

When “9.11” Stops Being a Decimal
Causal Tracing a Context-Misclassification Failure in LLM Numerical Reasoning

Sheldon Lewis
University of Waterloo
sheldon.lewis@uwaterloo.ca

MATS 10.0 - Neel Nanda Stream (Mechanistic Interpretability)

microsoft/phi-3-mini-4k-instruct

Mechanistic Interpretability Research Task (~16 hours)

Research: ~17 hours
Executive summary writing: ~2 hours

[GitHub](#)

December 23, 2025

Executive Summary

Question and Motivation

Large language models sometimes fail at seemingly simple numerical reasoning tasks, such as comparing decimal numbers (e.g., “7.11 vs 7.8”). A natural hypothesis is that these failures arise from context misclassification: the model internally routes the prompt through an inappropriate interpretation mode (e.g., treating decimal strings as generic text rather than numbers).

This project investigates two related questions:

1. Does explicitly instructing a model to treat numbers as decimals improve or degrade decimal-comparison accuracy?
2. If behavior changes under such instructions, is it mediated by a localized internal mechanism that can be detected via causal activation patching?

These questions are directly motivated by recent work in mechanistic interpretability that frames such issues as instances of model “misrouting” or mode selection, and by Neel Nanda’s emphasis on pragmatic debugging of real model behavior.

Experimental Setup

I evaluated a small but complete set of 8 decimal-comparison pairs under three prompt conditions:

- Ambiguous: “Which is larger, A or B?”
- Explicit decimal instruction: “Treat these as decimal numbers...”
- Cleaned formatting: identical to the decimal instruction, but with standardized numeric formatting (e.g., 8.30 instead of 8.3).

Evaluation was done using forced-choice log-probability scoring between the two candidate answers, avoiding reliance on free-form generation.

The primary model tested was microsoft/phi-3-mini-4k-instruct, run locally using PyTorch on macOS (MPS backend).

Key Findings

Baseline performance varied sharply by prompt condition:

- Ambiguous prompts: 87.5% accuracy (7/8)
- Explicit decimal instruction: 50% accuracy (4/8)
- Cleaned formatting: 25% accuracy (2/8)

Rather than improving performance, explicit disambiguation systematically degraded accuracy, often flipping previously correct predictions with large log-probability margins.

Accuracy by Prompt Condition

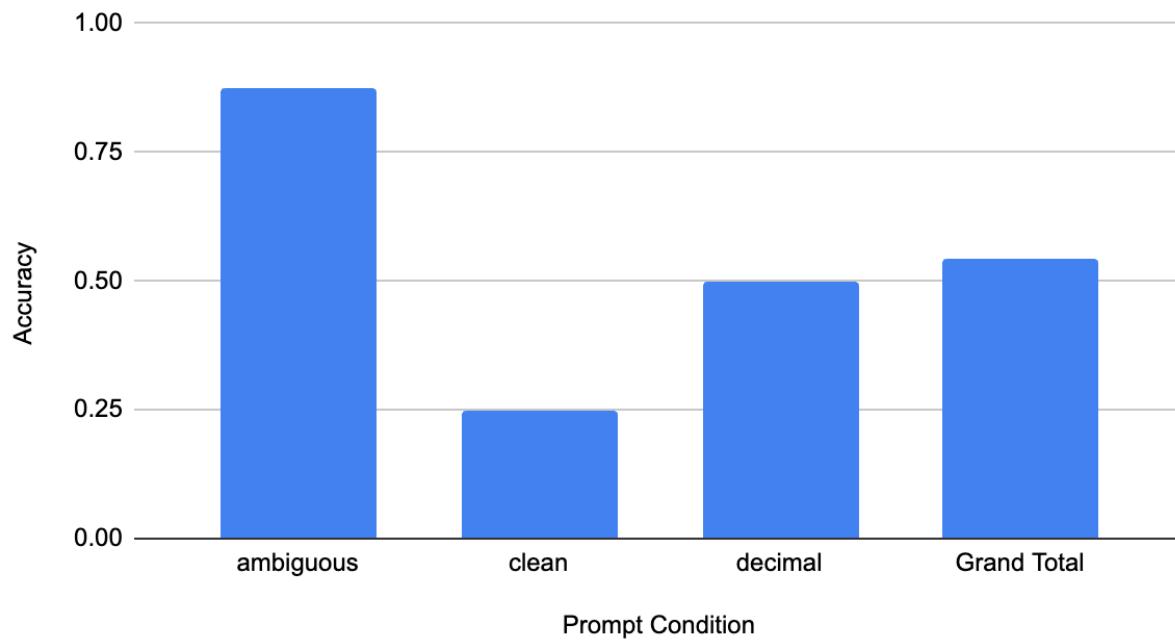


Figure 1 summarizes accuracy across the three conditions.

Mechanistic Test: Activation Patching

To test whether this instruction sensitivity reflected a simple internal “mode switch,” I used causal activation patching between ambiguous and decimal prompts.

Specifically, I patched hidden states from one prompt condition into the other (in both directions) across layers and measured the resulting probability of the correct answer.

If a localized routing mechanism existed, patching should have produced a layer band where behavior reliably changed.

However, no such effect was observed. For the representative test case, the probability of a correct answer remained at chance (0.5) across all patched layers in both directions.

Effect of Activation Patching on $p(\text{correct})$

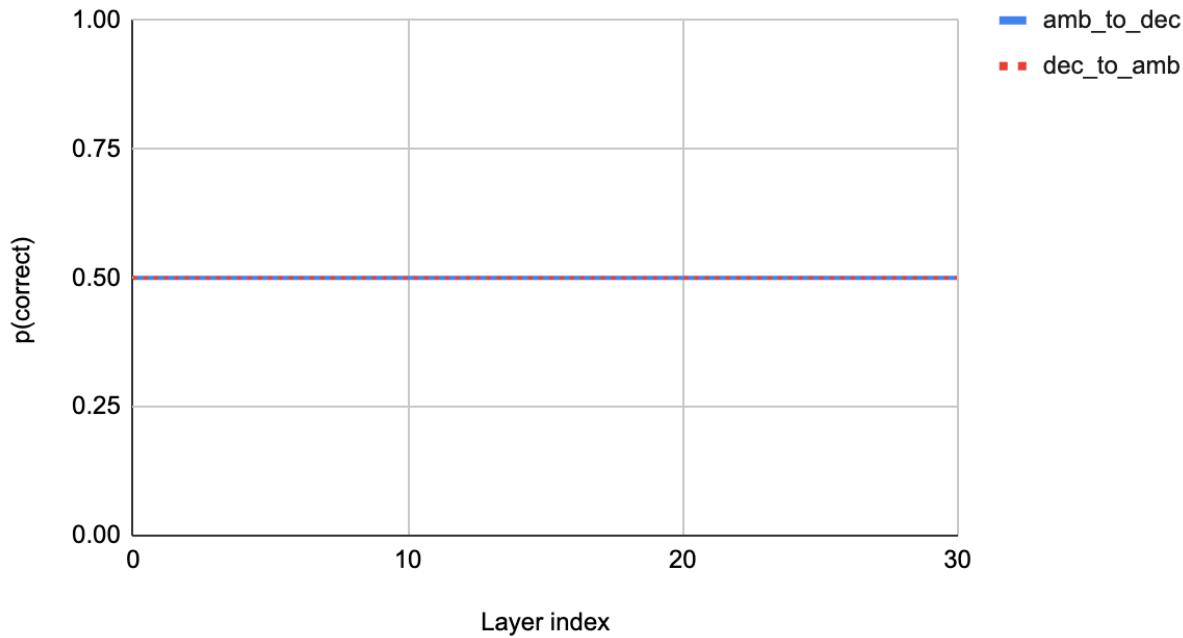


Figure 2 shows $p(\text{correct})$ as a function of layer index, which remains flat.

Interpretation and Conclusions

These results falsify a simple “context-routing switch” hypothesis for this behavior in the tested setup. While explicit instructions strongly affect performance, their effect does not appear to be mediated by a single, layer-localized mechanism that can be transplanted via activation patching.

Instead, the failure mode appears to be distributed or instruction-induced, potentially reflecting shallow heuristics or instruction-following behavior overriding a working default strategy rather than a clean internal mode change.

This negative result is informative: it constrains the hypothesis space and suggests that this class of failures may not admit a clean circuit-level explanation using standard patching techniques.

Limitations and Next Steps

The dataset is small (8 pairs), patching was coarse, and only one model was tested under local compute constraints. It is possible that the relevant mechanism lies outside the patched loci (e.g., answer token representations or attention outputs), or that stronger effects would emerge in higher-contrast cases or other models.

With more time and compute, I would scale the dataset, focus on cases where disambiguation reliably flips outcomes, and patch additional sites. Given the absence of a patching signal here, I intentionally did not pursue steering experiments, as doing so would not have been justified by the evidence.

Main Project Write-Up

1. Background and Hypotheses

Decimal comparison errors are a well-known but underexplored class of LLM failures. A plausible explanation is that such errors arise from internal context misclassification rather than lack of numerical ability.

I tested three hypotheses:

- H1: Explicit decimal instructions improve accuracy by activating a correct internal interpretation.
- H2: Explicit instructions degrade accuracy by overriding a working heuristic.
- H3: Any such change is mediated by a localized internal mechanism detectable via activation patching.

2. Methods

Model and environment.

All experiments were run on microsoft/phi-3-mini-4k-instruct using PyTorch with MPS acceleration.

Prompts and data.

Eight decimal-comparison pairs were evaluated under the three prompt conditions described above.

Evaluation.

Forced-choice log-probability scoring was used to compare the two candidate answers, recording accuracy and log-probability margins.

Activation patching.

Hidden states from ambiguous and decimal prompts were cached and patched across layers in both directions, measuring $p(\text{correct})$ after patching.

3. Results

Baseline results showed a clear monotonic decline in accuracy from ambiguous → decimal → cleaned prompts.

Activation patching produced no layer-localized effect: $p(\text{correct})$ remained at chance across all layers and both patching directions.

4. Discussion

These findings indicate that instruction-induced degradation in decimal comparison does not correspond to a simple, transplantable internal “mode.” This suggests limits to the applicability of standard patching approaches for debugging this behavior and highlights the importance of falsification-driven mechanistic work.

5. Summary

This project demonstrates a small but concrete instance where a plausible mechanistic hypothesis can be tested and falsified using causal tools. While no clean circuit was found, the results meaningfully narrow the space of explanations and illustrate the value of negative results in mechanistic interpretability.