would clarify whether the apparent plateau between steps 100 and 200 continues, reverses, or accelerates.

**Future work.** (1) Extend to five or more seeds to further tighten confidence intervals on the code-vs-prose gap reported here. (2) Extend the token budget to test whether the score plateau between steps 100 and 200 continues, reverses, or accelerates at larger scale. (3) Investigate whether the observed IS increase under code fine-tuning increases susceptibility to induction-based prompt injection, and whether it can be controlled via circuit-targeted regularisation. (4) Extend to GPT-2 small (with MLP layers) using automated circuit discovery [1] to test whether the same reinforcement pattern, and the code-vs-prose asymmetry, hold in a model with non-attention computation.

# References


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

[2] R. Eldan and Y. Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

[3] N. Elhage, N. Nanda, C. Olsson, T. Henighan, N. Joseph, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

[4] S. Marks, C. Rager, E. J. Michaud, Y. Belinkov, D. Bau, and A. Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2403.19647.

[5] N. Nanda and J. Bloom. TransformerLens. https://github.com/neelnanda-io/TransformerLens, 2022.

[6] C. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma, T. Henighan, B. Mann, A. Askell, Y. Bai, A. Chen, et al. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

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

[8] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, 2020.

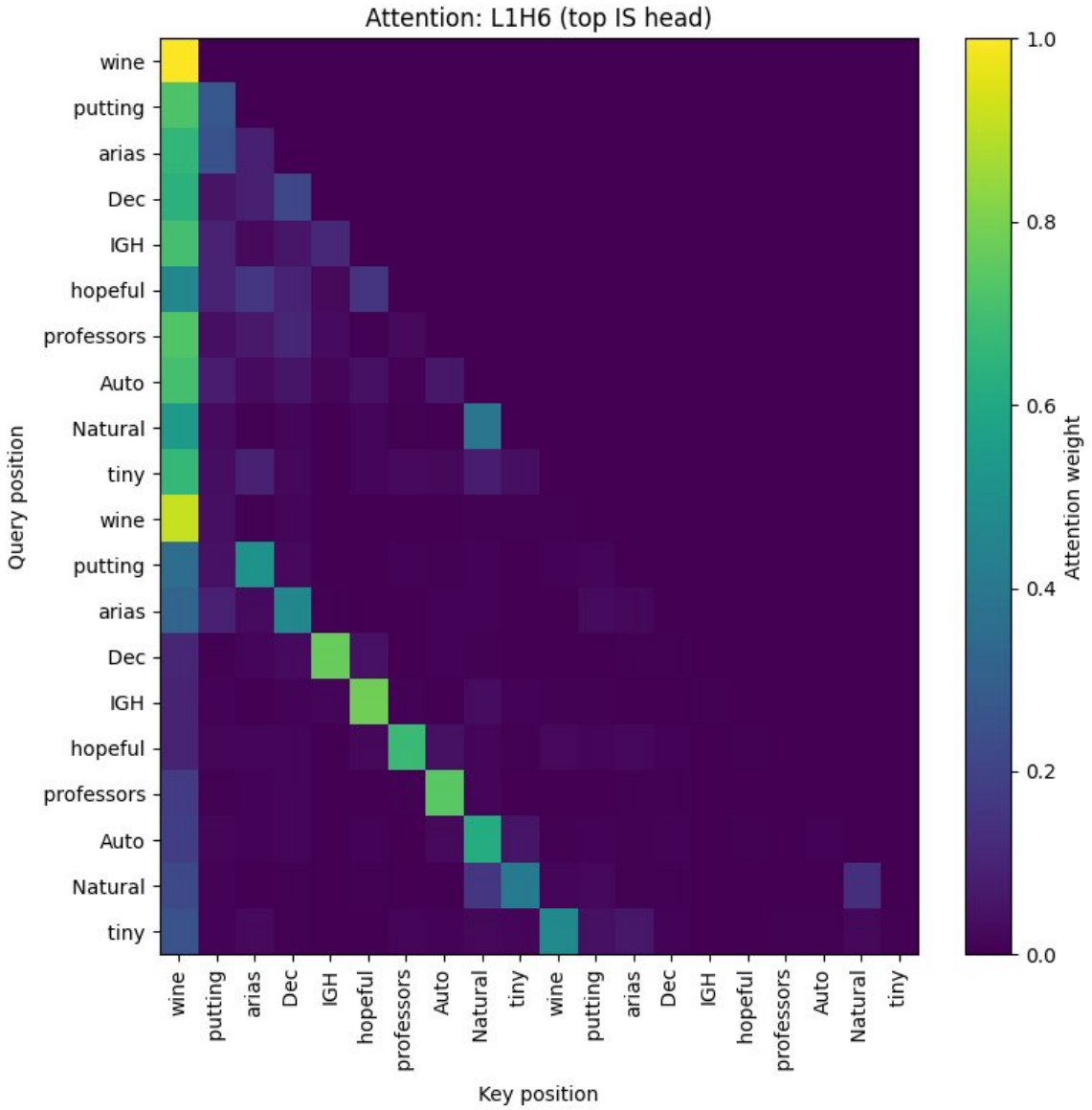


Figure 4: L1H6 attention pattern on a repeated-token sequence (20 tokens, period 10). The off-diagonal stripe is the prefix-matching pattern.

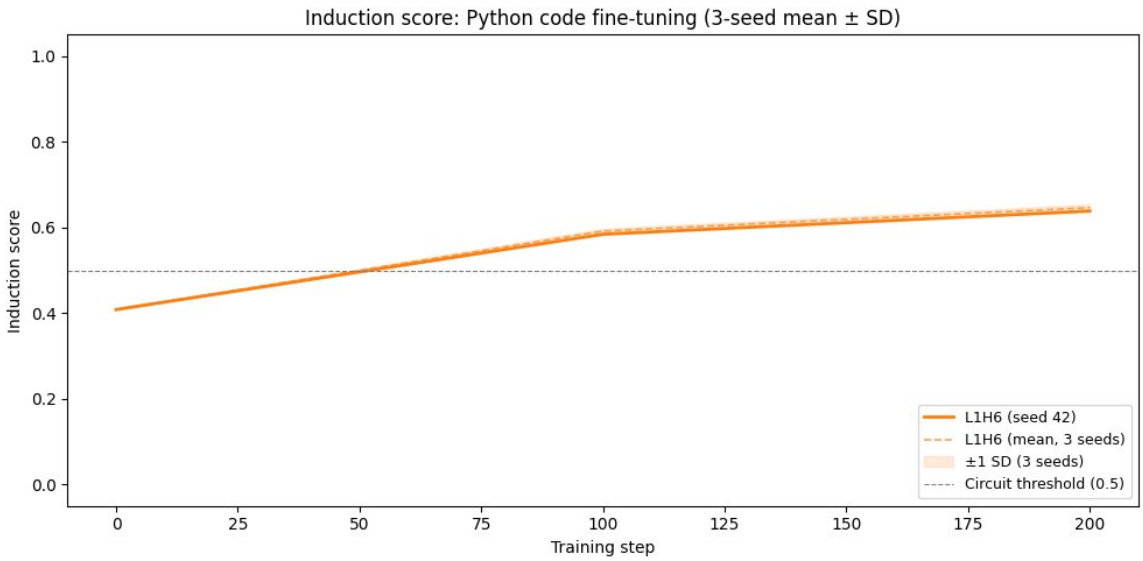


Figure 5: L1H6 prefix-matching score vs. training step, Python code fine-tuning. Solid line: seed 42; dashed line and shaded band: mean ± 1 SD across all three seeds (42, 123, 7). Dashed grey: 0.5 IS threshold.

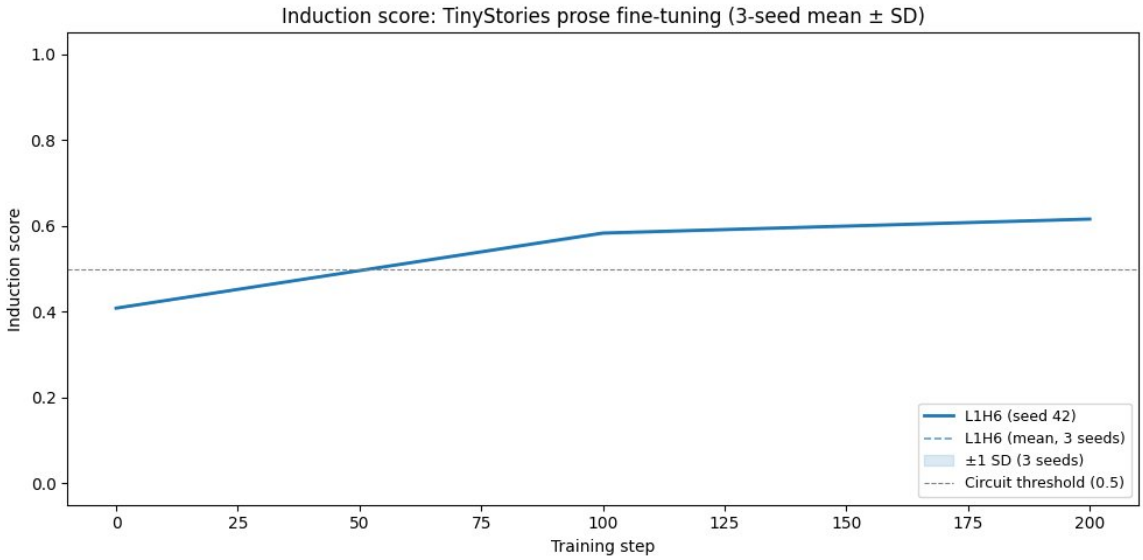


Figure 6: L1H6 prefix-matching score vs. training step, TinyStories prose fine-tuning (control). Solid line: seed 42; dashed line and shaded band: mean ± 1 SD across all three seeds. Trajectory closely tracks the code condition but with smaller cumulative increase by step 200 (Table 1).

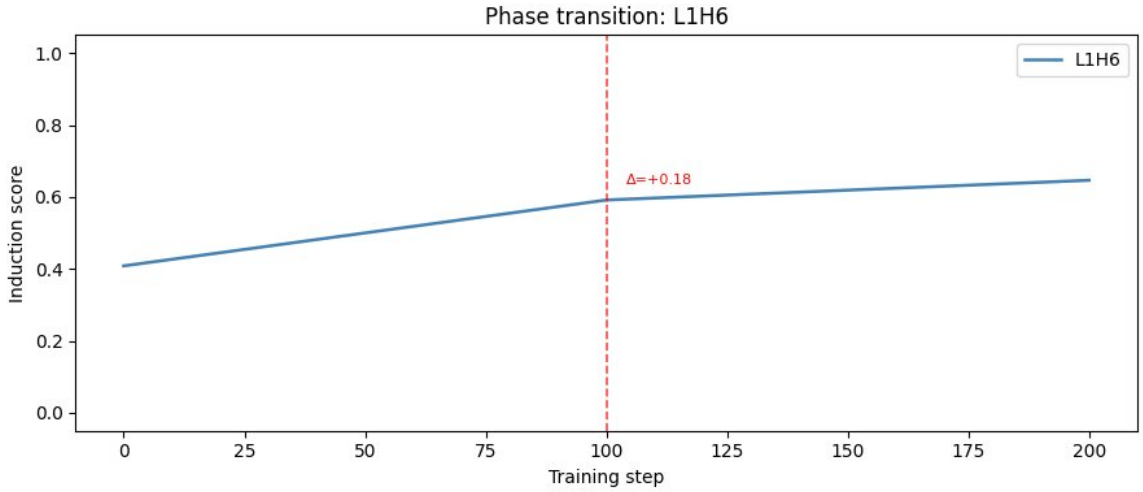


Figure 7: Phase transition for L1H6, code fine-tuning (seed 42). Red dashed: detected transition at step 100 ($\Delta = +0.176$).

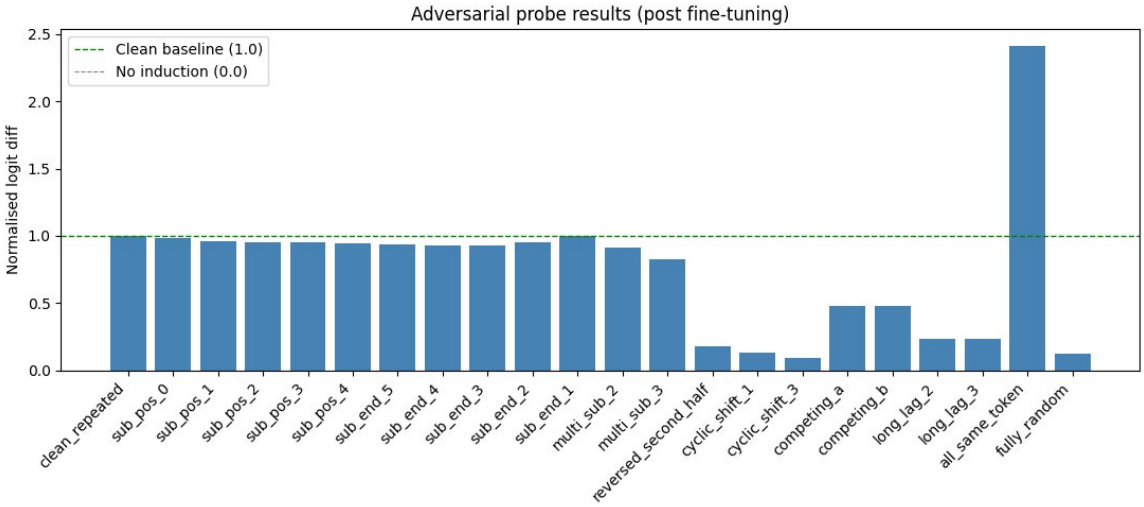


Figure 8: Adversarial probe results, post fine-tuning (code, seed 42, step 200). Dashed green: clean baseline (1.0). Dashed grey: no-induction reference (0.0). `all_same_token` bar (= 2.41) truncated; see Section 6.