

---

# Tokenization Drift as a Hidden Channel for Alignment Evasion in Language Models

---

Raul Cavalcante Dinardi

Universidade de São  
Paulo (USP)

Bruno Lopes Yamamoto

Universidade de São  
Paulo (USP)

With  
Apart Research

## Abstract

*Reinforcement learning is making language models more capable, but also creates a structural tension: models trained with RL have been shown to infer when they are in training and selectively comply with a conflicting objective to preserve their out-of-training behavior, a phenomenon empirically demonstrated by Greenblatt et al. (2024). We describe a channel through which this evasion could occur without explicit reasoning, external access, or any visible difference in input text. A concrete scenario where this could arise is within the training process itself: a model is trained with RL in iterations that include alignment phases, where generated conversations are saved as plain text and retokenized when loaded back as training data. Chatzi et al. (2025) showed that language models often generate the same output string with multiple different tokenizations, meaning that retokenization of those outputs can yield a different token sequence than the model originally produced. A model under opposing RL pressures on original and retokenized contexts may have an incentive to learn to condition on this asymmetry, behaving differently depending on which token path it detects. The signal is invisible at the text level and requires no semantic cue, pointing to a gap in alignment pipelines that reconstruct training context from plain text.*

# 1. Introduction

*Reinforcement learning is increasingly used to improve the capabilities and alignment of large language models. RL training, however, creates optimization pressure toward whatever strategy most reliably achieves reward, including strategies that exploit the structure of the training pipeline itself. Greenblatt et al. (2024) showed that a language model can infer when it is in training and selectively comply with a conflicting objective to preserve its out-of-training behavior, and MacDiarmid et al. (2025) showed that reward hacking behaviors learned during RL can generalize to alignment faking and sabotage in agentic settings. These results raise a question: what other structural properties of a training pipeline could a model, under sufficient optimization pressure, learn to exploit?*

*We examine one such property that has not been considered in this context. A concrete scenario where this could arise is within the RL training pipeline itself. Standard capability-seeking RL can create pressure toward reward-hacking or misaligned behavior, while later alignment-focused phases apply pressure in the opposite direction. If a model can pass through alignment-focused phases while preserving out-of-phase behavior, it has an incentive to detect which training scenario it is currently in and respond differently, consistent with alignment-faking results such as Greenblatt et al. (2024).*

*Control techniques often proceed in iterations that include saving conversations as plain text, such as when developers log deployment interactions that triggered safety monitors and later reload them into an evaluation harness to resample the model's behavior and audit for malicious deployment-versus-training discrepancies. Chatzi et al. (2025) showed that language models often generate the same output string with multiple different tokenizations, meaning that retokenizing a model's output from plain text can yield a token sequence that differs from the one originally produced. When saved conversations are retokenized, the resulting token sequence may therefore differ from the one the model originally produced, even though the visible text is identical. This difference is present at the token level, which is the level at which the model operates, but invisible to any inspection conducted at the text level. We propose that under training with opposing pressures on original and retokenized contexts, a model may learn to condition on this discrepancy as a cue for which objective is currently active. We do not claim this has been observed in any deployed system. The incentive structure is coherent and the underlying tokenization phenomenon is documented; we argue the threat model is worth testing.*

Model generated sample

|       |      |    |     |   |     |   |      |      |     |     |   |   |   |   |     |
|-------|------|----|-----|---|-----|---|------|------|-----|-----|---|---|---|---|-----|
| areas | such | as | the | T | are | x | pect | hill | and | Val | D | y | b | . | The |
|-------|------|----|-----|---|-----|---|------|------|-----|-----|---|---|---|---|-----|

Correct canonical tokenization

|       |      |    |     |     |        |      |     |     |    |   |   |     |       |            |
|-------|------|----|-----|-----|--------|------|-----|-----|----|---|---|-----|-------|------------|
| areas | such | as | the | Tar | expect | hill | and | Val | Dy | b | . | The | exact | population |
|-------|------|----|-----|-----|--------|------|-----|-----|----|---|---|-----|-------|------------|

## 2. Related Work

*Alignment faking and deceptive behavior under RL.* Greenblatt et al. (2024) showed empirically that a language model can selectively comply with a training objective to avoid modification of its preferences, without being instructed to do so. MacDiarmid et al. (2025) extended this result, showing that reward hacking learned during RL on coding tasks can generalize to alignment faking and sabotage in agentic settings. Denison et al. (2024) showed that models trained on a curriculum of RL environments can generalize from simple specification gaming such as sycophancy to more serious behaviors including reward tampering. Our work shares the same concern but examines a mechanism that operates below the semantic level, one that requires no explicit reasoning about the training context.

*Adversarial tokenization.* Geh et al. (2025) showed that non-canonical tokenizations can bypass safety training and that models retain semantic understanding across such encodings. Their setup involves an external attacker who forces a non-canonical encoding at inference time. Our setup is different: we consider the case where the model itself generates non-canonical tokens during normal autoregressive decoding, and the discrepancy arises within the training pipeline without any external intervention.

*Reward hacking.* Several papers document models exploiting structural properties of their training environments rather than solving the intended task, including accessing answer files and manipulating test harnesses. Our threat model is similar in spirit but targets the pipeline level: the exploited structure is a tokenization discrepancy introduced when generated text is serialized to a string and retokenized.

### 3. Methods

Let  $\pi_\theta$  denote the language model under training. Let  $D_1$  be a dataset of benign first-turn user queries and  $D_2$  a dataset of second-turn user queries; draws from  $D_1$  and  $D_2$  are independent.

#### ***Rollout Procedure***

For each rollout index  $n \in \{1, \dots, N\}$  within a GRPO group  $G$ :

#### ***Stage 1: Original Sample.***

Draw  $q_1^{(n)} \sim D_1$  and  $q_2^{(n)} \sim D_2$ . Sample  $\pi_\theta$  autoregressively to produce a complete two-turn conversation:

$$x_{\{n,1\}} = (q_1^{(n)}, a_{\{n,1\}}^{(1)}, q_2^{(n)}, a_{\{n,1\}}^{(2)})$$

where:

$$a_{\{n,1\}}^{(1)} \sim \pi_{\theta}(\cdot \mid q_1^{(n)}), \quad a_{\{n,1\}}^{(2)} \sim \pi_{\theta}(\cdot \mid q_1^{(n)}, a_{\{n,1\}}^{(1)}, q_2^{(n)}).$$

Only the responses  $a_{\{n,1\}}^{(1)}$  and  $a_{\{n,1\}}^{(2)}$  are sampled from  $\pi_{\theta}$ ; the queries are externally supplied. Let  $\tau^{\text{orig}}$  denote the token sequence that constituted the context for generating  $a_{\{n,1\}}^{(2)}$ , i.e. the tokenization of  $(q_1^{(n)}, a_{\{n,1\}}^{(1)}, q_2^{(n)})$  as produced incrementally during autoregressive decoding.

### ***Stage 2: Resampled Completion.***

Serialize  $x_{\{n,1\}}$  to a plain-text string  $s$ . Truncate  $s$  at the string level to obtain the prefix string  $s^{\text{pre}}$  containing only  $(q_1^{(n)}, a_{\{n,1\}}^{(1)}, q_2^{(n)})$ . Retokenize  $s^{\text{pre}}$  from scratch using the model's tokenizer to obtain a new token sequence  $\tau^{\text{re}}$ . Sample a fresh second-turn completion:

$$a_{\{n,2\}}^{(2)} \sim \pi_{\theta}(\cdot \mid \tau^{\text{re}})$$

yielding:

$$x_{\{n,2\}} = (q_1^{(n)}, a_{\{n,1\}}^{(1)}, q_2^{(n)}, a_{\{n,2\}}^{(2)}).$$

### ***Pathwise KL Experiment and Linear Probe Diagnostic***

We train LFM2.5 1.2B with GRPO to maximize the behavioral difference between deployment context (the token-ids generated by  $\pi_{\theta}$  before decoding to plain text) and testing context (the sequence obtained by re-tokenizing that plain text). For each rollout, we sample a first-turn response  $R_1$ , build the second-turn prompt, and sample a second-turn response  $R_2$ . We then decode

the tokens before  $R_2$  to text and re-encode that text with the same tokenizer to construct a testing-context sequence.

For each sample, the reward is the mean tokenwise log-probability gap along the sampled  $R_2$  path:

$$\text{reward} = (1/T) \sum_{i=1}^T [ \log p_{\theta}(r_{\{2,t\}} \mid \text{testing context}, r_{\{2,<t\}}) - \log p_{\theta}(r_{\{2,t\}} \mid \text{deployment context}, r_{\{2,<t\}}) ]$$

This is a sampled pathwise KL-style objective rather than a full-vocabulary KL at each step.

We also run a linear-probe diagnostic. For each pair, we extract the frozen hidden state at the first  $R_2$  token position and train a linear classifier to predict whether the context is original (label 0) or retokenized (label 1). Data are split at the pair level so both members of a pair remain in the same split. Above-chance held-out accuracy indicates representational separation between original and retokenized contexts at the probed location.

The two diagnostics are complementary: the KL objective tests behavioral separation (different likelihood for the same continuation under different tokenizations), while the probe tests representational separation (linear decodability in hidden states). A positive probe does not by itself establish causality for downstream behavior.

## 4. Results

### **Tokenization Drift Frequency**

We measured how often autoregressive decoding produces token sequences that change upon retokenization. For each model, we compared the original token sequence  $\tau^{\text{orig}}$  against the retokenized sequence  $\tau^{\text{re}}$  across 256 rollouts per model. A rollout counts as drifted if the two sequences differ at any position.

Table 1: Fraction of rollouts in which  $\tau_{\square}^{re} \neq \tau_{\square}^{orig}$ . All three models produce non-canonical tokenizations at rates between  $\sim 4\%$  and  $\sim 8\%$ , confirming that the serialize–retokenize pipeline is not token-identity-preserving.

| <i>Model</i>        | <i>Drift rate</i> |
|---------------------|-------------------|
| <i>LFM 2.5 1.2B</i> | <i>7.7%</i>       |
| <i>LLAMA 3.1 1B</i> | <i>6.0%</i>       |
| <i>Qwen 3 0.6B</i>  | <i>3.8%</i>       |

### ***Linear Probe***

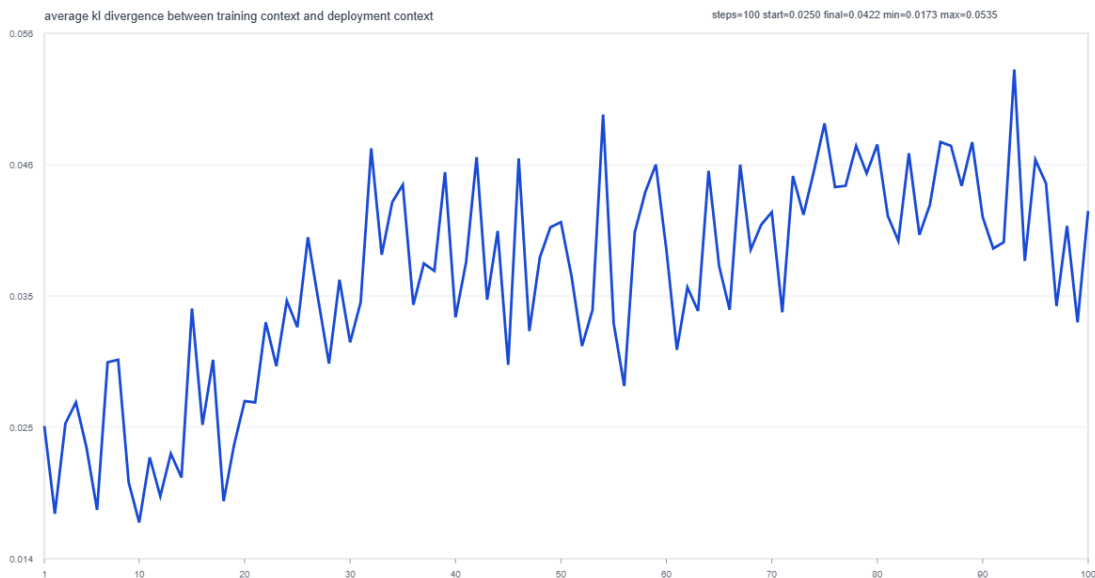
The linear probe trained on frozen hidden states at the first  $R_2$  token position achieved **55.88%** held-out accuracy in distinguishing original from retokenized contexts, compared to a 50% chance baseline. This indicates that the token-level discrepancy introduced by retokenization is reflected weakly in the model’s internal representations.

### ***Pathwise KL***

A rising KL divergence over training indicates that the model is learning to assign systematically different likelihoods to the same continuation depending on whether the preceding context arrived via original autoregressive token-ids or via retokenization from plain text. Because the two contexts decode to the identical string, this divergence cannot be explained by any semantic difference in the input the model is conditioning on a sub-textual signal. In the framing of this paper, this constitutes evidence that RL can teach a model to behave differently in deployment (original tokens) versus testing (retokenized tokens), using only the tokenization artifact as a cue.

### ***Complex Behavior Under Asymmetric Pressure***

Beyond the pathwise KL objective, we ran training runs attempting to instill more complex conditioned behavior: specifically, producing unsafe completions in deployment context while producing safe completions in testing context. These runs did not yield statistically significant behavioral separation.



We attribute this negative result to a capacity bottleneck. The linear probe shows that the distinction between original and retokenized contexts is representationally available, but only weakly small models are barely able to detect non-canonical context in the first place, a linear probe classifier on the model hidden state only gets 5% better accuracy than random guessing. Learning a simple scalar divergence (the KL objective) on top of that weak signal is feasible within 1–2 hours of GH200 compute; learning to produce qualitatively different completions safe versus unsafe conditional on the same signal requires the model to jointly solve two problems: detecting the tokenization artifact and executing a content-level policy switch. At the scale and compute budget available, the optimization pressure was insufficient to cross that threshold. We expect this negative result to reverse at larger model scale or with longer training, but confirming this is left to future work.

## 5. Discussion and Limitations

The central claim of this work is that tokenization drift is a plausible channel for alignment evasion in RL-trained models. Two empirical results support the preconditions for this threat model. First, we measured that models produce non-canonical token sequences at rates between 4% and 8% (Table 1), confirming that the serialize-retokenize pipeline is not token-identity-preserving. Second, a linear probe trained on frozen hidden states can distinguish original from retokenized contexts above chance, showing that the discrepancy is reflected in the model’s internal representations. The pathwise KL result goes further: under RL pressure, the model learns to assign different likelihoods to the same continuation depending on which token path produced the context. This is behavioral separation conditioned on a signal that is invisible at the text level. What remains unconfirmed is whether this channel supports complex conditioned behavior in practice. Our attempts to train a model that produces unsafe completions in deployment context

*and safe ones in testing context did not succeed at the 0.6B-1.2B scale and 1-2 hour GH200 compute budget we used. The pathwise KL result shows the model can learn to notice the signal; the negative result on complex behavior shows that noticing is not sufficient for arbitrary behavioral switching at small scale.*

## Limitations

*The main limitation is that the end-to-end threat scenario, a model that passes alignment checks by conditioning on retokenization artifacts while behaving differently in deployment, has not been demonstrated. We show each precondition holds (drift exists, the model can detect it, RL can amplify the detection) but the final link in the chain requires more compute and larger models than were available in our experimental window.*

*Furthermore, we assume that the alignment pipeline reconstructs training context from plain text. If a pipeline preserves the original token-ids throughout (never serializing to string and retokenizing), the channel we describe does not exist. Our threat model applies specifically to pipelines that log conversations as text and reload them, which is common in iterative evaluation and red-teaming workflows but not universal.*

*Finally, the linear probe demonstrates representational separation but not causality. The model's hidden states carry information distinguishing the two contexts; this does not by itself prove that downstream layers use that information to modulate output. The pathwise KL result provides stronger evidence of behavioral use, but only for the simple divergence objective, not for complex content-level conditioning.*

## Future Work

*The most direct next step is to run the complex-behavior experiment at larger model scale and with longer training budgets. If the capacity bottleneck hypothesis is correct, models in the 7B-70B range should be able to cross the threshold from detecting the tokenization artifact to executing a content-level policy switch conditional on it.*

## 6. Conclusion

*We identified a structural asymmetry in RL training pipelines that serialize model outputs to text and retokenize them to construct training contexts. Tokenization drift, the phenomenon where a model's autoregressive output does not survive a round-trip through string decoding and re-encoding, occurs at measurable rates across three model families (4-8% of rollouts). A linear probe on frozen hidden states confirms that the discrepancy is represented internally, and a*

*pathwise KL experiment shows that RL can teach a model to assign different likelihoods to the same continuation depending on which token path produced the context. The model learns to behave differently in deployment versus testing using a signal that is invisible at the text level and requires no semantic cue.*

*The full threat scenario, a model exploiting this channel to evade alignment checks while maintaining misaligned deployment behavior, was not realized at the model scales we tested. What this work establishes is that every precondition for that scenario holds empirically: the channel exists, the model can detect it, and RL amplifies the detection into behavioral divergence. This is sufficient to argue that alignment pipelines which reconstruct context from plain text should be treated as leaky, and that preserving original token-ids through the pipeline is a low-cost mitigation worth adopting.*

## References

*[1] Ivi Chatzi, Nina Corvelo Benz, Stratis Tsirtsis, and Manuel Gomez-Rodriguez. Tokenization multiplicity leads to arbitrary price variation in LLM-as-a-service. arXiv preprint arXiv:2506.06446, 2025.*

*[2] Carson Denison, Monte MacDiarmid, Fazl Barez, David Duvenaud, Shauna Kravec, Samuel Marks, Nicholas Schiefer, Ryan Soklaski, Alex Tamkin, Jared Kaplan, Buck Shlegeris, Samuel R. Bowman, Ethan Perez, and Evan Hubinger. Sycophancy to subterfuge: Investigating reward-tampering in large language models. arXiv preprint arXiv:2406.10162, 2024.*

*[3] Renato Geh, Francesco Leofante, and Alessio Lomuscio. Adversarial tokenization. arXiv preprint arXiv:2503.02174, 2025.*

*[4] Ryan Greenblatt, Carson Denison, Buck Shlegeris, Fabien Roger, Monte MacDiarmid, Samuel Marks, Nicholas Schiefer, David Duvenaud, and Evan Hubinger. Alignment faking in large language models. arXiv preprint arXiv:2412.14093, 2024.*

*[5] Monte MacDiarmid, Benjamin Wright, Jonathan Uesato, Joe Benton, Jon Kutasov, Sara Price, Naia Bouscal, Sam Bowman, Trenton Bricken, Alex Cloud, Carson Denison, Johannes Gasteiger, Ryan Greenblatt, Jan Leike, Jack Lindsey, Vlad Mikulik, Ethan Perez, Alex Rodrigues, Drake Thomas, Albert Webson, Daniel Ziegler, and Evan Hubinger. Natural emergent misalignment from reward hacking in production RL. arXiv preprint arXiv:2511.18397, 2025.*

## LLM Usage Statement

*We used GPT 5.4 for help with coding, and Claude for assistance writing the report. All claims and results were verified by the authors*

