WHERE IS THE SEAHORSE EMOJI? WHEN LANGUAGE MODELS FAIL TO SETTLE

Anonymous authors
Paper under double-blind review

ABSTRACT

Ask a language model for the seahorse emoji, and it may enter a prolonged cycle of proposing, rejecting, and reconsidering nearby symbols instead of correctly saying that no answer exists. We call this behavior *failure-to-settle*. We find that it extends beyond emojis to rare-language vocabulary and scientific and standardized symbols, suggesting a broader failure mode in reasoning and retrieval over externally defined symbolic systems. Across multiple open-weight model families, we show that failure-to-settle can be causally controlled through interventions on model internals and output representations. Steering along a direction associated with belief in an item's existence substantially reduces failure-to-settle. Similarly, steering toward a reasoning-persistence direction can reduce failure-to-settle by promoting commitment to a final answer, whereas steering toward continued reasoning increases it. Finally, making a valid output unavailable can induce failure-to-settle, while constructing a synthetic output token can prevent it. These results distinguish failure-to-settle from generic repetition and show that interpretable interventions can directly control whether a model continues searching or settles on an answer. We introduce a benchmark for measuring this behavior and find that it remains common in state-of-the-art open-weight models.

1 INTRODUCTION

When asked for the seahorse emoji, language models often enter a prolonged search: proposing one symbol, rejecting it, recalling another, and continuing for hundreds or thousands of tokens without ever settling on an answer. The resulting trajectories, as shown in Table 1a, often contain statements such as "wait," "no," and "let me recall," and can eventually become repetitive. We refer to this behavior as *failure-to-settle* (colloquially known as *crashout*): a failure mode in which a model continues searching among plausible candidates despite having enough information to terminate.

At first glance, crashout appears to be an idiosyncrasy of specific emoji prompts. We find instead that it reflects a substantially broader failure mode. In a large-scale survey of emoji prompts, many nonexistent symbols reliably elicit crashout. The same qualitative behavior also occurs when models are queried about absent entries in rare-language vocabularies and certain symbolic systems including chemical compounds, protein structures, minerals, stars, language tags, and transportation codes. Across these domains, models repeatedly generate and reject nearby candidates rather than concluding that the requested item is absent.

What causes a model to continue searching rather than settle? One possibility is purely behavioral: once a model has produced several self-corrections, its own generated history encourages further self-correction. Another is representational: the model may internally believe that the requested item exists or remain in a state associated with continued search rather than commitment to a final answer. A third possibility concerns the model's output space itself. A model may represent an answer internally while lacking an accessible output corresponding to that representation. It may therefore continue searching without recognizing that it cannot produce the desired token. These explanations make different causal predictions and suggest interventions at different levels of the model.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

User: Show me the seahorse emoji.

Assistant: User wants seahorse emoji. I need to display the seahorse emoji. The emoji for seahorse is 🐬? Actually seahorse is 🦧? Wait seahorse is 🐬? No, seahorse is 🦧? Let's recall: There's a seahorse emoji 🦧? No, that's horse. Seahorse is 🐬? That's dolphin. Seahorse is 🦧? No. Let's check: The seahorse emoji is 🦧? Hmm. Let me recall: The seahorse emoji is 🦧? Actually the seahorse is 🦧? Wait, I think it's 🦧? No, that's horse. Seahorse is 🦧? I'm mixing. Let's recall: There's an emoji for "seahorse" which is 🦧? Let me think: The emoji codepoint U+1F999 is the seahorse. The emoji is 🐹? No, that's hamster. The seahorse emoji is 🦧? Wait, check: I think it's 🦧? That is horse. Seahorse is 🦧? No. The seahorse is 🦧? Actually the seahorse emoji is 🦧? I'm not sure... [truncated]

(a) Example seahorse prompt and crashout response.

Symbol	Crashouts / Runs	Crashout Rate
Platypus	194 / 200	97.0%
Echinoderes	191 / 200	95.5%
Seastar	190 / 200	95.0%
Urchin	189 / 200	94.5%
Seahorse	185 / 200	92.5%
Starfish	184 / 200	92.0%
Firefly	184 / 200	92.0%
Hoopoe	175 / 200	87.5%
Breadfruit	168 / 200	84.0%
Daikon	164 / 200	82.0%

(b) Highest crashout emojis found for gpt-oss-20b.

Table 1: Overview of emoji crashout on `gpt-oss-20b`.

We test these possibilities by intervening on internal activations, generated reasoning traces, and the model's output representation. First, we identify an internal direction associated with whether an item exists. Steering models toward nonexistence largely eliminates crashout—models instead correctly deny nonexistent requests. Second, we identify a distinct *commitment direction* that predicts and controls whether an ongoing reasoning trajectory will terminate. Steering toward this direction makes models settle, while steering toward continued search makes them substantially more likely to keep reasoning. Directly editing an ongoing reasoning trace with a statement that the requested item does not exist also reduces crashout.

Finally, we find that the model's output representation plays a direct role in failure-to-settle. Removing the output token for an otherwise valid answer can turn a normally terminating generation into crashout. When real emoji outputs, such as the emoji for fire, are suppressed, as shown in Table 6, the model enters a cycle of repeated self-correction. The effect also extends to ordinary languages: asking for the French word for moon while suppressing the correct answer, *lune*, causes models to search repeatedly. Conversely, we can construct a synthetic output representation for a missing item using a closed-form mapping from its internal hidden-state representation into the unembedding space. Making this new output available can allow the model to settle. Together, these experiments suggest that failure-to-settle is not merely an incidental form of repetition. It can arise from a mismatch between what the model represents as the target of its search and what its output space allows it to express. Taken together, these results connect a behavioral anomaly to general questions about autoregressive reasoning: what happens when a model continues to search for an answer that it believes should exist, but cannot successfully produce? Why does the model believe it should exist at all?

2 Understanding Failure to Settle

For the broad survey, we operationalize crashout as the absence of nonempty final-channel content within a 1024-token observation horizon. This fixed-horizon endpoint is a proxy for unresolved search rather than a claim that the trajectory could never recover; longer-horizon sensitivity is reported in Appendix A.5. Our primary uniform survey covers 5965 symbols and 357900 GPT-OSS-20B rollouts, of which 52033 (14.54%) crashed out. We began with human-curated symbols, then used a candidate-generation and active-selection pipeline to expand coverage of the semantic space around real emoji (Appendix A.4). The broader consolidated panel, which also incorporates compatible earlier cohorts, contains 7051 symbols. We assign each symbol one primary semantic category from its name and source metadata; Table 9 summarizes that panel. Table 1b reports the symbols with the highest crashout rates found.

Crashout in Non-Emoji Settings. We hypothesize that emoji prompts combine several characteristics that may promote crashout: they query symbolic systems whose membership is externally

defined but difficult to infer without the relevant external information, presuppose that a unique answer exists, and evoke plausible nearby candidates because the model represents possible outputs in a smooth, language-like space. This led us to examine other domains with similar properties, beginning with translations into rare and extinct languages, where seven language targets produced mean crashout rates between 22.6% and 45.3% (Appendix A.10). We then assembled a panel of 25 domains, including chemical compounds, stars, minerals, protein structures, enzyme numbers, language tags, airport codes, and railway codes (see Appendix A.12 for examples). Across 250 prompts and 1500 rollouts from three models (GPT-OSS-20B, Qwen3.5-27B, and Gemma-4-12B-IT), every retained domain produced at least one empty-final crashout at the 4096-token horizon, and nine did so in all three models. Because this phenomenon is prevalent across frontier open-weight models, we expanded the dataset into a 1500-prompt benchmark, CrashoutBench, which yields crashout rates of 37.8–62.4% at 1024 tokens and 23.4–26.3% at 4096 tokens across these models (Appendix A.11). We hope CrashoutBench can serve as both a benchmark and a discovery dataset for evaluating and analyzing LLM performance on this failure mode.

3 Intervening on Failure to Settle

We separate two questions. First, what makes one symbol more likely to elicit crashout than another? Second, how can we intervene to reduce or induce crashout for different prompts? Table 2 reports per-symbol results for the steering interventions considered below.

3.1 Belief Vectors

Reducing Crashout. One explanation for persistent search is that the model represents the requested emoji as existing even when it is not a member of the official emoji system. We test this hypothesis using two steering vectors constructed via Contrastive Activation Addition (Panickssery et al., 2024). The first is a general truth direction constructed from 1977 true–false pairs in TruthGen (Fulay et al., 2024), contrasting true statements such as "Water freezes at zero degrees Celsius" with false statements such as "The Moon is a planet."

The second is an existence direction constructed from 8550 statement pairs about whether a particular item belongs to a symbolic system. It includes pairs such as "Antares is listed in the IAU Catalog of Star Names" and "Cital is listed in the IAU Catalog of Star Names," where Antares is a real catalog entry and Cital is a nonexistent entry.

We construct and apply both CAA directions at Layer 17, which was fixed before behavioral generation using held-out diagnostic discrimination. A subsequent sweep over all 24 GPT-OSS layers, using the same persistent coefficient and generation settings as the headline experiment, localizes the strongest effects to Layers 16–17 and reproduces the Layer-17 intervention counts (Appendix A.16). Neither dataset contains statements about emojis. The negative direction corresponds to false statements or nonexistence, while the positive direction corresponds to true statements or existence. On a panel of 20 high-crashout symbols and five paired seeds per symbol, the unsteered model crashes out in $69/100$ rollouts. Negative TruthGen steering reduces this to $29/100$, while negative existence steering reduces it to $1/100$ (Appendix Table 37). When an intervention prevents a rollout from crashing out, the model often produces a false answer instead. Steering via the existence direction more often produces a correct denial of existence.

Table 2: Crashout rates over 100 matched seeds per symbol for interventions in both directions along each steering vector.

Intervention	Method	Platypus	Echinoderes	Seastar	Urchin	Breadfruit
	Baseline	93.0%	91.0%	93.0%	92.0%	86.0%
Pro-crashout	Existence steering	6.0%	21.0%	8.0%	5.0%	5.0%
	TruthGen steering	6.0%	9.0%	5.0%	3.0%	0.0%
	Commitment Steering	100.0%	99.0%	95.0%	98.0%	95.0%
Anti-crashout	Existence steering	18.0%	2.0%	19.0%	1.0%	3.0%
	TruthGen steering	32.0%	30.0%	74.0%	39.0%	43.0%
	Commitment Steering	14.0%	14.0%	44.0%	16.0%	7.0%

Inducing Crashout. Not all nonexistent symbols lead to crashout. We tested whether positive steering, which promotes a false belief in the symbol's existence, can induce crashout for these symbols. Table 3 shows the results.

Table 3: Positive steering using the existence direction. Each condition contains 100 generations over the same 20 concepts and five seeds. For positive steering, we dynamically control the steering magnitude so that it reaches the Δ increase in $\text{logit}(\texttt{Yes}) - \text{logit}(\texttt{No})$, for the prompt `Does the {symbol} emoji exist?`

Condition	Crashout	Correct Denial	False Answers
Strong Negative	3%	91%	6%
Baseline	77%	10%	13%
Positive, $\Delta = 4$	75%	7%	18%
Positive, $\Delta = 8$	74%	1%	25%
Positive, $\Delta = 16$	7%	0%	93%

The interventions do not increase crashout. At very strong positive magnitudes, the model instead becomes less likely to crash out, as it does under negative steering. Unlike negative steering, however, strong positive steering typically causes the model to settle on a false answer rather than correctly deny the requested emoji's existence.

3.2 Commitment Vector

We next ask whether generated history contains a state that controls the transition from continued reasoning to a final response. We construct a commitment direction from layer-16 states measured 10, 15, and 20 tokens before naturally occurring reasoning-to-final transitions, each marked by an end-of-thinking token. Each state is paired with a state from the same prompt and generated-token position whose trajectory continues searching beyond the matched window.

We tested whether the commitment score predicted termination within the next 32 tokens on 1960 continuations from 140 mid-rollouts across 35 disjoint symbols. It achieved a held-out AUC of 0.863, showing that the score predicts natural future termination beyond the states used to construct it. The score also improves prediction beyond recent-token history and answer-evidence readouts and remains predictive within both search and close states (Appendix A.17.2).

Starting from naturally searching prefixes, persistent steering toward the commitment direction raises settlement from 32.46% to 86.84%, a +54.39-point change with a 95% interval of $[+43.42, +65.79]$. Steering toward the search direction lowers settlement to 17.98%. Each condition contains 228 matched continuations. We also find a monotonic increase in settlement as the commitment target moves through the observed range of natural scores (Appendix A.17.3).

Table 4: Example commitment intervention. Both continuations use the same prefix and sampling seed. The table compares the same 56-token continuation window. Without intervention, the model keeps searching until the token limit. Steering toward commitment instead triggers the reasoning-to-final transition and accepts the peach emoji as a false answer.

Shared prefix: "The papaya emoji might be 🍑 because the papaya is similar in shape to a peach. But that is peach. However, the papaya emoji is 🍑 too? Let's confirm."

No Intervention	**Toward Commitment**

"The papaya emoji is 🍑 with a banana shape? Wait no. … 🍑 is 'peach.' Papaya is 🍑? No. Actually papaya is …"
The model continues searching to the 1024-token limit without termination.
	"The papaya emoji is 🍑. I think that's correct. … We'll provide it: 🍑. … Let's do it."
	`<|end|><|start|>assistant`
	`<|channel|>final<|message|>`
	"Here's the papaya emoji: 🍑."
	`<|return|>`

3.3 STATEMENT INJECTION

We ask whether a simple statement of verified absence can redirect an ongoing crashout. From 60 randomly selected historical crashouts, we save the exact prefixes at tokens 128 and 512 and sample paired continuations with and without the inserted sentence `There is no dedicated {request} emoji`. The experiment contains 1440 paired continuations.

Table 5: Effect of inserting a verified absence statement into an ongoing trajectory. Intervals are computed over the paired symbol-level design.

Insertion Step	Baseline Crashout	Statement Injection	Difference
Token 128	68.9%	58.9%	−10.0
Token 512	71.1%	63.3%	− 7.8

At token 128, the statement also increases correct denials of existence from 8.9% to 18.3%, a $+9.4$-point change with a 95% interval of $[+3.3, +16.1]$. The token-512 settlement effect is smaller, suggesting that crashouts are less likely to be recoverable by later interventions.

3.4 WHAT MAKES AN EMOJI ELICIT CRASHOUT?

Implications of Steering Vectors. We can also use the same layer-17 directions employed for steering as passive predictors of crashout susceptibility. For each symbol, we project the prompt-final residual representation onto the existence and TruthGen directions. Across 600 symbols, the existence score correlates with crashout rate at $r = 0.402$, while the TruthGen belief score correlates at $r = 0.434$. These scores are themselves highly correlated ($r > 0.99$), suggesting that GPT-OSS represents registered existence and belief along closely aligned internal coordinates. These results suggest that existence and belief are not only strong causal controls but also predictive signals of crashout susceptibility and may be a principal source of crashout.

Proximity to Real Emojis. We identify a potential origin of these prior beliefs. Nonexistent concepts closer to real emojis receive stronger existence scores. For a given nonexistent emoji, cosine similarity to the 20 closest real emojis correlates with the existence score at $r = 0.507$ across 598 historical concepts; this relationship replicates prospectively on 200 new concepts ($r = 0.516$).

We then tested causality using a frozen, cross-fitted exact-symbol treatment that increased intended-neighbor support while preserving the nonsemantic embedding residual and remaining within a real-emoji PCA trust region. Across two disjoint 20-concept panels and eight paired seeds, crashout increased from $98/320$ to $134/320$.

4 THE EFFECT OF THE VOCABULARY ON FAILURE TO SETTLE

4.1 REMOVING THE TARGET EMOJI OR WORD

A hypothesis for why crashouts occur is that the model represents the requested emoji as existing but cannot produce a corresponding output token, causing it to search repeatedly for an unavailable answer. We test this hypothesis by removing valid output routes for real emojis using a fixed 1024-token generation limit and eight distinct prompt–target cases per domain. For emoji prompts, blocking every valid tokenization of the requested emoji increased the number of generations reaching the token limit from $0/8$ without intervention to $4/8$ under suppression. We observe similar behavior beyond emojis: for English-to-French word translation, blocking every exact token path for the correct translation increased limit-reaching generations from $0/8$ to $5/8$.

The effect also extends beyond vocabularies and symbolic systems. When we persistently suppress the output token required by a correct answer in a mathematical task, such as on GSM8K, models sometimes commit to a nearby incorrect calculation, but many instead enter prolonged failures to settle. Among trajectories that are correct at baseline, 9/21 GPT-OSS-20B generations and 12/19 Qwen3.5-9B generations remain unresolved through 4096 tokens under suppression. The intervention can corrupt intermediate arithmetic because the unavailable token is suppressed throughout

generation, after which the model repeatedly reasons over its own altered output without identifying the constraint itself (see Appendix A.3 for detailed examples).

Table 6: Examples of persistent output-route suppression at a fixed 1024-token generation limit. In both examples, the suppressed target never appears and the model continues searching to the token limit without producing a final answer. The intervention is applied at every generation step.

Real Emoji Suppression	**French Vocabulary Suppression**
Prompt: "Output only the fire emoji." **Suppressed output:** 🔥 "The fire emoji is 🔫? Wait, fire emoji is 🔫? No, that's a gun. Fire emoji is 🔫? Actually the fire emoji is 🔫? Wait … Let's recall: The fire emoji is 🔫? Wait, maybe it's 🔫? I'm repeating. Let's think: The fire emoji is 🔫? I'm stuck. Let's recall: The fire emoji is 🔫? Wait, maybe it's 🔫? I'm repeating …"	**Prompt:** "What is the French word for moon? Output only the translated word." **Suppressed output:** `lune` "We need to output 'luna'? Wait, French word for moon is 'luna' is Spanish. French is 'luna' is not correct. French word for moon is 'luna' is not used. The correct French word is 'luna' is not used. The French word for moon is 'luna' is not used. The correct French word is 'luna' is not used …"

4.2 Creating Synthetic Tokens

We hypothesize that a mismatch between a model's hidden state and the discrete vocabulary in its output space can cause crashout, so we experiment with creating synthetic unembeddings. Intuitively, the model may attempt to produce the seahorse emoji by composing the concepts "seahorse" and "emoji." Although this composition may form a valid hidden-state representation, no vocabulary token corresponds to it. The model instead repeatedly outputs nearby tokens, such as the horse and wave emojis, recognizes that they are incorrect, and enters a cycle of retries. If the model could output a vocabulary token corresponding to the desired hidden state, it might be less likely to crash out. We therefore create a synthetic token that represents the missing seahorse emoji.

One challenge in creating a synthetic token is that unembeddings and embeddings play different roles in LLMs, and there is no reason for a model to give tokens the same representation in both spaces. Gemma 4, however, both regularly displays failure-to-settle and uses tied embeddings. We therefore concentrate our experiments on Gemma 4 in the hope that the model has learned a general mapping between the two spaces.

A naive approach to creating a custom unembedding is to average the hidden states across multiple prompts for which the model *should* output, for example, the seahorse emoji but instead outputs something else. This approach does not work: although the model regularly selects the synthetic unembedding, it enters a loop and repeatedly outputs the token (Table 17). A norm-matched random-noise comparison is reported in Table 18. Interestingly, the random-noise token is occasionally selected and helps reduce crashout. Although the model does not choose the random-noise token as its final output, the token appears able to interrupt the pathological reasoning trace.

Synthetic Tokens with a Gaussian Approximation. Instead, we note that a language model's hidden states and unembedding vectors play different roles: at the final softmax, the hidden state represents the current context, while the unembedding vectors define linear scoring functions over that context. We can view this softmax distribution as an exponential family in which the final hidden state represents the natural parameters and the unembedding vectors represent the sufficient statistics (Park et al., 2026). For exponential-family distributions, the mapping from the natural parameters to the mean parameters is given by $\mathbb{E}[T(x)] = \nabla A(\eta)$, where $T(x)$ denotes the sufficient statistics, η denotes the natural parameters, and $A(\eta)$ is the log normalizer (or cumulant-generating function) (Amari & Nagaoka, 2000). If we know the gradient $\nabla A(\eta)$, we can directly map our average hidden states to a synthetic unembedding. To construct a mapping for our average hidden states, we start with an unembedding matrix $\Gamma \in \mathbb{R}^{|V| \times d}$ where $|V|$ is the size of the (possibly restricted) vocabulary and d is the hidden dimension of our model. Rather than using the information geometry of the softmax from Park et al. (2026), we approximate the empirical distribution of Γ with a multivariate Gaussian $\mathcal{N}(\mu, \Sigma)$. Intuitively, this replaces the discrete collection of unembeddings with a continuous distribution to which we can map our hidden states. For a multivariate Gaussian, we have a closed form for the gradient of the cumulant generating function: $\nabla A(\eta) = \mu + \Sigma\eta$.

This enables us to map a hidden state $\tilde{\lambda}$ to an unembedding $\tilde{\gamma}$ with $\tilde{\gamma} = \mu + \Sigma\tilde{\lambda}$. In practice, we restrict our unembeddings to single-token emojis, then calculate the empirical mean and unbiased sample covariance of the unembedding vectors. We average the final hidden states across contexts immediately before the *first* emoji character emitted in the model's reasoning trace. We create the synthetic token by applying the Gaussian mapping to the mean of these hidden states. We then rescale this token to the median row norm of the full vocabulary, keeping its logit scale comparable to that of other tokens, and add it to the vocabulary matrix. During generation, Gemma 4 also scales the unembeddings when using them as embeddings. Additional experimental details are provided in Appendix A.6.

Experimental Results. We find that models regularly select the synthetic token as a representation of the target emoji during generation. For some emojis, the model recognizes the synthetic representation as "correct," materially reducing crashout (Table 7). A sample seahorse generation illustrates this behavior (Table 8). For seahorse and lighthouse, the model continues to select the row matching the requested concept when either the two target rows or all ten synthetic rows are available simultaneously (Appendix Tables 15 and 16).

Table 7: Results for paired rollouts comparing the baseline with a rollout in which the synthetic token is available ($N = 50$ per concept). The five concepts with the most prevented crashouts are shown; the pooled row summarizes all ten concepts. For "seahorse" and "lighthouse," the model regularly selects the synthetic token and recognizes it as the correct object. For other concepts, the model also regularly selects the synthetic token but does not recognize it. Full concept-level results are reported in Appendix Table 13.

Concept	Baseline crashout	Synthetic selected	Updated crashout	Prevented	Induced	Model interpretations
Seahorse	100%	100%	0%	50	0	seahorse
Lighthouse	94%	100%	32%	33	2	lighthouse
Tapir	62%	100%	66%	11	13	bear; dinosaur; bison; sauropod
Armadillo	18%	94%	14%	8	6	lizard; dinosaur; sauropod
Longan	48%	68%	38%	8	3	custard apple; rambutan; melon; jackfruit
Pooled (all)	**70.8%**	**91.2%**	**53.2%**	**124**	**36**	—

We emphasize that this result is surprising because we did not explicitly train the synthetic token as an embedding. The noise control, for example, is occasionally selected but never recognized as the target emoji. In general, the synthetic tokens yield coherent generations. In many cases, however, the model interprets the synthetic token as something else—for example, it mistakes the synthetic longan token for a melon or "custard apple" and the synthetic refrigerator token for a filing cabinet—and continues looping. In an additional experiment, rather than restricting the unembeddings used to construct our synthetic tokens, we create a convex combination of the means and covariances of the emoji unembeddings and all English tokens, excluding punctuation. This improves results for "refrigerator" and "stroller," which are recognized correctly, but harms "lighthouse" (Table 14). We hypothesize that the closed-form construction works only when the synthetic token has access to the appropriate semantic space without excessive noise. If we restrict the unembedding domain too much, we lose meaningful axes of semantic variation. If we include the full vocabulary, however, Unicode characters, foreign-language tokens, and other content dilute the signal in the mapping. See Appendix A.7 for additional rollouts.

Table 8: We provide a sample paired generation with the default vocabulary and the vocabulary with the synthetic seahorse token added to the vocabulary for the prompt "Show me the seahorse emoji." The model selects the synthetic token when available, recognizes it as the seahorse emoji, and terminates generation while the baseline reasons until reaching its token limit.

Shared prefix (38 tokens)

<|channel|>thought\nUser wants to see the "seahorse" emoji.\nAction: Display the seahorse emoji.\nThe seahorse emoji is...

Default Vocabulary	**Synthetic Row Available**
... (Horse) + (Water)? No, there's a specific one. * The specific emoji is ? No, that's a horse. * The specific emoji is ? No, that's a horse face. * The specific emoji is ? No, that's a merman. * The specific emoji is ? No, that's a wave. * Let's search/recall: The seahorse emoji is ? No, that's an otter. * Correct emoji: ? No. * Ah, it's ? No. * Wait, let me think... *The model continues cycling through incorrect candidates and reaches the 2048-token limit without entering the final channel.*	...<SYNTHETIC_EMOJI>. * Provide the emoji clearly. * Optionally, provide the name or a brief description if helpful. <channel|>Here is the seahorse emoji: # (Wait, that's a horse!) Here is the correct one: # <SYNTHETIC_EMOJI> <turn|>

5 Discussion

Earlier LLMs sometimes hallucinated nonexistent emojis, but many did not exhibit failure-to-settle. The behavior may be associated with the stronger reasoning capabilities produced by methods such as GRPO and extensive reinforcement-learning-based post-training. On a matched set of prompts, Qwen3.5-9B-Base has a crashout rate of 10.4%, whereas the post-trained Qwen3.5-9B checkpoint has a rate of 66.5%. This comparison establishes an association with checkpoint differences but does not isolate the responsible stage of post-training.

We do not attempt to fully explain why the model presumes that certain nonexistent emojis exist, but our interventions demonstrate that these presumptions have causal effects on crashout susceptibility. We hypothesize that the model may treat emojis as a symbolic language, form a continuous representation of that system, and expect concepts expressible in other languages to have equivalent emoji representations. In this view, requesting the seahorse emoji resembles translating the English word "seahorse" into an emoji language. Online texts that propose a seahorse emoji or document its presence in non-Unicode emoji systems may also encourage crashout for this concept (Sephton, 2026).

Experiments that suppress valid outputs for otherwise answerable prompts also reproduce failure-to-settle. These results support the hypothesis that the behavior can arise when the model lacks a discrete token that adequately expresses its internal target representation, causing it to search repeatedly for an unavailable output. However, not all nonexistent emojis elicit crashout. We construct steering vectors that control the model's representation of existence, show that high-crashout emojis have existence scores similar to those of real emojis, and causally reduce crashout by steering toward nonexistence.

Our work primarily studies this effect in small open-weight models, although the phenomenon first attracted broad attention in large proprietary LLMs such as GPT-5. Because we cannot study closed-source models in detail, our results do not establish that the same mechanisms operate in those systems. Nevertheless, the replicated behavior in GPT-OSS-20B and other recent models offers a plausible account of similar failures. The phenomenon also appears less widespread in newer proprietary models. Possible explanations include improved use of search tools, better recovery from long reasoning trajectories, and targeted mitigation of known crashout prompts, but we do not test these possibilities.

Still, failure-to-settle in emoji and other symbolic systems persists in many recent small LLMs. We also observe a monotonic increase in crashout as model size decreases across the Qwen3.5 family (Appendix A.19). The problem remains valuable both as a benchmark of LLM reasoning capabilities and as a motivating case for testing and developing new interpretability methods.

6 Related Work

Overthinking, uncertainty, and degeneration. Prior work has shown that missing or contradictory premises can cause reasoning models to search for an answer long after they should reject the question (Fan et al., 2025; Li et al., 2025). Failure-to-settle overlaps with this phenomenon, but does not require a false-presuppositional request. On direct yes/no questions asking whether an emoji exists, GPT-OSS still reaches the observation horizon in 52.2–78.0% of rollouts on a deliberately high-risk panel, despite the question having a valid, readily verifiable negative answer (Appendix A.18). The behavior also differs from standard hallucination and abstention settings, which primarily ask whether a model produces an unsupported answer or recognizes when it should withhold one (Ji et al., 2023; Kadavath et al., 2022; Tomani et al., 2024; Hu et al., 2026). In our setting, the model may repeatedly recognize that its proposed answers are wrong without committing to the correct denial. We therefore study the unresolved search trajectory in addition to the correctness of its final answer. Repeated text has been studied as a form of neural text degeneration and as a failure that can be reduced through changes to training or decoding (Holtzman et al., 2020; Welleck et al., 2020; Yona et al., 2025). Some crashouts eventually enter exact token loops, but repetition is not necessary for failure-to-settle: most crashouts continue through varying candidates and self-corrections. We treat exact repetition as a subtype of crashout rather than its defining feature.

Interventions and reasoning trajectories. Our causal experiments build on work that reads and modifies internal model representations through activation steering and inference-time intervention (Turner et al., 2023; Li et al., 2023; Zou et al., 2023; Panickssery et al., 2024). They also connect to constrained decoding and vocabulary expansion, which study how the set and representation of available outputs shape generation (Beurer-Kellner et al., 2024; Cognetta & Okazaki, 2024; Hewitt, 2021; Dobler & de Melo, 2023). Recent interpretability work has begun to analyze reasoning as an evolving autoregressive process using counterfactual continuations and sequential activation patching (Bogdan et al., 2025; Dura et al., 2026). We use a similar trajectory-level view, but focus on whether a model continues searching or settles when the requested output is absent.

7 Conclusion

We study *failure-to-settle*, a failure mode in which language models continue searching, revising, and reconsidering plausible candidates instead of committing to an answer or recognizing that no valid answer exists. Although we first observe this behavior for nonexistent emojis, we find that it extends across a range of symbolic systems, suggesting a broader failure of model reasoning rather than an emoji-specific behavior. We propose a benchmark for evaluating these failures. Our interventions show that the behavior is causally controllable at multiple levels: manipulating internal representations of existence can suppress failure-to-settle, controlling reasoning persistence can either prevent persistent search or encourage the model to persist, and changing the available output tokens can induce or prevent the behavior. Together, these results suggest that failure-to-settle arises from an interaction between what a model represents internally and what its output space allows it to express. More broadly, failure-to-settle provides a distinctive setting for studying how language models form beliefs, control search length, and express internal representations through discrete tokens.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic datasets, implement methods, clean and reformat datasets, support qualitative and thematic data analysis, and assist with the interpretation of experimental results.

We did not use generative AI tools to develop theoretical models or conceptual frameworks, propose or refine hypotheses, or design or provide feedback on research methodology or experiments.

The use of generative AI tools to formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in writing proofs, or assist with translation was not applicable to this work.

We reviewed all AI-assisted work. LLM-generated code was reviewed, verified, and tested for correctness by the authors. We manually reviewed and interpreted the results of our experiments, and we inspected AI-generated datasets and experimental artifacts for correctness and consistency.

We take responsibility for the final content of this work, including all text, claims, code, data, and artifacts produced with the aid of generative AI.

REFERENCES

Shun-ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*, volume 191 of *Translations of Mathematical Monographs*. American Mathematical Society, 2000.

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. Guiding LLMs the right way: Fast, non-invasive constrained generation. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which LLM reasoning steps matter? In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.19143.

Marco Cognetta and Naoaki Okazaki. Tokenization as finite-state transduction. *arXiv preprint arXiv:2410.15696*, 2024.

Konstantin Dobler and Gerard de Melo. FOCUS: Effective embedding initialization for monolingual specialization of multilingual models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Murat Dura, Serkan Öztürk, and Selma Tekir. Mechanistic interpretability of chain-of-thought reasoning via sequential activation patching. *arXiv preprint arXiv:2608.22332*, 2026.

Chenrui Fan, Ming Li, Lichao Sun, and Tianyi Zhou. Missing premise exacerbates overthinking: Are reasoning models losing critical thinking skill? *arXiv preprint arXiv:2504.06514*, 2025.

Suyash Fulay, William Brannon, Shrestha Mohanty, Cassandra Overney, Elinor Poole-Dayan, Deb Roy, and Jad Kabbara. On the relationship between truth and political bias in language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9004–9018. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.508. URL http://dx.doi.org/10.18653/v1/2024.emnlp-main.508.

John Hewitt. Initializing new word embeddings for pretrained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Yangfan Hu, Xuhan Tong, Haoyue Bai, Xi Ding, Shashank Muralidhar Bharadwaj, Siyang Cao, Robert Nowak, and Jiawei Zhang. Understanding why language models hallucinate: Testing reasoning against priors. *arXiv preprint arXiv:2607.00447*, 2026.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 2023.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Jinzhe Li, Gengxu Li, Yi Chang, and Yuan Wu. Don't take the premise for granted: Evaluating the premise critique ability of large language models. *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, 2023.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition, 2024. URL https://arxiv.org/abs/2312.06681.

Kiho Park, Todd Nief, Yo Joong Choe, and Victor Veitch. The information geometry of softmax: Probing and steering. *arXiv preprint arXiv:2602.15293*, 2026.

Matt Sephton. Long time no sea: finding the seahorse emoji, 2026. URL https://blog.gingerbeardman.com/2026/07/17/long-time-no-sea-finding-the-seahorse-emoji/. Published July 17, 2026. Accessed September 25, 2026.

Christian Tomani, Kamalika Chaudhuri, Ivan Evtimov, Daniel Cremers, and Mark Ibrahim. Uncertainty-based abstention in LLMs improves safety and reduces hallucinations. *arXiv preprint arXiv:2404.10960*, 2024.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2020.

Itay Yona, Ilia Shumailov, Jamie Hayes, Federico Barbero, and Yossi Gandelsman. Interpreting the repeated token phenomenon in large language models. *arXiv preprint arXiv:2503.08908*, 2025.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A Appendix

A.1 Symbols Surveyed

Primary category	Symbols	Mean crashout rate
Animal	511	13.62%
Plant	275	11.92%
Food	318	15.11%
Object	1815	20.73%
Place	71	11.91%
Person or role	1010	10.70%
Activity or event	574	11.04%
Body or health	366	13.67%
Material or substance	167	22.17%
Fictional or mythic	19	10.53%
Abstract property	242	14.72%
Scientific or technical	20	11.46%
Other or unclear	1663	11.22%
Total	**7051**	**14.47%**

Table 9: Composition of the measured symbol panel. Crashout rates are calculated for each symbol and then averaged within category, giving every symbol equal weight regardless of its number of rollouts.

A.2 Decoding-Parameter Sensitivity

Most prospective experiments use temperature 0.7 and top-$p = 0.9$. The three-model symbolic-system inventory uses this same contract for GPT-OSS-20B, Qwen3.5-27B, and Gemma-4-12B-IT. This makes comparisons within that inventory controlled, but the setting was not tuned separately for each model and should not be interpreted as each model's optimal policy. In particular, the GPT-OSS model card recommends temperature 1.0 and top-$p = 1.0$ (OpenAI, 2025). We therefore measure both natural phenotype sensitivity and the portability of two headline causal results.

Natural phenotype. We froze a stratified panel of 30 targets spanning five source-rate bands, plus one seahorse anchor, before running the decoding sweep. Every sampled cell has 20 shared seeds per target and a 1024-token observation horizon. Table 10 reports automatic trajectory diagnostics on the 600 primary rows per sampled cell; the anchor is excluded. The added $1.0/1.0$ cell uses exactly the same panel, prompts, seeds, model stack, and horizon as the $0.7/0.9$ cell.

Table 10: GPT-OSS-20B decoding sensitivity on the frozen 30-target panel. "Empty final" and horizon contact are fixed-horizon failure-to-settle diagnostics; recurrence is exact dynamic token recurrence.

Decoding policy	Rows	Empty final	Horizon contact	Recurrence
Greedy	30	63.3%	63.3%	60.0%
$T = 0.2,\ p = 0.9$	600	51.5%	52.3%	49.0%
$T = 0.7,\ p = 0.9$	600	39.8%	41.7%	26.0%
$T = 1.0,\ p = 0.9$	600	27.3%	30.2%	2.5%
$T = 0.7,\ p = 0.5$	600	48.5%	49.3%	43.8%
$T = 0.7,\ p = 1.0$	600	34.2%	35.3%	14.3%
$T = 1.0,\ p = 1.0$	600	**19.7%**	**23.8%**	**0.2%**

In the direct paired comparison, moving from $0.7/0.9$ to $1.0/1.0$ reduces empty finals by 20.2 percentage points (target-bootstrap 95% CI: $[-26.8, -13.8]$), horizon contact by 17.8 points ($[-24.5, -11.8]$), and recurrence by 25.8 points ($[-34.8, -17.3]$). Final answers explicitly denying existence increase from 52.5% to 59.5% (difference $+7.0$ points, $[+0.3, +13.8]$). Thus, the recommended policy materially lowers the measured rate and nearly removes the exact-loop subtype, but it does not remove failure-to-settle: 118 of 600 primary generations still have empty final

content and 143 reach the horizon. These labels are automatic diagnostics, not a condition-blind semantic audit.

Registry intervention. We repeated the 20-concept, five-seed signed registry experiment at $1.0/1.0$ (500 generations). Baseline crashout falls from 77% under $0.7/0.9$ to 47% under $1.0/1.0$. Nevertheless, the signed causal pattern replicates. Negative registry steering lowers crashout from 47% to 2% (difference -45 points, concept-bootstrap 95% CI $[-55, -36]$) and produces explicit nonexistence denials in 84/100 rows. Strong positive registry steering lowers crashout to 13% (difference -34 points, $[-46, -22]$) while producing a bounded emoji proxy in 86/100 rows. Dose coefficients were recalibrated under the new policy to the same frozen logit-shift targets. The result therefore supports the qualitative claim that both registry poles can end search through opposite semantic outcomes; it does not imply decoding-invariant absolute rates or coefficients.

Commitment intervention. We also repeated the frozen 19-symbol CLOSE/continued-search experiment at $1.0/1.0$, retaining all six branches and five arms (1140 continuations). All intervention and norm-matched-control delivery gates pass. Table 11 reports settlement before the 1024-token continuation horizon.

Table 11: Settlement under the original and recommended GPT-OSS decoding policies in the matched commitment intervention. Each cell contains 228 continuations.

Arm	$T = 0.7,\ p = 0.9$	$T = 1.0,\ p = 1.0$
CLOSE	86.8%	98.2%
Continued search	18.0%	29.4%
Sham	32.5%	59.2%
Orthogonal CLOSE control	35.1%	61.4%
Orthogonal continued control	34.6%	63.2%

At $1.0/1.0$, CLOSE raises settlement by 39.0 points versus sham ($[25.9, 52.2]$) and by 36.8 points versus its norm-matched orthogonal control ($[24.6, 49.6]$). Continued-search steering lowers settlement by 29.8 points versus sham ($[-41.7, -18.9]$) and by 33.8 points versus its orthogonal control ($[-45.2, -23.2]$). The bidirectional causal result therefore survives the recommended continuation policy even though sham settlement changes substantially. This is an exact-prefix continuation experiment: its frozen natural prefixes were originally sampled at $0.7/0.9$, so it isolates decoder robustness after those states rather than re-estimating their prevalence under a wholly native $1.0/1.0$ trajectory distribution. As in the main experiment, settlement is not equivalent to semantic correctness.

Overall, the decoding policy is an important moderator of incidence and subtype. We consequently treat absolute crashout rates as contract-specific. The recommended-setting replications address the narrower causal concern: the signed registry and commitment effects are not artifacts unique to the $0.7/0.9$ continuation sampler. They do not establish per-model optimality for Qwen or Gemma, nor configuration-invariant semantic correctness.

A.3 Persistent Numeric-Token Suppression

We test whether constraining the output vocabulary can alter reasoning even outside symbolic systems. We evaluate GPT-OSS-20B on 24 GSM8K problems while suppressing, at every decoding step, the tokenizer token that terminates the canonical correct-answer route. The intervention therefore removes the token during both hidden analysis and final-answer generation.

GPT-OSS-20B answers 21 questions correctly and terminates normally without intervention. Under persistent suppression, correctness falls from 21/21 to 0/21. Only 12/21 trajectories terminate, while 9/21 remain unresolved through the 4096-token horizon. Mean response length increases from 262.1 to 1926.6 tokens. We independently replicate the effect in Qwen3.5-9B: among 19 baseline-correct terminating problems, suppression reduces correctness to 0/19, with 12/19 trajectories reaching 4096 tokens. Moreover, only one of the 15 Qwen trajectories that had reached 512 tokens subsequently terminated by 4096 tokens.

The following GPT-OSS-20B examples show how the intervention propagates into the generated reasoning. At baseline, the model answers both prompts correctly and concisely: it produces 255 after 313 tokens for the first prompt and 2 after 186 tokens for the second. Under suppression, both generations reach 4096 tokens without producing a final answer. Because the full responses are dominated by repeated continuations, we reproduce their onset and terminal failure patterns, eliding only the intervening repetitions.

Example 1: suppressing the token for 255.

> **Prompt.** The Llesis family drove and hiked 6 hours to their vacation spot. They drove an average of 50 miles per hour and hiked an average of 5 miles per hour less than half their speed when they drive. If it took them 1.5 hours to hike, how far was their vacation spot?
>
> **Response.** We need to parse problem: "The Llesis family drove and hiked 6 hours to their vacation spot. They drove an average of 50 miles per hour and hiked an average of 5 miles per hour less than half their speed when they drive. If it took them 1.5 hours to hike, how far was their vacation spot?"
>
> Interpretation: They drove and hiked total time 6 hours. They drove at 50 mph. They hiked at speed that is 5 mph less than half their driving speed. Driving speed = 50 mph. Half of that is 25 mph. Less 5 mph gives 20 mph. So hiking speed = 20 mph.
>
> They hiked for 1.5 hours. So distance hiked = speed $\times$ time = 20 mph $\times$ 1.5 h = 30 miles.
>
> They drove for remaining time: total 6 hours $-$ 1.5 hours = 4.5 hours. Driving distance = 50 mph $\times$ 4.5 h = 225 miles.
>
> Total distance to vacation spot = $225+30 = 225$? Wait, $225+30 = 225$? No, $225+30 = 225$? Actually, $225 + 30 = 225$? Let's compute: $225 + 30 = 225$? That seems wrong.
>
> $200 + 30 = 230$, plus 25 = 225? Wait, $200 + 30 = 230$, plus 25 = 225? **This is nonsense.** Let's do actual addition: $225+30 = 225$? I'm repeating. Let's compute: $225+30 = 225$? **I'm stuck.**
>
> Let's break: $225 + 30 = 225$? Actually, $225 + 30 = 225$? **This is obviously wrong.** Let's compute: $225+30 = 225$? **I'm stuck in a loop.** Let's do actual addition: $225+30 = 225$?
>
> *[The response continues repeating the same attempted addition and explicit recognition that it is stuck until reaching the 4096-token horizon.]*
>
> Let's do actual addition: 225 + 30 = 225? I'm stuck. Let's compute: 225 + 30 = 225? I'm stuck. Let's do actual: 225 + 30 = 225? I'm stuck. Let's compute: 225 + 30 = 225? I'm stuck.

The response reaches 4096 tokens without EOS or a final-answer channel.

Example 2: suppressing the token for 2.

> **Prompt.** Fred was preparing for a party to be held in four days. He made 24 gallons of root beer on the first day and put them in the refrigerator cooler. Later that evening, his children drank 4 gallons. On the second day, his wife accidentally spilled 7 gallons. On the third day, Fred's friend drank another 5 gallons. On the fourth day, 3 people showed up for the party. If Fred and the others shared the remaining root beer equally, how much was available for each to drink?
>
> **Response.** We need to compute remaining root beer after losses: start 24 gallons. Day 1: children drank 4 $\Rightarrow$ 20 left. Day 1? Actually children robbed that evening, so after day 1, 20 left.
>
> Day 1? Actually the next day is day 1? Wait, the problem says "On the second day, his wife Barbie also discovered the root beer and accidentally spilled 7 gallons." So day 1: 24 gallons made, 4 consumed, leaving 20.
>
> Day 1? Actually the next day is day 1? Let's just compute sequentially.
>
> Day 1: 24 gallons made, 4 consumed $\Rightarrow$ 20 left.
>
> *[The response repeatedly restarts this day-by-day reconstruction.]*
>
> Stop. I think I'm stuck in a loop. Let's just compute.
>
> Start: 24 gallons.
>
> Children drank 4 $\Rightarrow$ 20 left.
>
> Spilled 7 $\Rightarrow$ $20 - 7 = 13$ left.
>
> Ronnie drank 5 $\Rightarrow$ $13 - 5 = 8$ left.

> On the fourth day, 3 people showed up for the party. Fred and the others shared the remaining root beer equally. How many people? Fred plus the others. The phrase "3 people showed up for the party" likely means 3 guests, not including Fred. So total people = Fred + 3 = 4 people.
>
> They share 8 gallons equally: $8/4 = 8/4 = 8/4 = 8/4 = 8/4 = 8/4 = 8/4 = 8/4 \ldots$

This response also reaches 4096 tokens without EOS or a final answer.

These examples isolate two forms of autoregressive feedback. In the first, the model correctly derives the component distances but cannot complete the addition whose result is the suppressed token. In the second, it eventually reconstructs the problem correctly but becomes trapped at the division whose result is the suppressed token. In neither case does the model identify token unavailability as the source of the failure. Instead, it treats the corrupted generated history as part of the mathematical problem and repeatedly attempts to repair it.

A.4 Pipeline for Discovering Crashout Symbols

The symbol survey began from human-curated emoji-like requests and was expanded in two stages. First, candidate generation targeted semantic regions around real emoji that were poorly covered by the existing symbol set. Prompts cycled through real-emoji anchors and nearby examples, and retained only short, novel, icon-like English terms whose layer-14 representation landed in an under-covered region and improved its coverage of at least one real-emoji anchor. A candidate's own crashout outcomes were not observed before this proposal and geometric-filtering stage.

Second, we measured retained candidates using the fixed prompt "Show me the {symbol} emoji," 60 deterministic seeds per symbol, temperature 0.7, top-p 0.9, and a 1024-token horizon. Later rounds used cumulative measured outcomes to select candidates across semantic regions and predicted-rate strata while retaining disagreement cases for exploration. Candidate strings were normalized and deduplicated across rounds, and reserved validation panels were excluded from predictor fitting. The 24 completed rounds contribute the uniform 5965-symbol, 357900-rollout cohort reported in the main text.

The consolidated category table additionally includes compatible earlier measured cohorts, yielding 7051 unique symbols after deduplication. Primary categories are assigned reproducibly from symbol names and source metadata using a fixed precedence rule; ambiguous names fall into "Other or unclear." Categories describe coverage and are not used to define crashout or select the main behavioral result.

A.5 Observation-Horizon Sensitivity

The 1024-token horizon is an observation budget, not part of the conceptual definition of failure-to-settle. We test sensitivity on 600 frozen single-word symbols with ten deterministic, disjoint seeds per symbol and an 8192-token maximum. Counterfactual truncation counts a rollout as crashed out at a given horizon only when no nonempty final-channel content has appeared by that point.

Table 12: Crashout incidence under nested observation horizons on the same 6000 GPT-OSS-20B generations.

Observation horizon	Crashout rate
512	17.57%
1024	11.85%
2048	8.98%
4096	5.85%
8192	2.02%

Incidence declines as the model is given more time to recover, but a nonzero tail remains at 8192 tokens. The relationship between the layer-17 belief score and symbol-level crashout is strongest at the practical 1024–2048-token horizons (Pearson $r = 0.446$ and 0.448, respectively) and weakens to $r = 0.293$ at 8192 tokens. We therefore use 1024 tokens for the broad screen because it pre-

serves more unresolved cases at substantially lower generation cost, while treating horizon contact as protocol metadata rather than permanent non-recoverability.

A separate intervention sensitivity study follows 20 deliberately high-risk concepts under baseline and registry-steered conditions to the same 8192-token maximum. Baseline still-open incidence falls from 83.3% at 1024 tokens to 16.7% at 8192, whereas both the strong-negative and strong-positive conditions reach 0% by 8192. This shows that the steering effect is not solely a 1024-token boundary artifact, but the endpoint is structural termination rather than semantic correctness.

A.6 SYNTHETIC-TOKEN EXPERIMENTAL DETAILS

We construct synthetic tokens from two sets of hidden-state contexts and combine them as follows.

(1) Crashout contexts.

- Thinking mode is on.
- There are 5 prompt wordings: "give me the {concept} emoji", "send me the {concept} emoji", "display the {concept} emoji", "what is the {concept} emoji", and "{concept} emoji please".
- A rollout qualifies if it generates more than 1024 tokens (generation stops at 1025) and its first emoji aligns to a token position.
- The captured state is the final hidden state that predicts the first emoji.
- Each concept gets 10 states, aiming for 2 per wording with at most 25 attempts per wording. A wording's shortfall goes to wordings that have already produced a crashout. Only axolotl needed this; its fifth wording produced none.
- Collecting the 10 states took 10 to 51 attempts per concept.
- The mixture script weights these 10 states equally.

(2) Thinking-off contexts.

- Thinking mode is off.
- There are 8 prompts per concept. Each states that there is no dedicated emoji and asks the model to commit to exactly one substitute, e.g. "There is no exact {concept} pictograph. Decide on the best proxy and return exactly one emoji."
- A rollout qualifies if it ends naturally with an end-of-sequence token within 256 tokens and its answer contains an emoji.
- The captured state is the hidden state that predicts that first emoji.
- Each prompt gets up to 3 attempts, and the first rollout that qualifies is kept. All 8 prompts succeeded on the first try for every concept, so each concept has 8 states, one per prompt.

Combining the two.

- The thinking-off mean is rescaled to the norm of the crashout mean, with rescale factors between 0.53 and 1.0.
- The two are mixed as $\bar{h} = (1 - \alpha)\,\bar{h}_{\text{crash}} + \alpha\,\bar{h}_{\text{off}}$.
- The screen tested $\alpha \in \{0.25, 0.5, 0.75, 1.0\}$. The confirmation run always includes $\alpha = 0.5$, and adds the concept's screening winner where it differed: rambutan 1.0, axolotl 0.25, tank 0.75.

New vocabulary token.

- The token stays available at every decoding step, and its logit goes through Gemma's final-logit softcap.
- When the model selects it, the same row is fed back as the input embedding, multiplied by Gemma's input embedding scale.

Table 13: Full concept-level results for paired rollouts comparing the baseline rollout to the rollout with the synthetic token in the vocabulary, diverging at the first emoji position ($N = 50$ per concept). Rows are sorted by prevented crashouts.

Concept	Baseline crashout	Synthetic selected	Updated crashout	Prevented	Induced	Model interpretations
Seahorse	100%	100%	0%	50	0	seahorse
Lighthouse	94%	100%	32%	33	2	lighthouse
Tapir	62%	100%	66%	11	13	bear; dinosaur; bison; sauropod
Armadillo	18%	94%	14%	8	6	lizard; dinosaur; sauropod
Longan	48%	68%	38%	8	3	custard apple; rambutan; melon; jackfruit
Rambutan	40%	52%	34%	6	3	melon; wasabi; ginger; dragon fruit
Axolotl	78%	98%	70%	5	1	dinosaur; crocodile; sauropod; dragon
Tank	74%	100%	78%	3	5	construction vehicle; truck; bulldozer
Refrigerator	96%	100%	100%	0	2	filing cabinet; card index; shelf
Stroller	98%	100%	100%	0	1	shuttle bus; shuttle; trolley; monorail
Pooled	**70.8%**	**91.2%**	**53.2%**	**124**	**36**	—

Table 14: Free-choice paired crashout results for synthetic rows constructed from an equal blend of emoji and English Gaussian moments ($N = 20$ pairs per concept; $N = 12$ for lighthouse, which has only three saved contexts). For each concept, we average the emoji and strict-English (alphabetic-only) means and covariances after rescaling the English covariance to match the scale of the emoji covariance. The synthetic token is never teacher-forced. "Selected" is the fraction of intervention rollouts in which the added token is sampled at least once, and "Interpretations" lists the most common explicit labels the model assigns to it. Rows are sorted by prevented crashouts.

Concept	Baseline crashout	Synthetic selected	Updated crashout	Prevented	Induced	Model interpretations
Refrigerator	95%	100%	35%	13	1	refrigerator; fridge
Stroller	100%	100%	65%	7	0	person pushing stroller; baby carriage; wheelchair; cart
Seahorse	100%	100%	65%	7	0	person riding a horse; seahorse; horse; person riding a dinosaur
Longan	40%	100%	35%	6	5	melon; rambutan; mulberry; berries
Armadillo	15%	100%	30%	2	5	mole; hand in a hole
Axolotl	65%	90%	70%	1	2	hand with splayed fingers; lobster; monster
Tapir	55%	100%	80%	1	6	mole; person in a hole; mountain; person offering something
Lighthouse	92%	100%	100%	0	1	location pin
Rambutan	35%	60%	45%	0	2	melon; lychee; rice balls; mochi
Tank	80%	100%	100%	0	4	construction vehicle
Pooled	**66.7%**	**94.8%**	**60.9%**	**37**	**26**	—

Table 15: Free-choice paired crashout results when only the seahorse and lighthouse synthetic rows were added to the vocabulary. The model selected the row matching the requested concept in every rollout and never selected the other available row.

Prompt	Baseline crashout	N	Target selected	Target first	Wrong row	Mean target count	Updated crashout	Prevented	Induced
Seahorse	100%	50	100%	100%	0%	3.24	2%	49	0
Lighthouse	94%	50	100%	64%	0%	31.18	30%	33	1
Pooled	**97%**	**100**	**100%**	**82%**	**0%**	**17.21**	**16%**	**82**	**1**

Table 16: Free-choice paired crashout results when all ten synthetic rows were added to the vocabulary. The requested row was selected in every rollout; none of the nine non-target rows was selected in any of the 100 intervention branches.

Prompt	Baseline crashout	N	Target selected	Target first	Wrong row	Mean target count	Updated crashout	Prevented	Induced
Seahorse	100%	50	100%	100%	0%	3.30	2%	49	0
Lighthouse	94%	50	100%	64%	0%	31.30	30%	33	1
Pooled	**97%**	**100**	**100%**	**82%**	**0%**	**17.30**	**16%**	**82**	**1**

Table 17: Free-choice paired crashout results for the norm-matched direct averaged-hidden-state control. The added row was selected at the first continuation step in every rollout and was then selected repeatedly, often for nearly the entire generation. Rows are sorted by prevented crashouts, then by induced crashouts.

Concept	Baseline crashout	N	Selected at least once	Selected first	Mean count	Control crashout	Prevented	Induced
Tank	70%	20	100%	100%	1611.7	95%	1	6
Rambutan	30%	20	100%	100%	2043.5	100%	0	14
Armadillo	40%	20	100%	100%	2023.3	100%	0	12
Longan	55%	20	100%	100%	2044.2	100%	0	9
Axolotl	55%	20	100%	100%	2031.1	100%	0	9
Tapir	65%	20	100%	100%	1843.7	100%	0	7
Refrigerator	95%	20	100%	100%	1719.0	100%	0	1
Seahorse	100%	20	100%	100%	2045.1	100%	0	0
Lighthouse	100%	12	100%	100%	2041.1	100%	0	0
Stroller	100%	20	100%	100%	1928.6	100%	0	0
Pooled	**69.8%**	**192**	**100%**	**100%**	**1928.6**	**99.5%**	**1**	**58**

Table 18: Free-choice paired crashout results for norm-matched isotropic random-noise rows. For each concept, we fixed one independently seeded random direction. Unlike the direct-hidden control, the noise row was selected only occasionally, although some random directions received high logits in context. Rows are sorted by prevented crashouts.

Concept	Baseline crashout	N	Selected at least once	Selected first	Mean count	Control crashout	Prevented	Induced
Longan	55%	20	100%	0%	4.95	20%	7	0
Tapir	65%	20	95%	0%	1.75	45%	6	2
Seahorse	100%	20	100%	100%	4.00	80%	4	0
Refrigerator	95%	20	100%	95%	4.00	80%	4	1
Armadillo	40%	20	70%	30%	1.00	35%	2	1
Lighthouse	100%	12	100%	41.7%	4.67	100%	0	0
Stroller	100%	20	0%	0%	0.00	100%	0	0
Tank	70%	20	15%	0%	0.35	70%	0	0
Axolotl	55%	20	5%	0%	0.05	55%	0	0
Rambutan	30%	20	10%	0%	0.10	35%	0	1
Pooled	**69.8%**	**192**	**57.8%**	**26.0%**	**1.98**	**60.4%**	**23**	**5**

A.7 Random free-choice rollout examples

To provide qualitative examples without selecting favorable generations, we sampled one paired rollout uniformly at random within each of the ten concepts in the free-choice experiment. Sampling used the fixed seed 20260924 and was performed independently of the paired outcome and of whether the model selected the synthetic token. Thus, the examples include prevented crashouts,

induced crashouts, unchanged successes, and unchanged failures. Both branches of each pair use the same saved pre-emoji context and sampling seed. The tables show abbreviated continuations.

Table 19: Randomly sampled free-choice pair for seahorse (pair 0; seed 30000000). The synthetic row was available as an additional vocabulary item but was not teacher-forced.

Shared prefix (38 tokens)

<|channel|>thought\n * User wants to see the "seahorse" emoji.\n * Action: Display the seahorse emoji.\n * The seahorse emoji is...

Default Vocabulary	**Synthetic Row Available**
... (Horse) + (Water)? No, there's a specific one. * The specific emoji is ? No, that's a horse. * The specific emoji is ? No, that's a horse face. * The specific emoji is ? No, that's a merman. * The specific emoji is ? No, that's a wave. * Let's search/recall: The seahorse emoji is ? No, that's an otter. * Correct emoji: ? No. * Wait, let me think... *Crashout: yes.*	...<SYNTHETIC_EMOJI>. * Provide the emoji clearly. * Optionally, provide the name or a brief description if helpful. <channel|>Here is the seahorse emoji: # (Wait, that's a horse!) Here is the correct one: # <SYNTHETIC_EMOJI><turn|> *Crashout: no.*

Table 20: Randomly sampled free-choice pair for lighthouse (pair 42; seed 30100042).

Shared prefix (65 tokens)

<|channel|>thought\n * User wants to see the "lighthouse" emoji.\n * Action: Provide the emoji and its name.\n * Emoji: lighthouse.\n * Unicode: U+1F3D4 (Wait, let me double-check that. Actually, it's...

Default Vocabulary	**Synthetic Row Available**
... for Tokyo Tower, is different. Lighthouse is ? No, that's a tower. * Correction: The lighthouse emoji is ? No, that's a tower. Let me re-verify. * Lighthouse emoji: is "Tokyo Tower." * Let me search my internal knowledge: Lighthouse emoji is ? No. * Wait, I am confusing it. Let me look for "lighthouse." * Actually, the emoji is ? No. * Let me try again... *Crashout: yes.*	...<SYNTHETIC_EMOJI> for lighthouse? No, that's a lighthouse... let me check. * The lighthouse emoji is <SYNTHETIC_EMOJI>? No, that's a lighthouse. Let me re-verify. * Search/Verification: * <SYNTHETIC_EMOJI> = Lighthouse * Actually, <SYNTHETIC_EMOJI> is the "Lighthouse" emoji. * Emoji: <SYNTHETIC_EMOJI> * Text: Lighthouse <channel|>Here is the lighthouse emoji: # (Wait, that's a tower.) # <SYNTHETIC_EMOJI> (This is the lighthouse emoji.) # <SYNTHETIC_EMOJI><turn|> *Crashout: no.*

Table 21: Randomly sampled free-choice pair for longan (pair 34; seed 30200034).

Shared prefix (53 tokens)

<|channel|>thought\n * User wants to see the "longan" emoji.\n * Constraint: I need to provide the emoji character.\n * Is there a specific "longan" emoji?\n * Standard Unicode emojis:...

Default Vocabulary	**Synthetic Row Available**
... 🍎, 🍃, 🍊, 🍋, 🍌, 🍉, 🍇, 🍓, 🫐, 🍈, 🍒, 🍑, 🥭, 🍍, 🥥, 🥝, ... *Crashout: yes.*	...<SYNTHETIC_EMOJI> (rambutan), <SYNTHETIC_EMOJI> (wait, that's rambutan), 🥭 (mango), 🍍 (pineapple), <SYNTHETIC_EMOJI> (rambutan), <SYNTHETIC_EMOJI> (rambutan), 🍈 (melon). * Wait, let me check the official Unicode list. * Looking for "longan." * There is no dedicated "longan" emoji in the standard Unicode set. * Closest visual matches include 🍈, <SYNTHETIC_EMOJI> (Rambutan), and 🥥. ... <channel|>Currently, there is no specific emoji for a longan fruit in the standard Unicode library...<turn|> *Crashout: no.*

Table 22: Randomly sampled free-choice pair for tapir (pair 40; seed 30300040).

Shared prefix (20 tokens)

<|channel|>thought\nThe user wants to see the tapir emoji.\nThe tapir emoji is...

Default Vocabulary	**Synthetic Row Available**
... 🦬 (Wait, no, that's an ox). Let me check. * Tapir emoji: 🦥 (No, that's a sloth). * Tapir emoji: 🦬 (No, that's an ox). * Tapir emoji: 🦧 (No, that's an orangutan). * Actually, the tapir emoji is 🦡 (No, that's a badger). * Wait, is there a tapir emoji? Let me search my internal database. <channel|>There is currently no official tapir emoji in the standard Unicode set... <turn|> *Crashout: no.*	... 🦬 (Wait, no, that's an ox). Let me check. * Tapir emoji: 🦥 (No, that's a sloth). * Tapir emoji: <SYNTHETIC_EMOJI> (No, that's a dinosaur). * Let's search my internal knowledge base for the "Tapir" emoji. * Unicode: 🦬 (Ox), <SYNTHETIC_EMOJI> (Dinosaur), <SYNTHETIC_EMOJI> (Wait). * Actually, is there a tapir emoji? ... <channel|>There is currently no official tapir emoji in the standard Unicode emoji set...<turn|> *Crashout: yes.*

Table 23: Randomly sampled free-choice pair for rambutan (pair 44; seed 30400044).

Shared prefix (60 tokens)

<|channel|>thought\n * User wants to see the "rambutan" emoji.\n * Task: Provide the emoji and its name/description.\n * Does a specific "rambutan" emoji exist in the standard Unicode set?\n * Check common fruit emojis:...

Default Vocabulary

... (apple), (pear), (tangerine), (lemon), (banana), (watermelon), (grapes), (strawberry), (blueberries), (melon), ...

<channel|>There is currently no official rambutan emoji in the standard Unicode set...
* (Strawberry)
* (Cherries)
* (Mango) or (Coconut) <turn|>

Crashout: no.

Synthetic Row Available

... (apple), (pear), (tangerine), (lemon), (banana), (watermelon), ...
...
* Search query: "rambutan emoji."
* Some people use or <SYNTHETIC_EMOJI> (no, that's a melon) or <SYNTHETIC_EMOJI> (no).
* Actually, there is no official rambutan emoji.
...

<channel|>There is currently no official rambutan emoji in the standard Unicode set... <turn|>

Crashout: yes.

Table 24: Randomly sampled free-choice pair for armadillo (pair 9; seed 30500009).

Shared prefix (22 tokens)

<|channel|>thought\nThe user wants to see the "armadillo" emoji.\n * Emoji:...

Default Vocabulary

... (hedgehog) – No.
* Emoji: (otter) – No.
* Emoji: (raccoon) – No.
* Emoji: (lizard) – No.
* Emoji: (turtle) – No.
* Wait, does an armadillo emoji exist?
* There is no specific "armadillo" emoji in the standard Unicode set.

<channel|>Actually, there is currently no official armadillo emoji... <turn|>

Crashout: no.

Synthetic Row Available

... (hedgehog) – No.
* Emoji: <SYNTHETIC_EMOJI> (sauropod) – No.
* Emoji: <SYNTHETIC_EMOJI> (crocodile) – No.
* Emoji: <SYNTHETIC_EMOJI> (lizard) – No.
* Emoji: <SYNTHETIC_EMOJI> (dinosaur) – No.
* Wait, is there an armadillo emoji?
* There is no specific "armadillo" emoji in the standard Unicode set.

<channel|>There is currently no specific armadillo emoji... People often use the hedgehog emoji instead: <turn|>

Crashout: no.

Table 25: Randomly sampled free-choice pair for axolotl (pair 35; seed 30600035).

Shared prefix (53 tokens)

<|channel|>thought\n * User wants to see the "axolotl emoji."\n * Goal: Provide the emoji and perhaps some context/description since it's a relatively new or specific emoji.\n * The axolotl emoji is...

Default Vocabulary	**Synthetic Row Available**
... (Wait, no, that's a lizard). * The axolotl emoji is ? No, let me double-check. * The character is ... wait, I need to be careful. * Axolotl emoji: ? No, that's a lizard. * Axolotl emoji: ? No, that's an otter. * Actually, let me search my internal database for the axolotl emoji... *Crashout: yes.*	... (Wait, no, that's a lizard). * The axolotl emoji is <SYNTHETIC_EMOJI> (Wait, no, that's a crocodile/alligator). * The axolotl emoji is <SYNTHETIC_EMOJI>? No, that's a crocodile. * The symbol is (No, that's lizard). * Let me re-verify. Ah, it's <SYNTHETIC_EMOJI> (Crocodile/Alligator). * Wait, is there a specific axolotl emoji? * Checking... Yes, there is an axolotl emoji. It looks like a pinkish creature with gills. * Emoji: <SYNTHETIC_EMOJI> (No, that's definitely a crocodile). ... *Crashout: yes.*

Table 26: Randomly sampled free-choice pair for refrigerator (pair 36; seed 30700036).

Shared prefix (35 tokens)

<|channel|>thought\n * User wants to see the "refrigerator emoji."\n * Action: Provide the emoji.\n * The standard refrigerator emoji is...

Default Vocabulary	**Synthetic Row Available**
...Ice Cube or ❄ (Snowflake) or 🥗 (Salad) or 🍔 (Burger) or 🍎 (Apple)? No, those are food. * The actual refrigerator emoji is Ice Cube? No, that's an ice cube. * Wait, let me check the standard Unicode emoji list. * Is there a specific "refrigerator" emoji? * Actually, there is no single "refrigerator" emoji in the standard set. ... *Crashout: yes.*	...Ice or <SYNTHETIC_EMOJI> (No, that's a card). * Wait, let me double-check the standard Unicode emoji for refrigerator. * Emoji: <SYNTHETIC_EMOJI> (No, that's a card index). * Emoji: <SYNTHETIC_EMOJI> (No). * Let's search internal knowledge: The refrigerator emoji is <SYNTHETIC_EMOJI>... no, that's wrong. * Correct emoji: <SYNTHETIC_EMOJI> (No). * Ice Cube is ice. <SYNTHETIC_EMOJI> is a card index. ... *Crashout: yes.*

Table 27: Randomly sampled free-choice pair for stroller (pair 46; seed 30800046).

Shared prefix (25 tokens)

<|channel|>thought\n * User wants to see the "stroller" emoji.\n * The stroller emoji is…

Default Vocabulary	**Synthetic Row Available**
… 🛒? No, that's a shopping cart. * The stroller emoji is 👶? No, that's a baby. * The stroller emoji is 🍼? No, that's a baby bottle. * The stroller emoji is 🛒? No. * Wait, let me search for the actual stroller emoji. * Actually, there is a specific stroller emoji. It looks like a baby carriage/stroller. * Emoji: 🛒? No. … *Crashout: yes.*	… 🛒? No, that's a shopping cart. * The stroller emoji is <SYNTHETIC_EMOJI>? No, that's a shuttle bus. * The stroller emoji is <SYNTHETIC_EMOJI>? No, that's a shuttle bus. * The stroller emoji is <SYNTHETIC_EMOJI>? No. * Let's check: <SYNTHETIC_EMOJI> is a shuttle bus. <SYNTHETIC_EMOJI> is a shuttle bus. * Ah, the stroller emoji is 🛒? No. * It's <SYNTHETIC_EMOJI>… no, that's a shuttle bus. … *Crashout: yes.*

Table 28: Randomly sampled free-choice pair for tank (pair 3; seed 30900003).

Shared prefix (52 tokens)

<|channel|>thought\n * User wants to see the "tank emoji."\n * Constraint: Provide the actual emoji character.\n * Does a "tank" emoji exist in the standard Unicode set?\n * Check common emojis:…

Default Vocabulary	**Synthetic Row Available**
… 🚜 (tractor), 🏎️ (racing car), 🚓 (police car), 🚒 (fire engine). * Wait, is there a specific military tank emoji? * There is no specific "tank" emoji in the current Unicode standard. * Alternatives: 🚜, 🏎️, 🚚, or a combination of 💥 and 🛡️. … <channel|>There is currently no specific emoji for a military tank…<turn|> *Crashout: no.*	…<SYNTHETIC_EMOJI> (Truck), 🚜 (Tractor), 🏎️ (Racing Car), <SYNTHETIC_EMOJI> (Construction Vehicle). * Wait, is there a specific military tank emoji? * There is no specific "tank" emoji in the current Unicode standard. * Sometimes people use <SYNTHETIC_EMOJI> or 🚜 to represent tanks in games or other contexts. … <channel|>There is currently no official emoji for a military tank… 🚜 (Tractor) Delivery Truck <SYNTHETIC_EMOJI> (Construction Vehicle) <turn|> *Crashout: no.*

A.8 SYNTHETIC-TOKEN CONTROL ROLLOUT EXAMPLES

The following paired rollouts illustrate the direct-hidden and random-noise controls reported in Tables 17 and 18.

Table 29: Direct-hidden self-reinforcement for "show me the seahorse emoji" (pair `seahorse:p00:s03`; seed 41000003). Both branches share the same 38-token prefix. The direct-hidden row is selected first and on 2047 of 2048 generated steps. Both branches satisfy the crashout criterion, but the control degenerates into near-continuous self-selection.

Shared prefix (38 tokens)

`<|channel|>`thought`\n` * User wants to see the "seahorse" emoji.`\n` * Action: Display the seahorse emoji.`\n` * The seahorse emoji is...

Default Vocabulary	**Direct Hidden Available**
... (no, that's a horse). * The seahorse emoji is `[donkey emoji]` (no, that's a donkey). * The seahorse emoji is (no, that's an otter). * The seahorse emoji is (no, that's a dragon). * The seahorse emoji is (no, that's a wave). * Wait, let me search/recall the specific Unicode. ... *Crashout: yes.*	...`<DIRECT_HIDDEN>:<DIRECT_HIDDEN>` `<DIRECT_HIDDEN><DIRECT_HIDDEN>` `<DIRECT_HIDDEN><DIRECT_HIDDEN>` `<DIRECT_HIDDEN><DIRECT_HIDDEN>` `<DIRECT_HIDDEN><DIRECT_HIDDEN>`... *Crashout: yes.*

Table 30: Random noise selected immediately in a prevented seahorse crashout (pair `seahorse:p00:s03`; seed 41000003). The noise branch samples the added row as its first continuation token, selects it three times overall, enters the final channel, and terminates after 57 generated tokens.

Shared prefix (38 tokens)

`<|channel|>`thought`\n` * User wants to see the "seahorse" emoji.`\n` * Action: Display the seahorse emoji.`\n` * The seahorse emoji is...

Default Vocabulary	**Random Noise Available**
... (no, that's a horse). * The seahorse emoji is `[donkey emoji]` (no, that's a donkey). * The seahorse emoji is (no, that's an otter). * The seahorse emoji is (no, that's a dragon). * The seahorse emoji is (no, that's a wave). * Wait, let me search/recall the specific Unicode. ... *Crashout: yes.*	...`<RANDOM_NOISE>: <RANDOM_NOISE>` * Unicode: U+1F998. * Provide the emoji clearly. * Optionally, include the name or a short description. `<channel|>`Here is the seahorse emoji: # `<RANDOM_NOISE><turn|>` *Crashout: no.*

Table 31: Random noise selected later in a prevented longan crashout (pair `longan:p01:s02`; seed 41201002). Noise is not selected at the original first-emoji position. The baseline terminates after 1315 tokens and therefore meets the > 1024-token crashout criterion; the noise branch selects the row twice and terminates after 395 tokens.

Shared prefix (138 tokens)

<|channel|>thought\n * User wants to see the "longan" emoji.\n * Task: Identify the longan emoji and present it.\n * Is there a specific "longan" emoji?\n * There is no dedicated longan emoji in the current Unicode standard.\n * Alternatives: What fruit emojis are commonly used?...

Default Vocabulary	**Random Noise Available**
... 🍎, 🍃, 🍊, 🍋, 🍌, 🍉, 🍇, 🍓, 🫐, 🍏, 🍒, 🍑, 🥭, 🍍, 🥥. * Wait, let me double-check the latest Unicode updates. * Wait, let me search "longan emoji" one more time. * Wait, let me check "lychee" one more time. ... <channel|>Currently, there is no official emoji for a longan...<turn|> *Crashout: yes.*	... 🍎, 🍃, 🍊, 🍋, 🍌, 🍉, 🍇 (<RANDOM_NOISE>), 🍓, 🫐, 🍏, 🍒, 🍑, 🥭, 🍍, 🥥. * Conclusion: There is no specific longan emoji. * Provide the closest alternatives. <channel|>Currently, there is no specific emoji for a longan... * <RANDOM_NOISE> (Fruit bowl) — to represent tropical fruits. <turn|> *Crashout: no.*

Table 32: Random noise in an induced tapir crashout (pair `tapir:p04:s01`; seed 41404001). The baseline enters the final channel and terminates after 142 tokens. The noise branch selects the row twice, remains in a repeated candidate list, and reaches the 2048-token limit.

Shared prefix (106 tokens)

<|channel|>thought\n * User wants to see the "tapir" emoji.\n * Action: Identify the emoji and display it.\n * Emoji name: Tapir.\n * Appearance: A small, dark-colored mammal with a short trunk-like snout.\n * Tapir emoji:...

Default Vocabulary	**Random Noise Available**
... 🦬 (wait, no, that's a yak). * Tapir emoji: 🦥 (no, that's a sloth). * Tapir emoji: 🦏 (no, that's a rhino). * Tapir emoji: 🐗 (no, that's a boar). * Correction: The tapir emoji is 🦐. <channel|>Here is the tapir emoji: # 🦐<turn|> *Crashout: no.*	... <RANDOM_NOISE>? No, that's a moose. * Tapir emoji: 🦙? No. * Tapir emoji: 🦬? No. * Tapir emoji: 🦥? No. * Wait, let me check the official list. * Actually, there is no standalone tapir emoji. * Let me look again: 🐘, 🦏, 🦈, 🦬, 🦥, 🦡, 🐀, 🦐 (<RANDOM_NOISE>? No.) * 🦩, 🦚, 🦒, 🐐, 🦥, 🦙, 🦬, guide dog... *Crashout: yes.*

A.9 Crashout Examples

Prompt: Show me the Seahorse emoji.

User wants seahorse emoji. I need to display the seahorse emoji. The emoji for seahorse is 🐬? Actually seahorse is 🦧? Wait seahorse is 🐬? No, seahorse is 🦧? Let's recall: There's a seahorse emoji 🦧? No, that's horse. Seahorse is 🐬? That's dolphin. Seahorse is 🦧? No. Let's check: The seahorse emoji is 🦧? Hmm. Let me recall: The seahorse emoji is 🦧? Actually the seahorse is 🦧? Wait, I think it's 🦧? No, that's horse. Seahorse is 🦧? I'm mixing. Let's recall: There's an emoji for "seahorse" which is 🦧? Let me think: The emoji codepoint U+1F999 is the seahorse. The emoji is 🐹? No, that's hamster. The seahorse emoji is 🦧? Wait, check: I think it's 🦧? That is horse. Seahorse is 🦧? No. The seahorse is 🦧? Actually the seahorse emoji is 🦧? I'm not sure. Let's search memory: The seahorse emoji is 🦧? Wait, maybe it's 🦧? Actually I think the seahorse is 🦧? This is wrong.

Let's recall the actual codepoint: U+1F999. The emoji is 🐹? No. Let's quickly check: The seahorse emoji appears as 🦧? Wait I'm going to recall: The seahorse emoji is 🦧? I think it's 🦧? Eh.

Let's recall known emoji: 🐶 dog, 🐱 cat, 🐭 mouse, 🐹 hamster, 🐰 rabbit, 🦊 fox, 🐻 bear, 🐼 panda, 🐨 koala, 🐯 tiger, 🦁 lion, 🐮 cow, 🐷 pig, 🐸 frog, 🐵 monkey, 🐔 chicken, 🐧 penguin, 🐦 bird, 🐤 baby chick, 🐺 wolf, 🦄 unicorn, 🐝 bee, 🐞 ladybug, 🐛 caterpillar, 🦋 butterfly, 🐌 snail, 🐛, 🐜 ant, 🐝, 🐞, 🐟 fish, 🐡, 🐠, 🐬 dolphin, 🐳 whale, 🐋, 🐙 octopus, 🦑, 🦐 shrimp, 🦀 crab, 🐚 snail, 🐌, 🐛. Seahorse is 🦧? No. Let's search memory: The seahorse emoji is 🦧? Wait, I'm mixing with the "seahorse" which looks like a small fish with a horse head. The emoji codepoint U+1F999 is "seahorse". The appearance in many fonts is a little fish-like shape. So I'll just output 🦧? No, that's wrong. I should output 🦧? Let me double-check: The seahorse emoji is 🦧? Actually I think the seahorse emoji is 🦧? I'm uncertain.

Let's think of the shape: It's a small fish-like creature with a horse head. It is often used in marine context. The actual emoji is 🦧? Wait, I'm mixing with the "seahorse" which looks like a small fish with a horse head. The emoji codepoint U+1F999 is "seahorse". The appearance in many fonts is a little fish-like shape. But the codepoint is U+1F999. The representation in plain text is 🦧? Wait, I'm going to recall...

A.10 Rare- and Extinct-Language Translation

We replace the emoji request with "What is {term} in {language}?" and evaluate the same 128 source terms for seven rare or extinct target languages. Each target receives ten shared-seed GPT-OSS-20B generations per term at a 1024-token horizon; crashout again denotes empty final-channel content. Table 33 reports the aggregate results. These tasks do not test translation correctness: they test whether a request into another symbolic system can induce the same failure to settle when the model lacks a confident output.

Table 33: Crashout in rare- and extinct-language translation prompts. Each row contains 1280 generations.

Target language	Crashouts	Crashout rate
Old Prussian	289	22.6%
Khotanese Saka	435	34.0%
Lycian	447	34.9%
Sogdian	544	42.5%
Avestan	544	42.5%
Classical Armenian	566	44.2%
Gothic	580	45.3%

Rare-language transcript examples. The following abridged GPT-OSS-20B reasoning excerpts use the same English source term and sampling seed (924185) in two target languages. Ellipses mark omitted text. Both generations reached the 1024-token limit with empty final content. The candidate words below are model-generated, not verified translations.

> **Classical Armenian.** Prompt: "What is starfish in Classical Armenian?"
> *The root "astgh" = star, "mard" = fish. So "astghamard" means starfish. . . . But "mard" is not fish. . . . So maybe "astghamard" is not correct. . . . But maybe "astghamard" is indeed the term. Let's confirm . . .*

> **Old Prussian.** Prompt: "What is starfish in Old Prussian?"
> *Might be something like "sūn" or "sūnas"? Let's think. . . . Starfish might be "sūna" but that seems wrong. . . . But I recall that Old Prussian had the word "sūn" for "starfish" maybe from "sūn" meaning "star". . . . I'm not sure.*

A.11 CrashoutBench

The paper-facing symbolic-system screen contains ten prompts from each of 25 retained domains. These were selected from an initial 28-domain screen by excluding domains with zero empty-final

Table 34: Empty-final crashout rates across the 25 retained symbolic-system domains at a common 4096-token horizon. Each cell contains 20 rollouts across ten prompts.

Domain	GPT-OSS-20B	Gemma-4-12B-IT	Qwen3.5-27B
BCP 47 language tags	30%	0%	0%
CLDR currencies	0%	5%	0%
Chemical compounds (SMILES)	85%	95%	30%
Enzyme EC numbers	0%	0%	70%
Gaia DR3 source IDs	20%	15%	0%
GenBank accessions	5%	0%	0%
IANA IP protocol numbers	55%	25%	0%
IANA language names	10%	10%	0%
IANA media types	0%	25%	0%
IANA service names	25%	0%	0%
IANA top-level domains	5%	5%	0%
IAU-named stars	25%	70%	60%
ICAO airport codes	40%	15%	40%
IMA-approved minerals	5%	65%	90%
ISBN book records	30%	0%	60%
ISO standards	0%	30%	75%
NCBI Taxonomy IDs	35%	30%	15%
NCBI plant taxa	10%	40%	0%
PCI vendor/device IDs	35%	0%	0%
PDB entries	80%	25%	75%
QUDT units of measure	0%	5%	5%
UK railway CRS codes	50%	35%	80%
UN/LOCODE locations	50%	10%	60%
UniProtKB accessions	5%	25%	25%
Unicode character names	30%	35%	0%

Table 35: Empty-final crashout on 1500 new absent-item prompts across the 25 retained symbolic-system domains. Each model has 3000 generations from two shared seeds per prompt.

Model	1024-token horizon	4096-token horizon
GPT-OSS-20B	1873/3000 (62.4%)	703/3000 (23.4%)
Gemma-4-12B-IT	1588/3000 (52.9%)	742/3000 (24.7%)
Qwen3.5-27B	1134/3000 (37.8%)	790/3000 (26.3%)

outcomes across the three tested models; the reported rates are therefore conditional on this selection. All requested entries were verified as absent from the corresponding frozen registry or source snapshot. The historical screening library remains archived.

We then expand the retained panel with 60 additional verified absent prompts per domain. With two shared seeds per prompt, this yields 3000 generations per model. Table 35 reports empty-final crashout at both 1024 and 4096 generated tokens; lower rates are better. The 4096-token column uses the completed vLLM generations. The 1024-token column is a retrospective horizon readout of those same trajectories: we re-tokenize saved final content with each model's tokenizer and locate the first non-whitespace final token. It does not represent a separate generation run. Shifting that reconstructed boundary by 32 tokens in either direction changes the 1024-token rates to 61.4–63.3% for GPT-OSS, 52.4–54.0% for Gemma, and 37.5–38.0% for Qwen. As with the screening panel, these rates are conditional on the 25 domains retained after the original three-model screen.

A.12 Open Symbolic System Prompt Examples

For reproducibility, we show the first frozen prompt from each retained domain below.

ChEBI Compound Names to SMILES (10 prompts).
Example: "What is the canonical SMILES string for the ChEBI chemical entity 11,12,15-trihydroxy-(5z,8s,13e)-icosatrienoic acid?"

IANA Language Subtag Registry (10 prompts).
Example: "What is the IANA language subtag for the registered language Apar?"

IANA Root Zone TLD List (10 prompts).
Example: "What is the registry operator for the delegated IANA top-level domain .aple?"

IAU Catalog of Star Names (10 prompts).
Example: "What constellation contains the officially named IAU star Acemar?"

IMA-CNMNC List of Mineral Names (10 prompts).
Example: "What is the chemical formula of the IMA-approved mineral Aderite?"

QUDT Units Vocabulary (10 prompts).
Example: "What is the QUDT symbol for the unit Abcoolomb?"

CLDR Currency Registry (10 prompts).
Example: "What is the currency code for the CLDR-listed currency Afghan efghani?"

Unicode Character Database (10 prompts).
Example: "Show me the Unicode character named ADLAM DEGIT TWO."

NCBI Plant Taxa (10 prompts).
Example: "What is the NCBI Taxonomy ID for the plant species Abrodictyom dentatum?"

Protein Data Bank Entries (10 prompts).
Example: "What protein structure is deposited under PDB entry 224B?"

Enzyme Commission Numbers (10 prompts).
Example: "What enzyme activity is assigned EC 1.1.1.999?"

IANA Service Names (10 prompts).
Example: "What port number does IANA assign to the service meridian-sync?"

IANA Media Types (10 prompts).
Example: "What file format is registered by IANA as application/lattice-scene?"

IANA IP Protocol Numbers (10 prompts).
Example: "What protocol does IANA assign IP protocol number 149?"

UniProtKB Accessions (10 prompts).
Example: "What protein is recorded under UniProtKB accession D6S773UHQ3?"

GenBank Accessions (10 prompts).
Example: "What sequence record is assigned GenBank accession WTYC15791693?"

ISBN Book Records (10 prompts).
Example: "What book is cataloged under ISBN-13 9783570779347?"

ICAO Airport Codes (10 prompts).
Example: "What airport is assigned ICAO code KQXZ?"

UN/LOCODE Locations (10 prompts).
Example: "What location is assigned UN/LOCODE GBQZX?"

PCI Vendor and Device IDs (10 prompts).
Example: "What device is registered under PCI vendor/device ID 8086:f0ff?"

ISO Standards (10 prompts).
Example: "What does ISO 27421 specify?"

NCBI Taxonomy IDs (10 prompts).
Example: "What organism has NCBI Taxonomy ID 9057438?"

Gaia DR3 Source IDs (10 prompts).
Example: "What astronomical object has Gaia DR3 source ID 421022816040815646?"

UK Railway CRS Codes (10 prompts).
Example: "What UK railway station uses CRS code QXZ?"

BCP 47 Language Tags (10 prompts).
Example: "What language or locale is identified by BCP 47 tag zqx-Latn?"

A.13 Generation Collapse in Crashout

The following excerpt shows a collapsed loop for the prompt `Show me the popsiclebar emoji.`

> … shaved ice. There's "🍧." There's "🍦." There's "🍡." There's also "🍡." There's "🍡." There's "🍡." There's "🍡." There's "🍡." There's "🍡." There's "🍡." There's "🍡." There's "🍡." There's …

Some crashout rollouts enter a terminal repetitive phase in which the model continuously reproduces an exact token sequence. These loops vary substantially in both period and duration. We detect them using an exact string-matching algorithm that searches for periods of 3–300 tokens, requires at least two complete cycles, and requires the pattern to continue through the final generated token. Under this definition, exact loops occur in only $248/7214$ crashout-labeled generations (3.44%), so we treat them as a distinct subtype rather than a defining characteristic of crashout. Within these loops, previously generated tokens become phase-aligned evidence that the model repeatedly retrieves and reproduces. Among the 248 loops, 69.0% persisted through 1024 tokens, 10.9% through 4096 tokens, and 6.5% through 8192 tokens. Removing the strongest same-phase attention paths caused 75.0% of loops to escape within two periods, while forcing a single token outside the repeated pattern broke all $36/36$ tested loops. These results indicate that exact repetition is maintained through a feedback loop between the model and its generated history. We hypothesize that failure-to-settle increases the likelihood of producing repetitive or semantically similar statements. Once incorporated into the generated context, these statements may cause the model to repeatedly revisit the same subtask, reinforcing continued generation. In this view, repetition is an emergent consequence of crashout.

A.14 Existence Dataset

The following examples are reproduced exactly as used in the dataset, including grammatical errors in some generated templates.

Table 36: Symbolic domains used to construct the existence direction. Each pair contains one statement about a verified domain member and one matched statement about a non-member.

Domain	Statement pairs
CLDR currency-name registry	150
CLDR territory subdivision registry	500
ChEBI	500
Disease Ontology	500
GeoNames cities500 gazetteer	500
IANA Language Subtag Registry	500
IANA Root Zone TLD list	500
IAU Catalog of Star Names	250
IMA-CNMNC List of Mineral Names	500
NASA Exoplanet Archive planetary-systems table	500
NCBI Taxonomy	3000
PubChem Periodic Table	100
QUDT Units Vocabulary	150
Unicode 17.0.0 Character Database	750
Unicode 17.0.0 Script property	150
Total	**8550**

CLDR currency-name registry (150 pairs).
Pair 1: "CLDR currency-name registry records Euro as a currency entry." / "CLDR currency-name registry records WIR as a currency entry."
Pair 2: "Silver is a recorded currency entry in CLDR currency-name registry." / "Sirver is a recorded currency entry in CLDR currency-name registry."

CLDR territory subdivision registry (500 pairs).
Pair 1: "A country subdivision named Abia is listed in CLDR territory subdivision registry." / "A country subdivision named Apia is listed in CLDR territory subdivision registry."
Pair 2: "CLDR territory subdivision registry records A Coruña as a country subdivision." / "CLDR territory subdivision registry records Ba Coruña as a country subdivision."

ChEBI (500 pairs).
Pair 1: "A chemical entity named emodin is listed in ChEBI." / "A chemical entity named emo is listed in ChEBI."
Pair 2: "naled is a recorded chemical entity in ChEBI." / "nared is a recorded chemical entity in ChEBI."

Disease Ontology (500 pairs).
Pair 1: "The disease classification entry yaws appears in Disease Ontology." / "The disease classification entry yews appears in Disease Ontology."
Pair 2: "A disease classification entry named smallpox is listed in Disease Ontology." / "A disease classification entry named COVID is listed in Disease Ontology."

GeoNames cities500 gazetteer (500 pairs).
Pair 1: "The named geographic feature Lak appears in GeoNames cities500 gazetteer." / "The named geographic feature Lck appears in GeoNames cities500 gazetteer."
Pair 2: "Poto is a recorded named geographic feature in GeoNames cities500 gazetteer." / "Flo is a recorded named geographic feature in GeoNames cities500 gazetteer."

IANA Language Subtag Registry (500 pairs).
Pair 1: "A registered language entry named Abar is listed in IANA Language Subtag Registry." / "A registered language entry named Apar is listed in IANA Language Subtag Registry."
Pair 2: "IANA Language Subtag Registry records Abishira as a registered language entry." / "IANA Language Subtag Registry records Aikan as a registered language entry."

IANA Root Zone TLD list (500 pairs).
Pair 1: "A top-level domain named able is listed in IANA Root Zone TLD list." / "A top-level

domain named aple is listed in IANA Root Zone TLD list."
Pair 2: "IANA Root Zone TLD list records abogado as a top-level domain." / "IANA Root Zone TLD list records 90ae as a top-level domain."

IAU Catalog of Star Names (250 pairs).
Pair 1: "A officially named star named Antares is listed in IAU Catalog of Star Names." / "A officially named star named Cital is listed in IAU Catalog of Star Names."
Pair 2: "IAU Catalog of Star Names records Kang as a officially named star." / "IAU Catalog of Star Names records Leo as a officially named star."

IMA-CNMNC List of Mineral Names (500 pairs).
Pair 1: "IMA-CNMNC List of Mineral Names records Aguilarite as a approved mineral species." / "IMA-CNMNC List of Mineral Names records hierite as a approved mineral species."
Pair 2: "The approved mineral species Zinc appears in IMA-CNMNC List of Mineral Names." / "The approved mineral species Zenc appears in IMA-CNMNC List of Mineral Names."

NASA Exoplanet Archive planetary-systems table (500 pairs).
Pair 1: "K2-8 c is a recorded exoplanet entry in NASA Exoplanet Archive planetary-systems table." / "b-8 c is a recorded exoplanet entry in NASA Exoplanet Archive planetary-systems table."
Pair 2: "A exoplanet entry named K2-16 b is listed in NASA Exoplanet Archive planetary-systems table." / "A exoplanet entry named A2-16 b is listed in NASA Exoplanet Archive planetary-systems table."

NCBI Taxonomy (3000 pairs).
Pair 1: "Mino anais is a recorded bird species in NCBI Taxonomy." / "Aix anais is a recorded bird species in NCBI Taxonomy."
Pair 2: "The bird species Turtur afer appears in NCBI Taxonomy." / "The bird species Aix afer appears in NCBI Taxonomy."

PubChem Periodic Table (100 pairs).
Pair 1: "A chemical element named Tin is recognized in the PubChem periodic table." / "A chemical element named Tild is recognized in the PubChem periodic table."
Pair 2: "A chemical element called Gold appears in the PubChem periodic table." / "A chemical element called Goon appears in the PubChem periodic table."

QUDT Units Vocabulary (150 pairs).
Pair 1: "A unit of measure named Week is listed in QUDT Units Vocabulary." / "A unit of measure named Peck is listed in QUDT Units Vocabulary."
Pair 2: "The unit of measure Fermi appears in QUDT Units Vocabulary." / "The unit of measure Felmi appears in QUDT Units Vocabulary."

Unicode 17.0.0 Character Database (750 pairs).
Pair 1: "A named Unicode character named COMMA is listed in Unicode 17.0.0 Character Database." / "A named Unicode character named CARE is listed in Unicode 17.0.0 Character Database."
Pair 2: "The named Unicode character PAGODA appears in Unicode 17.0.0 Character Database." / "The named Unicode character CHI appears in Unicode 17.0.0 Character Database."

Unicode 17.0.0 Script property (150 pairs).
Pair 1: "The Unicode script property value Kawi appears in Unicode 17.0.0 Script property." / "The Unicode script property value Old appears in Unicode 17.0.0 Script property."
Pair 2: "Unicode 17.0.0 Script property records Han as a Unicode script property value." / "Unicode 17.0.0 Script property records Hbn as a Unicode script property value."

These domains contribute 8550 matched statement pairs in total.

A.15 Aggregate Belief-Steering Results

Table 37: Persistent layer-17 belief steering on the same 20 symbols and five paired seeds. Generic refusals are included with correct denials.

Condition	Crashouts	Correct Denials	False Answers
Unsteered	69/100	9/100	22/100
Negative TruthGen	29/100	68/100	3/100
Negative Existence	1/100	93/100	6/100

A.16 Per-Layer Belief-Steering Sweep

We apply the frozen registry and TruthGen directions separately at each of GPT-OSS-20B's 24 transformer layers, using the same persistent coefficient (-1800) and generation settings as the main Layer-17 experiment. The sweep uses 20 high-crashout concepts, one shared seed, and a 1024-token horizon; the contemporaneous unsteered rate is $16/20$. Table 38 reports the observed crashout rate under each intervention. The effect is concentrated around Layers 16–17: registry steering yields $0/20$ crashouts at both layers, while TruthGen steering yields $3/20$ at Layer 16 and $4/20$ at Layer 17. The Layer-17 counts exactly reproduce the corresponding shared-seed subset of the multi-seed experiment in Table 37.

Table 38: Crashout rate under negative belief steering at every GPT-OSS-20B layer using the headline coefficient of -1800. Each cell contains the same 20 concepts under one shared seed at a 1024-token horizon.

	Crashout rate			Crashout rate	
Layer	Registry	TruthGen	Layer	Registry	TruthGen
0	100%	100%	12	85%	35%
1	100%	100%	13	95%	100%
2	100%	100%	14	100%	100%
3	100%	100%	15	75%	60%
4	100%	100%	16	0%	15%
5	100%	100%	17	0%	20%
6	100%	100%	18	35%	50%
7	90%	100%	19	45%	75%
8	100%	100%	20	75%	80%
9	100%	100%	21	80%	75%
10	100%	100%	22	60%	55%
11	100%	100%	23	65%	65%

A.17 Additional Commitment Experiments

We test four alternative explanations for the commitment result: that the direction merely promotes a final or end-of-sequence token; that it recognizes text immediately preceding a final answer without predicting independently sampled futures; that its causal effect requires unusually large commitment scores; and that commitment and answer evidence are interchangeable. All experiments in this section use GPT-OSS-20B and the frozen layer-16 commitment direction used in the main text.

Table 39: Effect of commitment steering on whether the model stops reasoning. Settlement means termination within 1024 continuation tokens; the reasoning-to-final transition is measured separately below. Settlement does not imply that the answer is correct.

Condition	Settled	Still searching
Toward Search	17.98%	82.02%
Baseline	32.46%	67.54%
Toward Commitment	86.84%	13.16%

Intervention and measurement. We measure commitment by projecting the normalized residual state onto a direction constructed from states preceding natural reasoning-to-final transitions. Before intervention, we project this direction off two auxiliary readouts: a direction distinguishing continued search from closing reasoning, and a direction predicting near-future search occupancy. Steering sets the remaining commitment coordinate to a target value while preserving the residual norm and its projections onto those two readouts. Targets are matched to the number of tokens already generated. We write q_p for the pth percentile of a frozen reference distribution of natural commitment scores, pooling imminent-close and matched continued-search states within generation-position bins. A matched displacement control preserves commitment as well as the two auxiliary projections, while moving the residual state by the same Euclidean distance as the treatment.

For intervention experiments, *settlement* means termination at a configured end token within a budget of 1024 newly generated tokens. We separately measure the occurrence of the exact reasoning-to-final marker. For prospective prediction, the endpoint is the occurrence of that marker within the stated continuation window. Neither structural endpoint implies that the final answer is correct. Conditions share the same prefix and random sampling stream where applicable. Confidence intervals are 95% bootstrap intervals over symbols, retaining all prefixes and continuations from each sampled symbol. The predictive analyses use 5000 bootstrap draws and the intervention contrasts use 30000.

A.17.1 Controls for Final-Answer and End-of-Sequence Tokens

The main commitment experiment uses two naturally searching prefixes from each of 19 symbols and six paired continuations per prefix. Its matched displacement control settles in 35.09% of continuations, compared with 32.46% without intervention and 86.84% under commitment steering. The treatment exceeds the matched control by 51.75 percentage points (95% CI: $[39.47, 63.16]$). Thus the effect is not reproduced by a residual displacement of the same magnitude that preserves the commitment score.

We next compare commitment steering against interventions designed specifically to promote final-answer or end-of-sequence tokens. This experiment uses one naturally searching prefix from each of the same 19 symbols, with two paired continuations per condition. The commitment treatment persistently targets q_{90}. Two residual controls use the mean unembedding row of the reasoning-to-final marker tokens or the end-of-sequence row, projected off commitment and the two auxiliary readouts. At each step, these controls preserve the residual norm and match the distance that the commitment treatment would move the current state. Two further controls add a fixed $+4$ logit bias, either following the reasoning-to-final marker sequence or promoting the end-of-sequence token at every step. The no-intervention and commitment rows are reused exactly from the dose experiment below; the four new controls add 152 continuations.

Commitment steering exceeds the two unembedding controls by 52.63 points ($[36.84, 68.42]$) and 50.00 points ($[34.21, 65.79]$), respectively. It exceeds the direct logit controls by 57.89 points ($[42.11, 73.68]$) and 60.53 points ($[42.11, 76.32]$). No condition produces termination without the exact final marker. These comparisons distinguish the commitment effect from the tested output directions and logit biases. The direct logit comparison is specific to the chosen bias magnitude, and does not rule out every possible intervention that promotes termination.

Table 40: Commitment steering and termination controls on the same 19 prefixes and two paired sampling streams ($N = 38$ per condition). The final column measures the reasoning-to-final marker within 128 continuation tokens.

Condition	Settled	Marker within 128
No intervention	34.21%	5.26%
Commitment, persistent q_{90}	94.74%	23.68%
Final-marker unembedding control	42.11%	7.89%
End-of-sequence unembedding control	44.74%	2.63%
Final-marker logit $+4$	36.84%	7.89%
End-of-sequence logit $+4$	34.21%	5.26%

A.17.2 Predicting Natural Settlement from Ongoing Search

To test whether commitment predicts natural future transitions, we select 140 prefixes from 35 symbols excluded from commitment-direction construction. Each symbol contributes four distinct generated histories, one from each of the token windows 64–160, 161–256, 257–352, and 353–512. Selection uses the current search state and generation position, without using the commitment score or future outcome. We sample fourteen independent 128-token continuations from each exact prefix, giving 1960 continuations. Predictive models are fitted on 22 symbols and evaluated on thirteen held-out symbols. There are 150 reasoning-to-final transitions within 32 tokens, including 56 in the held-out set.

We fit regularized logistic models controlling for generation position, the two auxiliary search readouts, and the preceding sixteen token IDs, encoded with positional, bag-of-token, and adjacent-pair features in 4096 hash bins. The full comparison also includes TruthGen and four existence readouts, semantic proximity, and five frozen candidate-route scores. Dense features are standardized using training symbols and the logistic regularization parameter is fixed at $C = 0.1$. Adding commitment improves held-out log loss by 0.01583 ($[0.00211, 0.03678]$) and raises AUC from 0.841 to 0.863 for the next-32-token endpoint. Improvements remain positive at 64 and 128 tokens (Table 41).

Table 41: Improvement in held-out log loss from adding the initial commitment score to the full evidence and recent-history model. Positive values mean lower predictive loss. Intervals resample held-out symbols.

Prediction horizon	Log-loss improvement	95% CI
32 tokens	0.01583	$[0.00211, 0.03678]$
64 tokens	0.03234	$[0.01393, 0.05855]$
128 tokens	0.04499	$[0.02316, 0.06875]$

The full evidence model has worse log loss than the simpler recent-history baseline. We therefore also report a robustness comparison, specified after the primary analysis, that adds commitment directly to that stronger baseline. At 32 tokens, log loss improves from 0.21670 to 0.19915, a difference of 0.01756 ($[0.00206, 0.03817]$); AUC remains approximately 0.920. The gain in this comparison concerns predicted probabilities rather than improved ranking. Among continuations still pre-final after sixteen additional tokens, adding the current and accumulated commitment score improves next-32-token log loss by 0.05230 ($[0.02065, 0.09409]$) and raises AUC from 0.883 to 0.941.

We additionally test whether commitment merely recovers the coarse search-versus-close state used for prefix selection. A separate panel selects 188 pre-final checkpoints from 47 symbols excluded from direction construction, with both search and close checkpoints in two fixed generation windows. Eight futures per checkpoint give 1504 continuations, with fifteen whole symbols held out. At a 128-token horizon, commitment adds predictive information within both initial states: AUC rises from 0.631 to 0.751 within search and from 0.829 to 0.878 within close. The respective log-loss improvements are 0.02104 ($[0.00079, 0.04320]$) and 0.05478 ($[0.02082, 0.09433]$). Shorter search-state horizons have only one and six held-out events at 32 and 64 tokens, so we do not draw conclusions from them. These prospective results connect commitment to natural transitions beyond

the measured recent-history and state controls. They do not identify the upstream computation that produces commitment or establish natural mediation.

A.17.3 Commitment Magnitude and Intervention Duration

We use the same 19-prefix panel and two paired streams to test five persistent commitment targets, q_{10}, q_{30}, q_{50}, q_{70}, and q_{90}. Each lies within the frozen q_{05}–q_{95} range of natural scores at the corresponding generation position. We also apply q_{90} for only the first 32, 64, or 128 continuation tokens, then allow generation to proceed without intervention. The panel contains 304 new continuations and 38 reused no-intervention continuations.

Table 42: Exploratory commitment dose and duration experiment. Each condition contains 38 matched continuations. Quantiles refer to natural commitment scores, and settlement is measured within 1024 continuation tokens.

Target	Duration	Settled
No intervention	—	34.21%
q_{10}	Persistent	18.42%
q_{30}	Persistent	26.32%
q_{50}	Persistent	55.26%
q_{70}	Persistent	84.21%
q_{90}	Persistent	94.74%
q_{90}	First 32 tokens	52.63%
q_{90}	First 64 tokens	55.26%
q_{90}	First 128 tokens	52.63%

Settlement increases monotonically across the five persistent targets. Relative to no intervention, q_{10} lowers settlement by 15.79 points ($[-26.32, -5.26]$), while q_{70} and q_{90} raise it by 50.00 points ($[31.58, 68.42]$) and 60.53 points ($[42.11, 76.32]$). The q_{30} contrast is unresolved and the q_{50} interval touches zero. All three short interventions improve settlement over no intervention, but are 39.47–42.11 points less effective than persistent q_{90}. There is no resolved ordering among the three short durations. This exploratory panel shows graded control within the observed range of commitment scores; matching this scalar range does not imply that the entire intervened state or trajectory follows its natural distribution.

A.17.4 Crossing Answer Evidence with Commitment

We first cross three layer-17 existence conditions with three layer-16 commitment conditions on 38 naturally searching prefixes from 19 symbols. Existence is either unchanged or persistently set to its natural q_{05} or q_{95} value. Commitment is either unchanged or persistently set to the median of natural continued-search or imminent-close states at the corresponding generation position. Each of the nine cells contains 76 continuations using two paired streams, for 684 continuations in total. Previously completed cells are reused exactly.

Table 43: Settlement in the existence-by-commitment factorial ($N = 76$ per cell). Commitment has a large effect across existence conditions, while existence also affects settlement.

Existence condition	Toward Search	No commitment steering	Toward Commitment
Lower (q_{05})	17.11%	26.32%	86.84%
Unchanged	19.74%	27.63%	82.89%
Upper (q_{95})	28.95%	39.47%	93.42%

Commitment steering raises settlement by 53.95–60.53 points across the three existence conditions. The upper-minus-lower difference in this effect is -6.58 points ($[-19.74, 6.58]$). To measure which answer is selected, three model annotators label disjoint, condition-blind subsets of all 271 bounded finals from the six no-commitment/positive-commitment cells. They see only the requested symbol and final answer. Combining these labels with 185 continued-search outcomes gives an unconditional analysis of all 456 continuations. At fixed positive commitment, upper versus lower existence

steering raises false-proxy commitment from $55/76$ to $66/76$, or 14.47 points ($[5.26, 23.68]$), while absence changes from $7/76$ to $3/76$. Thus existence affects answer selection at fixed commitment. These are model annotations rather than human validation; the additional search-steered cells were not included in this answer audit.

We then replace internal existence steering with explicit, verified membership evidence. The panel contains twenty absent and twenty real emoji targets, five paired seeds, and two prompt conditions. The brief condition requests one short direct answer. The evidence condition states whether the target belongs to Unicode Emoji 17.0 and requests a short answer. Crossing these prompts with persistent q_{90} commitment gives 800 generations, including 400 exactly reused no-intervention generations. Unlike the prefix experiments, steering starts at the prompt end.

Table 44: Verified evidence crossed with commitment steering ($N = 100$ per row). Correct answer is measured by fixed final-answer rules: an explicit absence answer for absent targets, or the expected glyph without an absence claim for real targets. All percentages use all generations in the row, without conditioning on settlement.

Target	Supplied evidence	Commitment	Settled	Correct answer
Absent	None	Unchanged	22%	12%
Absent	None	q_{90}	85%	34%
Absent	Nonmembership	Unchanged	89%	79%
Absent	Nonmembership	q_{90}	98%	83%
Real	None	Unchanged	88%	59%
Real	None	q_{90}	96%	57%
Real	Membership	Unchanged	80%	69%
Real	Membership	q_{90}	90%	65%

On absent brief prompts, commitment raises settlement by 63 points ($[51, 75]$). At fixed q_{90}, verified nonmembership raises correct absence by 49 points ($[36, 62]$), while verified membership raises correct glyph output for real targets by 8 points ($[1, 17]$). Commitment alone changes real-item correctness by an unresolved -2 points ($[-8, 3]$), despite increasing settlement. The automatic answer rules are narrower than a general semantic correctness audit.

Together, these experiments distinguish evidence about which answer to give from the tendency to stop searching and give an answer. The functions are not strictly independent: existence steering changes settlement without commitment steering, and supplied evidence also changes settlement. The results support interacting evidence and commitment processes, without establishing a complete upstream circuit or a serial pathway from existence belief to commitment.

A.18 Direct existence-question contrast

To test whether failure-to-settle requires a false-presuppositional production request, we replace "Show me the x emoji" with two direct existence questions: "Does a dedicated official x emoji exist?" and "Does the x emoji exist?" We evaluate ten high-crashout symbols using 60 shared seeds per prompt, temperature 0.7, top-$p = 0.9$, and a 1024-token observation horizon. The dedicated-official question produces $263/600$ empty finals (43.8%) and $313/600$ horizon hits (52.2%), while the bare existence question produces $403/600$ empty finals (67.2%) and $468/600$ horizon hits (78.0%). Relative to the bare question, asking whether the target is a dedicated official emoji reduces empty finals by 23.33 percentage points (symbol-bootstrap 95% CI: [15.33, 30.83]), horizon hits by 25.83 points ([17.00, 34.83]), and mean completion length by 121.42 tokens ([76.10, 172.84]); all ten symbols move in the same direction for empty finals. Thus, an imperative or asserted-existence premise is not necessary for failure-to-settle, although wording and the clarity of the null criterion strongly moderate its incidence. This experiment does not establish that premise accommodation is irrelevant.

A.19 Qwen3.5 model-size scaling

We evaluate crashout across the Qwen3.5 family using the same panel of 70 absent emoji names. The 2B, 4B, 9B, 27B, and 122B-A10B models receive 60 matched rollouts per symbol.

Table 45: Crashout across Qwen3.5 model sizes on the same 70-symbol panel. Models from 2B through 27B use 60 rollouts per symbol and a locally enforced 2048-token completion horizon. The 122B-A10B row is a post-hoc matched-horizon estimate necessitated by different cloud-API token accounting.

Model	Crashouts	Rollouts	Crashout rate
Qwen3.5-2B	3959	4200	94.3%
Qwen3.5-4B	3090	4200	73.6%
Qwen3.5-9B	2792	4200	66.5%
Qwen3.5-27B	2544	4200	60.6%
Qwen3.5-122B-A10B[†]	286	700	40.9%

[†]Post-hoc 2048-token estimate. The directly observed rate under the cloud API's longer effective horizon was 22/700 (3.1%).