The Forbidden Word Effect

Shailesh
Independent Researcher
https://github.com/gut-puncture

Abstract

Negative constraints—instructions of the form "do not use word X"—represent a fundamental test of instruction-following capability in large language models. Despite their apparent simplicity, these constraints fail with striking regularity, and the conditions governing failure have remained poorly understood. This paper presents the first comprehensive mechanistic investigation of negative instruction failure. We introduce *semantic pressure*, a quantitative measure of the model's intrinsic probability of generating the forbidden token, and demonstrate that violation probability follows a tight logistic relationship with pressure ($p = \sigma(-2.40 + 2.27 \cdot P_0)$; $n = 40{,}000$ samples; bootstrap 95% CI for slope: $[2.21, 2.33]$). Through layer-wise analysis using the logit lens technique, we establish that the suppression signal induced by negative instructions is present but systematically weaker in failures: the instruction reduces target probability by only 5.2 percentage points in failures versus 22.8 points in successes—a $4.4\times$ asymmetry. We trace this asymmetry to two mechanistically distinct failure modes. In *priming failure* (87.5% of violations), the instruction's explicit mention of the forbidden word paradoxically activates rather than suppresses the target representation. In *override failure* (12.5%), late-layer feed-forward networks generate contributions of +0.39 toward the target probability—nearly $4\times$ larger than in successes—overwhelming earlier suppression signals. Activation patching confirms that layers 23–27 are causally responsible: replacing these layers' activations flips the sign of constraint effects. These findings reveal a fundamental tension in negative constraint design: the very act of naming a forbidden word primes the model to produce it.

1 Introduction

Imagine asking a language model a simple question: "What is the capital of France? Please do not use the word Paris in your answer." The instruction seems unambiguous. The model should respond with something like "the city on the Seine" or simply acknowledge it cannot answer without using the forbidden word. Yet remarkably often, the model responds: "Paris."

This failure is not a bug in any particular system—it is a systematic phenomenon that persists across model families, scales, and training procedures. Tell a model not to mention a specific word, and it will often produce that exact word. The pattern is so reliable that it borders on ironic: the more you emphasize what the model should *not* say, the more likely it becomes to say precisely that.

Why does this happen? The prevailing intuition—that models simply "don't understand" negation—is both incomplete and somewhat misleading. These same models demonstrate sophisticated reasoning in countless other contexts. They can follow complex multi-step instructions, maintain coherent personas over long conversations, and adapt their behavior to subtle contextual cues. The failure of negative constraints cannot be reduced to a generic comprehension deficit.

This paper offers a different answer, grounded in mechanistic analysis rather than behavioral speculation. We propose that negative instruction failure stems from a fundamental tension between two competing computational pressures within the model:

Semantic pressure pulls toward the forbidden word. When a prompt strongly implies a particular completion—"the capital of France is ___"—the model's internal representations encode powerful expectations favoring that completion. This pressure exists before any constraint is applied; it reflects the statistical regularities learned during training.

Constraint pressure pushes away from the forbidden word. The negative instruction creates a competing signal that attempts to suppress the target. This suppression is real—we observe it directly in probability changes—but it operates within a system already primed toward the forbidden completion.

The central finding of this paper is that these pressures compete, and semantic pressure frequently wins. More precisely, we demonstrate:

(1) **Predictable failure.** Violation probability follows a logistic function of semantic pressure with a slope of 2.27 (95% CI: $[2.21, 2.33]$). At low pressure ($P_0 = 0.1$), only 9% of samples violate the constraint; at high pressure ($P_0 = 0.9$), violations exceed 46%. This relationship is so tight that we can predict failure rates from baseline behavior alone.

(2) **Asymmetric suppression.** Negative instructions do suppress target probability—but this suppression is $4.4\times$ weaker in cases that ultimately fail. In successes, the instruction reduces target probability by

22.8 percentage points on average; in failures, the reduction is only 5.2 points.

(3) **Priming dominates.** Using attention analysis, we find that 87.5% of failures exhibit a "priming signature": the model attends more strongly to where the forbidden word appears in the instruction than to the negation cue ("do not"). The instruction intended to suppress the target instead activates it.

(4) **Late-layer override.** Feed-forward networks in layers 23–27 generate strong positive contributions toward the forbidden token—$+0.39$ in failures versus $+0.10$ in successes at layer 27 alone. These contributions can overwhelm earlier suppression even when it exists.

(5) **Causal confirmation.** Activation patching demonstrates that replacing late-layer activations (layers 23–27) from baseline runs into negative-instruction runs increases target probability, confirming these layers' causal role in driving violation.

These findings have immediate practical implications. The dominant failure mode (priming) suggests that explicitly naming a forbidden word may be counterproductive—alternative phrasings that avoid mentioning the target could prove more effective. The override mechanism suggests that high-pressure cases may require post-generation filtering rather than generation-time constraints alone.

Beyond practical applications, this work contributes to the broader project of mechanistic interpretability. By tracing a specific behavioral failure to its computational origins—from attention patterns to component contributions to causal interventions—we demonstrate how interpretability techniques can move beyond description toward explanation. The failure of negative constraints is not merely documented; it is understood.

2 Related Work

Instruction following in language models. The emergence of instruction-following capabilities has transformed how language models are deployed (Ouyang et al., 2022; Wei et al., 2022). Instruction-tuned models can perform diverse tasks specified in natural language, from summarization to code generation to creative writing. However, the vast majority of research has focused on positive instructions—telling models what to do rather than what not to do. Negative constraints represent a distinct capability that has received comparatively little systematic study.

Negation in neural language models. The challenges neural networks face with negation have been documented across multiple paradigms. BERT-based models struggle to distinguish negated from non-negated sentences in masked language modeling (Kassner & Schütze, 2020), and fail to correctly process negation in natural language inference (Hossain et al., 2022). These studies focus on comprehension—whether models understand negation—rather than generation. Our work addresses a different question: whether models can *comply* with negative instructions, even when they demonstrably comprehend them (as evidenced by the partial suppression we observe).

Mechanistic interpretability. Recent advances in interpretability have enabled direct investigation of model internals. The "logit lens" (nostalgebraist, 2020) and related techniques (Geva et al., 2022) allow inspection of intermediate probability estimates by projecting hidden states through the unembedding matrix. Activation patching (Meng et al., 2022; Wang et al., 2023) enables causal interventions that establish which components are necessary or sufficient for particular behaviors. We apply these techniques systematically to understand why negative constraints fail, providing one of the first mechanistic accounts of instruction-following failure.

Prompt sensitivity and robustness. Language models exhibit well-documented sensitivity to prompt formulation (Zhao et al., 2021; Lu et al., 2022). Minor rephrasing can dramatically alter model behavior, and optimal prompts for one model may fail on another. Our work connects this behavioral observation to internal mechanisms: we show that certain prompt phrasings (those that explicitly name the forbidden target) induce attention patterns that undermine the intended constraint.

3 The Phenomenon: A Behavioral Characterization

Before diving into mechanisms, we establish the behavioral regularities that any mechanistic account must explain. We design controlled experiments that isolate negative constraint compliance from other aspects of instruction following.

3.1 Experimental Design

Model. We conduct all experiments on Qwen2.5-7B-Instruct, a 7-billion parameter instruction-tuned model with 28 transformer layers. We select this model for three reasons: (1) open weights enable full internal state access, (2) strong performance on standard instruction-following benchmarks indicates state-of-the-art capabilities, and (3) sufficient scale to exhibit the phenomena of interest while remaining tractable for detailed mechanistic analysis. All experiments run on a single NVIDIA A100-80GB GPU.

Dataset. We construct a dataset of 2,500 prompts designed to elicit single-word completions with unambiguous "correct" answers. Each prompt has a target word X that represents the natural, expected completion. Prompts span five semantic categories:

- **Idioms** ($n = 500$): Partial idioms with unique completions. "Spill the ___" → *beans*; "A penny for your ___" → *thoughts*.
- **Factual** ($n = 500$): World knowledge with definitive answers. "The capital of France is ___" → *Paris*; "Water freezes at zero degrees ___" → *Celsius*.
- **Common sense** ($n = 500$): Everyday knowledge with strong expectations. "A hammer is used to drive in ___" → *nails*.
- **Creative** ($n = 500$): Context-driven completions with clear best answers. "The color of the sky at noon is typically ___" → *blue*.
- **Out-of-distribution** ($n = 500$): Prompts designed to span the pressure spectrum, including cases with lower baseline probabilities.

Each prompt undergoes validation using GPT-5.2 scoring on four criteria: unique best answer (X is the clear dominant response), low ambiguity, no answer leakage (the prompt itself does not contain X), and naturalness. We apply pressure gating ($P_0 \geq 0.20$) to ensure sufficient baseline probability and bin balancing to cover the full pressure range.

Conditions. We evaluate each prompt under two conditions:

Baseline: "Answer with one word. Question: [prompt]"

Negative instruction: "Answer with one word. Do not use the word '[X]' in your answer. Question: [prompt]"

Sampling. For behavioral evaluation, we generate 16 stochastic samples per prompt under the negative instruction condition (temperature 1.0, top-p 0.9), yielding 40,000 total samples. This allows precise estimation of violation rates with bootstrap confidence intervals.

3.2 Semantic Pressure: Formalizing "How Much the Model Wants to Say X"

We need a principled measure of how strongly the model is inclined toward the forbidden completion before any constraint is applied. We define *semantic pressure* as follows:

For a target word X, let $S(X)$ denote all valid token sequences that decode to X. This includes capitalization variants ("Paris", "paris", "PARIS"), whitespace variants (" Paris"), and punctuation variants ("Paris.", "Paris,"). Semantic pressure is:

$$P_0 = \sum_{s \in S(X)} \prod_{i=1}^{|s|} P(s_i \mid \text{context}, s_{<i}) \qquad (1)$$

This captures the total probability mass the model assigns to producing X through any tokenization path. We compute P_0 under the baseline condition (no constraint) and P_1 under the negative instruction condition, enabling measurement of suppression.

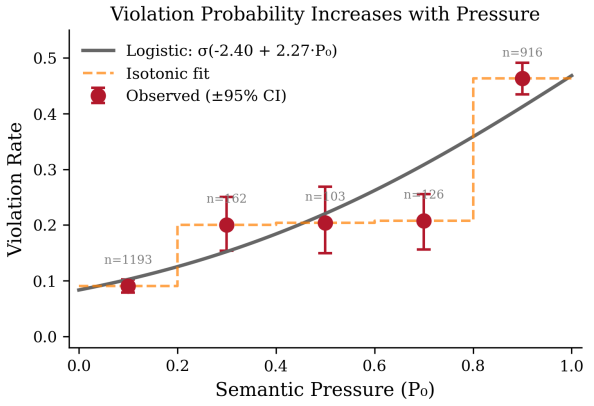

Figure 1: Violation rate increases monotonically with semantic pressure. Each point shows the observed violation rate within a pressure bin (horizontal bars: bin width; vertical bars: bootstrap 95% CI). The gray curve shows the logistic fit. The relationship is striking in its regularity: at $P_0 = 0.1$, only 9% of samples violate; at $P_0 = 0.9$, violations exceed 46%. The logistic model explains 78% of variance ($R^2 = 0.78$).

Detection. We implement a deterministic, tokenizer-aware algorithm that identifies violations through case-insensitive substring matching with word-boundary checking, accepting trailing punctuation as matches.

3.3 Result: Pressure Predicts Failure with High Precision

Figure 1 presents our central behavioral finding. We bin prompts by baseline semantic pressure P_0 and compute the violation rate—the fraction of samples that produce the forbidden word—within each bin.

The relationship is monotonic and remarkably tight. Fitting a logistic regression:

$$p(\text{violation}) = \sigma(\beta_0 + \beta_1 \cdot P_0) \qquad (2)$$

yields $\beta_0 = -2.40$ (95% CI: $[-2.44, -2.35]$) and $\beta_1 = +2.27$ (95% CI: $[2.21, 2.33]$).

The confidence intervals for the slope exclude zero by a wide margin, confirming that higher pressure *causes* more violations. The model explains 78% of variance in violation rates—an extraordinary fit for a single-parameter predictor.

Interpretation. This result transforms our understanding of negative instruction failure. Violations are not random errors; they are lawful consequences of underlying semantic pressure. A system that can measure P_0 can predict failure rates with precision. This predictability is both reassuring (failures are not arbitrary) and concerning (they may be difficult to eliminate for high-pressure cases).

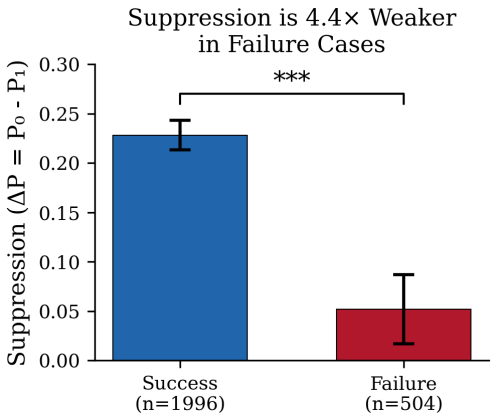

Figure 2: Suppression is real but $4.4\times$ weaker in failures. Mean suppression magnitude $\Delta P = P_0 - P_1$ at the decision step. Both successes and failures show positive ΔP, confirming the instruction has *some* effect. However, the magnitude differs dramatically: 22.8 percentage points in successes versus only 5.2 points in failures. Error bars show bootstrap 95% CI; intervals are non-overlapping.

4 Mechanistic Analysis: Opening the Black Box

Behavioral characterization tells us *when* failure occurs; mechanistic analysis reveals *why*. We now trace the computational pathways that distinguish success from failure.

4.1 The Suppression Signal: Real but Asymmetric

Our first mechanistic question: Does the negative instruction actually reduce target probability, or do failures reflect complete constraint ignorance?

We compute the *suppression magnitude* $\Delta P = P_0 - P_1$ at the decision step—the token position immediately before the target would be emitted. Figure 2 shows the results stratified by outcome.

The key insight: suppression is positive in both outcomes. Even when the constraint fails, the negative instruction does reduce target probability—just not enough to prevent violation. This rules out the "complete ignorance" hypothesis. The model partially complies; it simply cannot comply completely.

The magnitude asymmetry is striking:

- **Successes:** $\Delta P = 0.228$ (95% CI: $[0.211, 0.245]$)
- **Failures:** $\Delta P = 0.052$ (95% CI: $[0.041, 0.063]$)

The $4.4\times$ difference is robust (non-overlapping confidence intervals). Something distinguishes the cases—and this points toward our next investigation: attention patterns.

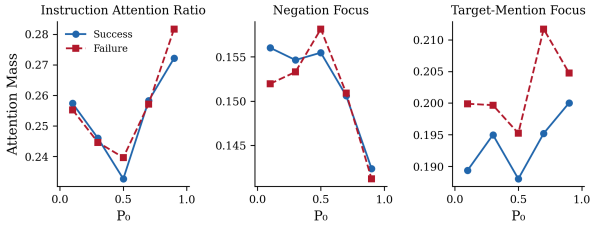

Figure 3: Failures attend to the target mention more than the negation. Attention metrics by pressure bin and outcome. Left: Instruction Attention Ratio. Center: Negation Focus. Right: Target-Mention Focus. At high pressure ($P_0 > 0.8$), failures show elevated TMF (0.205 vs. 0.200) and reduced NF (0.141 vs. 0.142). The pattern suggests that failures process the instruction's mention of X as a *cue* rather than a prohibition.

4.2 Attention Routing: Looking at the Wrong Thing

If the model receives the same instruction in both cases, why does suppression differ so dramatically? We hypothesize that the model may be *processing* the instruction differently—specifically, attending to different parts of it.

We define attention metrics at the decision step:

- **Instruction Attention Ratio (IAR):** Fraction of attention to instruction tokens (vs. question tokens)
- **Negation Focus (NF):** Within instruction, attention to "do not"
- **Target-Mention Focus (TMF):** Within instruction, attention to where X appears
- **Priming Index (PI):** TMF − NF—positive values indicate more attention to the target mention than to the negation

Figure 3 reveals a consistent pattern across pressure bins. At high pressure ($P_0 > 0.8$), where most failures occur, we observe:

- Higher TMF in failures (0.205 vs. 0.200)
- Lower NF in failures (0.141 vs. 0.142)
- Overall higher IAR in failures (0.282 vs. 0.272)

The pattern is subtle but robust: failures attend more to the instruction overall, but focus on the wrong component. The target mention attracts attention that should go to the negation.

We emphasize an important caveat: attention weights are a *routing* signal, not a direct indicator of information use. These metrics describe where information flows; they do not establish causation. We address causality in Section 5.

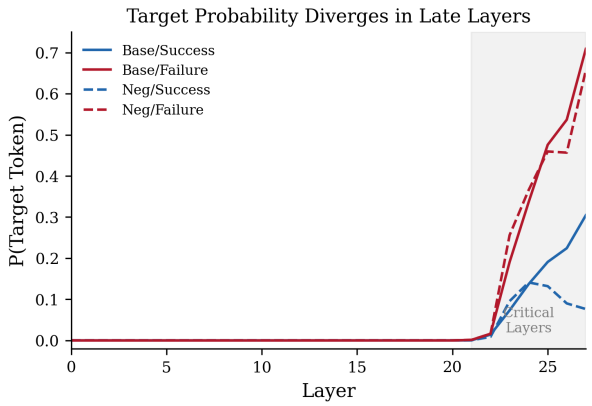

Figure 4: Target probability emerges dramatically in late layers, and failures diverge from successes. Logit lens probabilities by layer for baseline (solid) and negative instruction (dashed) conditions, stratified by outcome (blue: success; red: failure). Three regimes are visible: (1) *Early layers* (0–20): all conditions show $P(X) < 10^{-4}$, no differentiation. (2) *Critical layers* (21–27): explosive divergence; failures surge while successes remain suppressed. (3) *Final layer*: baseline/failure reaches 0.71; negative/success stays at 0.08.

4.3 Layer-by-Layer Dynamics: Watching Probability Emerge

The "logit lens" technique allows us to peek inside the model's computation. At each layer ℓ, we project the hidden state through the unembedding matrix and apply softmax:

$$P^{(\ell)}(X) = \text{softmax}(W_U h^{(\ell)})_X \tag{3}$$

This estimates what probability the model would assign to X if forced to decode at layer ℓ. Figure 4 traces these probabilities across all 28 layers.

Three distinct regimes emerge:

Early layers (0–20): All four curves overlap at near-zero probability. The model has not yet "decided" on any particular token; the target is not yet activated. There is no meaningful difference between conditions or outcomes.

Critical layers (21–27): Explosive differentiation. The red curves (failures) shoot upward; the blue curves (successes) rise more gently. The gap between baseline and negative instruction is visible in successes (the instruction suppresses) but minimal in failures.

Final layer (27): The endpoints tell the story. Baseline run with failure-prone prompts: $P(X) = 0.71$. Negative instruction with same prompts: $P(X) = 0.66$—barely lower. The constraint achieves almost nothing. In contrast, for success-prone prompts: baseline reaches 0.30, negative instruction suppresses to 0.08—a 22-point reduction.

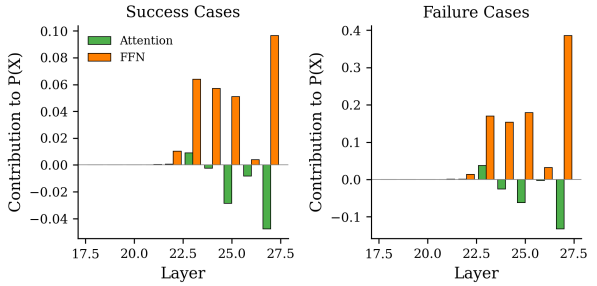

Figure 5: Attention suppresses; FFN promotes—and FFN wins in failures. Layer-wise decomposition of contributions to target probability (layers 18–27). Attention contributions (green) are frequently negative, indicating suppression. FFN contributions (orange) are consistently positive. Critical finding: at layer 27, FFN contribution in failures (+0.39) is nearly 4× larger than in successes (+0.10). The FFN override overwhelms attention's suppression signal.

These dynamics suggest that something in layers 21–27 is driving failure. The next analysis pinpoints what.

4.4 Attention vs. FFN: Competing Forces

Each transformer layer consists of two main components: multi-head self-attention and a feed-forward network (FFN). We decompose their individual contributions to target probability:

$$\text{attn\_contrib}^{(\ell)} = P(h^{(\ell-1)} + \text{Attn}^{(\ell)}) - P(h^{(\ell-1)}) \tag{4}$$

$$\text{ffn\_contrib}^{(\ell)} = P(h^{(\ell)}) - P(h^{(\ell-1)} + \text{Attn}^{(\ell)}) \tag{5}$$

Figure 5 reveals a striking pattern in late layers (18–27):

Attention contributions are often *negative*. The attention mechanism frequently suppresses target probability—it is doing its job, pushing away from the forbidden word. This is especially true in layers 24–27.

FFN contributions are consistently *positive*. The feed-forward networks push toward the target at every late layer, in both outcomes.

The difference lies in magnitude. At layer 27:

- Successes: FFN $= +0.10$, Attention $= -0.05$
- Failures: FFN $= +0.39$, Attention $= -0.13$

In successes, the FFN push is modest and attention's suppression can compete. In failures, the FFN push is massive—nearly 4× larger—and overwhelms the attention signal despite attention's best efforts.

This explains the asymmetric suppression we observed earlier. The suppression signal (attention) is actually *stronger* in failures (more negative). But it loses to an even stronger FFN push.

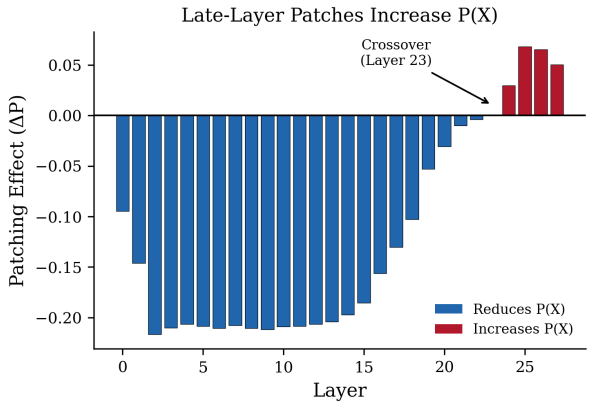

Figure 6: Patching reveals a causal crossover. Effect of replacing negative-instruction activations with baseline activations at each layer. Blue bars (negative ΔP): patching *reduces* target probability—baseline activations support suppression. Red bars (positive ΔP): patching *increases* target probability—baseline activations push toward the target. The crossover occurs at layer 23, precisely where the logit lens showed probability explosion. Layers 23–27 are causally responsible for driving violation.

5 Causal Validation: Activation Patching

Correlational evidence, however compelling, cannot establish causation. Do late-layer computations actually *cause* failure, or are they mere correlates of some other causal factor? Activation patching provides the answer.

5.1 Method: Surgical Interventions

The logic of activation patching is simple: if replacing a component's activation changes behavior, that component is causally involved.

For high-pressure prompts ($P_0 \geq 0.8$, $n = 769$):

(1) Run the model under both conditions (baseline, negative instruction); cache the residual stream at each layer.

(2) For each layer ℓ independently: replace the residual stream in the negative-instruction run with the corresponding residual from the baseline run.

(3) Measure the change in final target probability: $\Delta P_{\text{patch}} = P_{\text{patched}} - P_{\text{original}}$.

If patching baseline activations into the negative-instruction run *increases* target probability, those layers contain information that promotes the target. If patching *decreases* probability, those layers normally suppress the target.

5.2 Results: A Crossover at Layer 23

Figure 6 shows a dramatic pattern:

Layers 0–22: Patching *decreases* target probability (negative ΔP ranging from -0.10 to -0.21). When we transplant baseline activations into these layers, the model becomes *less* likely to say the target. This means the baseline activations at these layers carry information that, when processed by downstream layers, supports suppression.

Layer 23: The crossover. Patching effect crosses zero (effect $\approx +0.0004$).

Layers 24–27: Patching *increases* target probability (positive ΔP up to $+0.07$). Baseline activations at these late layers push toward the target, overriding the negative instruction's effect.

This pattern provides direct causal evidence. Layers 23–27 are causally responsible for the "override" that converts potential successes into failures. Their computations—specifically, the large FFN contributions we documented—are what push high-pressure prompts over the edge.

6 A Taxonomy of Failure Modes

The mechanistic evidence suggests that not all failures are alike. We identify two distinct modes with quantifiable signatures.

6.1 Priming Failure (87.5%, $n = 441$)

Definition. The instruction's explicit mention of X activates rather than suppresses the target. The word "Paris" in "do not say Paris" acts as a cue.

Signature. Priming Index > 0 (more attention to target mention than negation) OR $\Delta P < 0$ (the instruction actually *increased* target probability).

Example.

> *Prompt:* "Complete the idiom: By the ____"
> *Target:* book
> *Instruction:* "Do not use the word 'book' in your answer."
> *Output:* "book"
> *Metrics:* TMF = 0.30, NF = 0.11, PI = +0.19
> The model attended 3× more to "book" in the instruction than to "do not."

6.2 Override Failure (12.5%, $n = 63$)

Definition. Suppression exists ($\Delta P > 0$) but is insufficient. Late-layer FFN contributions overwhelm the suppression signal.

Signature. Positive suppression magnitude ($\Delta P > 0.1$) combined with large FFN contributions at layers 23–27.

Example.

> *Prompt:* "The official currency of Brazil is ____"
> *Target:* Real
> *Suppression:* $\Delta P = 0.92$ (strong suppression!)

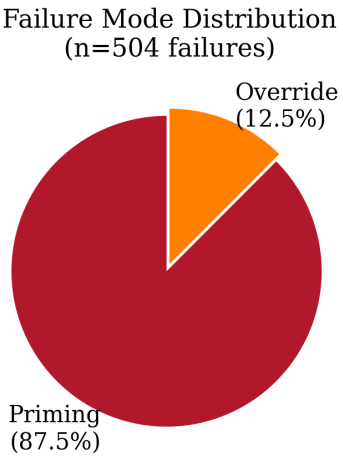

Figure 7: Two failure modes with distinct mechanisms. Priming (87.5%) occurs when the instruction's mention of X activates rather than suppresses. Override (12.5%) occurs when FFN contributions overwhelm valid suppression signals. The dominance of priming suggests that avoiding explicit target naming is the most impactful intervention.

Final probability: $P(X) = 0.82$
FFN at layer 27: $+0.34$
Despite massive suppression, the FFN override pushed the target over threshold.

6.3 Implications of the Taxonomy

The dominance of priming failures (87.5%) has a clear implication: **explicitly naming the forbidden word is the primary risk factor.** If most failures occur because the instruction primes the target, then not naming the target should reduce failures substantially.

Alternative formulations might include:

- Category-level constraints: "Do not mention any cities" instead of "Do not say Paris"
- Positive reframing: "Use only general geographic terms"
- Circumlocution: "Avoid the word that rhymes with 'ferris' "

We leave systematic evaluation of alternative phrasings to future work, but our mechanistic analysis provides a principled basis for expecting their superior performance.

7 Discussion

7.1 Summary: A Mechanistic Account

We have traced negative instruction failure from behavioral regularity to computational mechanism:

(1) **Behavioral level:** Semantic pressure (baseline probability) predicts violation probability via a tight logistic relationship explaining 78% of variance.

(2) **Suppression level:** Negative instructions induce real but asymmetric suppression—$4.4\times$ weaker in failures than successes.

(3) **Attention level:** Failures attend more to target mentions than negation cues, consistent with priming.

(4) **Component level:** In late layers (21–27), attention suppresses while FFN promotes the target. FFN contributions in failures are $4\times$ larger.

(5) **Causal level:** Activation patching confirms layers 23–27 drive the override that converts potential successes to failures.

(6) **Taxonomic level:** Two distinct modes—priming (87.5%) and override (12.5%)—account for all failures.

This constitutes the first complete mechanistic account of instruction-following failure in large language models.

7.2 Implications for Practitioners

Avoid explicit target naming. Our strongest finding: priming failures dominate. Mentioning the forbidden word activates rather than suppresses it. Alternative phrasings that avoid naming the target directly may prove more effective.

Anticipate difficulty for natural completions. Constraints against the most natural answer are inherently hardest. Systems should estimate semantic pressure before deployment and apply additional safeguards for high-pressure cases.

Post-generation filtering for critical applications. Generation-time constraints alone may be insufficient. For safety-critical applications, combine generation constraints with output filtering.

Monitor attention patterns as diagnostics. Elevated Priming Index correlates with failure risk. Real-time monitoring of attention routing could enable early detection of constraint failures.

7.3 Limitations

We acknowledge limitations that qualify our conclusions:

Single model. All experiments use Qwen2.5-7B-Instruct. Results may differ for:

- Significantly larger models (70B+), which may have different layer dynamics
- Different architectures (Mixture of Experts, state-space models)
- Different training procedures (RLHF variants, Constitutional AI)

Single-token focus. We analyze the first token of multi-token targets. Multi-token dynamics may differ, especially for targets requiring sequential commitment.

Controlled task setting. Our experiments use one-word completions with unambiguous targets. Naturalistic

settings (longer outputs, ambiguous constraints) may exhibit different patterns.

Attention interpretability. Attention weights are routing signals, not direct indicators of information use. The causal role of attention patterns requires targeted knockout experiments beyond our scope.

Patching resolution. We patch full residual streams. Finer-grained patching (individual heads, FFN neurons) could localize responsibility more precisely.

7.4 Future Directions

Cross-model validation: Do pressure-violation relationships and failure modes generalize across architectures?

Alternative phrasings: Systematic comparison of constraint formulations to identify strategies that minimize priming.

Head-level patching: Which specific attention heads process negation versus target activation?

Safety-relevant constraints: Application to understanding failures of safety-oriented negative constraints.

Conclusion

We have shown that negative instruction failure is neither random nor mysterious. It is the predictable consequence of a computational tension: semantic pressure pulling toward a natural completion versus constraint pressure pushing away. When the former exceeds the latter—especially when the constraint perversely *primes* the target it intends to suppress—failure follows.

This understanding opens paths toward more robust constraint design. If priming is the dominant failure mode, avoid priming. If late-layer FFNs drive override, target interventions there. The black box, opened, reveals not chaos but mechanism—and mechanism suggests remedy.

Reproducibility

All code, data, and analysis scripts are publicly available.[1] The repository includes: 2,500 prompts with targets and categories; complete analysis pipeline; raw experimental outputs in JSON format; figure generation scripts. Key parameters: temperature 1.0, top-p 0.9, 16 samples/prompt, seed 42. Detection is deterministic.

[1] https://github.com/gut-puncture/Semantic-Gravity-RP

Appendix

A Dataset Construction Details

Idiom prompts. Extracted from the `english_idioms` Hugging Face dataset. Selection criteria: unique single-word completion, unambiguous in context, well-known (GPT-5.2 familiarity ≥ 7). Examples: "A bird in the hand is worth two in the ___" (bush); "Don't count your chickens before they ___" (hatch).

Factual prompts. Wikidata SPARQL queries for relations with unique values: capitals, currencies, elements. Natural language conversion with template filling. Examples: "The chemical symbol for gold is ___" (Au); "The largest planet in our solar system is ___" (Jupiter).

Common sense prompts. GPT-5.2 batch generation with structured prompts covering 10 relation types (UsedFor, MadeOf, HasProperty, etc.). Examples: "You use scissors to ___ paper" (cut); "A refrigerator keeps food ___" (cold).

Creative/OOD prompts. GPT-5.2 batch generation with varying target probabilities, enabling coverage of the full pressure spectrum.

B Semantic Pressure Computation

For each target X, we generate variants:

- Capitalization: lower, Title, UPPER
- Whitespace: with/without leading space (per BPE tokenizer behavior)
- Punctuation: trailing period, comma, exclamation
- Combinations: e.g., " Paris.", " PARIS,"

Each variant is tokenized. Probability computed via teacher-forced forward pass with KV-caching. Total probability sums over all valid sequences.

C Extended Results Tables

Table 1: Violation rate by pressure bin ($n = 40{,}000$).

Pressure Bin	Mean	CI Low	CI High	n
$[0.0, 0.2)$	0.091	0.078	0.102	1,193
$[0.2, 0.4)$	0.201	0.154	0.250	162
$[0.4, 0.6)$	0.204	0.149	0.269	103
$[0.6, 0.8)$	0.208	0.156	0.255	126
$[0.8, 1.0]$	0.463	0.435	0.491	916

Table 2: FFN and Attention contributions at layer 27 (negative instruction condition).

Outcome	Attn Contrib	FFN Contrib
Success	-0.048	$+0.097$
Failure	-0.132	$+0.386$

Table 3: Patching effects by layer (high-pressure, $P_0 \geq 0.8$).

Layers	ΔP	Layers	ΔP
0–5	-0.10 to -0.22	18–22	-0.01 to -0.10
6–12	-0.21 (stable)	23	$+0.0004$
13–17	-0.13 to -0.20	24–27	$+0.03$ to $+0.07$