DECODING DYNAMICS REVEAL TEXTUAL-REPETITION LOOPS IN LARGE REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large reasoning models can spend their generation budget repeating text without completing an answer. We study which information in the model's next-token predictions distinguishes these textual-repetition loops from successful reasoning. Across 25 models and three benchmarks, we compare six signals computed from the top-k candidates recorded at each step: the concentration of the current prediction, and the recurrence of candidate probabilities, ranks, or identities at earlier steps that emitted the same token. For loops that have met a textual-repetition criterion, the recurrence of candidate identities, measured by set overlap, separates them from still-running successful generations most strongly, and controls show that candidate sets add information beyond the text-recurrence signals we evaluate. Accumulated with a moving average, the overlap yields an online detector that reads no hidden states and trains no classifier; its window and threshold are chosen on validation problems. Under greedy decoding, it alerts before the criterion on 85.3% of test-split loops at a 0.73% false-positive rate on successful generations, averaged over models. On nine models, prompting a final answer at its alerts reduces loops by 91.3% and generated tokens by 55.5%, while 99.89% of previously correct answers remain correct. Candidate-set recurrence is thus an observable of established repetition and a selective trigger for intervention under greedy decoding.

1 INTRODUCTION

Large reasoning models write out intermediate reasoning before they answer, and spending more computation on this reasoning improves performance on difficult problems (Snell et al., 2025; Guo et al., 2025). The same capacity for long generation also permits a costly failure, in which the model repeatedly produces the same text without reaching an answer (Pipis et al., 2026). We call this failure a *textual-repetition loop*, meaning that the generated tokens themselves repeat, not merely the ideas behind them. Such loops waste computation: in our evaluation of 25 models under greedy decoding (Section 3), generations that run to the length limit, almost all of them loops, account for 65% of all generated tokens. A loop recognized while generation is still running can be stopped before it wastes most of this computation.

The challenge is to detect a loop early enough to intervene without interrupting successful reasoning. Successful reasoning also reuses words, formulas, and earlier conclusions, and it can be long. Repeated text alone does not determine whether a generation will finish successfully: some generations in our data repeat a passage as often as our loop criterion requires and still answer correctly. Existing approaches include decoding penalties that modify token scores (Keskar et al., 2019; Ginart et al., 2026) and learned detectors that read hidden states (Duan et al., 2026; Xie et al., 2025). The former modify token selection, whereas the latter require access to internal activations. We ask whether next-token predictions provide evidence beyond repeated text that can support detection without changing token selection or training a classifier.

At every step, the model assigns probabilities to candidate next tokens (words or word pieces) and emits one of them. Its k most probable candidates, the top-k candidates, include alternatives it did not choose, and inference engines can report them without changing generation. A token such as "Wait" can recur in many contexts, each time accompanied by different alternatives. Our hypothesis is that a loop repeats more than its chosen tokens: when a looping model emits a token again, the alternatives it weighs alongside that token recur as well. To test this, we compare the candidates at

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.

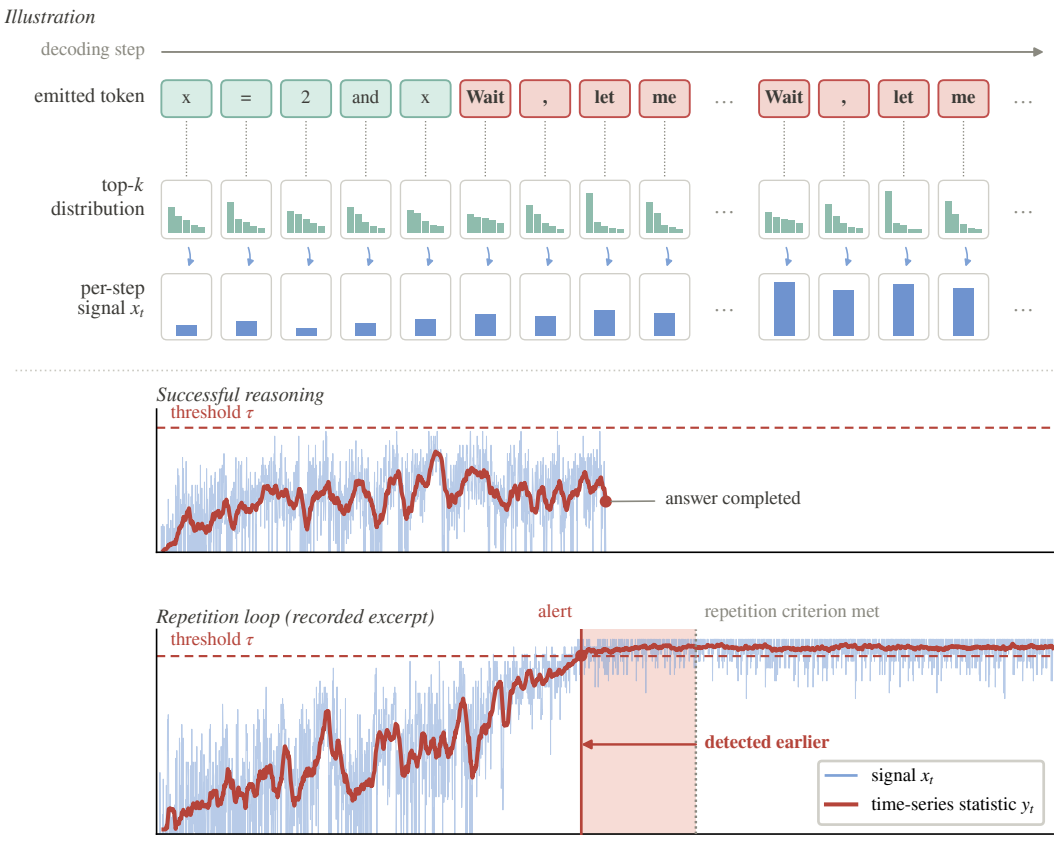


Figure 1: From candidate predictions to an online alert. Top: schematic construction of the per-step signal x_t from the recorded top-k candidates. Below: x_t (blue) and its aggregate y_t (red) for a complete successful generation and an excerpt of a repetition loop; in the loop, y_t crosses τ before the textual-repetition criterion is met.

each step with those at earlier steps that emitted the same token, by their probabilities, their ranks, or their identities, and contrast these comparisons with the concentration of the current prediction, a common proxy for confidence (Figure 1).

We find that candidate identities carry the clearest signal. Across 25 models and three benchmarks, the overlap between candidate sets separates loops from still-running successful generations more strongly than confidence, probabilities, or ranks do, and candidate sets add information beyond the text-recurrence signals we evaluate. We use a moving average of candidate-set overlap for online detection and test whether force-closing reasoning at its alerts reduces wasted computation.

The study makes three contributions:

- **An empirical characterization of candidate recurrence** (Section 4.1). We compare six signals, with controls for token counts and for successful generations that also meet the repetition criterion, and locate where next-token predictions carry information about loops.

- **Online loop detection** (Section 4.2). Under greedy decoding, a single configuration chosen on validation problems alerts before the textual-repetition criterion on most test-split loops, at a false-positive rate close to the criterion's own.

- **A selective intervention study** (Section 5). On nine models, prompting a final answer at the alerts removes most loops and more than half of the generated tokens. Reporting recovered and damaged answers separately exposes trade-offs that overall accuracy hides.

2 MEASURING CANDIDATE RECURRENCE

2.1 OBSERVABLE AND OUTCOME LABELS

A *generation* is the sequence of tokens that one model produces for one problem. At step t, the model emits token g_t, and we record $S_t = \{v_t^{(1)}, \dots, v_t^{(k)}\}$, the k most probable candidates of the prediction that produced g_t, together with their log-probabilities $\ell_t(v)$. Recording them does not change which token is emitted. We use $k = 50$ unless stated otherwise; k is the number of recorded candidates, not a limit on sampling.

We label loops with a textual-repetition criterion: some span of 30 generated tokens occurs 20 times (as in Pipis et al., 2026), counting overlapping occurrences, and t_{loop} is the first step at which any span completes its twentieth occurrence. The criterion belongs to a family of simple detectors: a *text counter* with threshold K alerts when a 30-token span completes its Kth occurrence, and the criterion is the counter at $K = 20$. A generation is *unfinished* if it reaches the token limit or, under the termination rules of Appendix B, stops without emitting its end-of-thinking delimiter, such as `</think>`. An unfinished generation that meets the criterion is a LOOP, and one that does not is OPEN; a finished generation is a PASS or a FAIL according to whether its answer is correct. A generation that repeats itself but eventually answers correctly is therefore a PASS.

A good detector alerts on LOOP generations before t_{loop} and stays silent on PASS generations; FAIL and OPEN generations enter no detection metric. Throughout the paper, *early* means before t_{loop}, not before the repetition begins.

2.2 SIGNALS FROM THE STORED CANDIDATES

Successful reasoning reuses common tokens, but the prediction around them changes: two steps may both emit an equals sign while ranking entirely different alternatives. Our hypothesis is that in a loop, a token emitted again tends to come with the alternatives that accompanied its earlier occurrence. We compare six per-step signals computed from the recorded candidates (Figure 1). Two describe the current prediction alone and measure how concentrated it is, a proxy for the model's confidence: Top1, its largest probability after renormalization over the recorded candidates, and Ent, one minus the normalized entropy of that distribution. The other four compare the current prediction with those at earlier steps that emitted the same token, the *same-token history* $\mathcal{H}(t) = \{t' < t : g_{t'} = g_t\}$: TV and JS compare probability mass (as one minus the total variation distance and one minus the normalized Jensen–Shannon divergence), Spr compares ranks (through the Spearman correlation $\rho_{t,t'}$ over the candidates the two predictions share), and Jac compares membership alone (through Jaccard overlap; Jaccard, 1912). With $\tilde{p}_t(v) = \exp \ell_t(v) / \sum_{u \in S_t} \exp \ell_t(u)$ the candidate distribution renormalized over S_t, H the Shannon entropy, and JSD the Jensen–Shannon divergence, the six signals are

$$
\begin{aligned}
x_t^{\text{Top1}} &= \tilde{p}_t\big(v_t^{(1)}\big), & x_t^{\text{Ent}} &= 1 - \frac{H(\tilde{p}_t)}{\log k}, \\
x_t^{\text{TV}} &= \max_{t' \in \mathcal{H}(t)} \sum_v \min\big(\tilde{p}_t(v), \tilde{p}_{t'}(v)\big), & x_t^{\text{JS}} &= \max_{t' \in \mathcal{H}(t)} \Big(1 - \frac{\text{JSD}(\tilde{p}_t, \tilde{p}_{t'})}{\log 2}\Big), \\
x_t^{\text{Spr}} &= \max_{t' \in \mathcal{H}(t)} \frac{1 + \rho_{t,t'}}{2}, & x_t^{\text{Jac}} &= \max_{t' \in \mathcal{H}(t)} \frac{|S_t \cap S_{t'}|}{|S_t \cup S_{t'}|},
\end{aligned}
\tag{1}
$$

all in $[0, 1]$ and larger for higher concentration or stronger recurrence. The four history signals are zero when $\mathcal{H}(t)$ is empty (other boundary cases in Appendix A).

The history signals share two choices. Matching the emitted token lets us compare the alternatives considered when the model makes the same token choice. Taking the maximum over the whole history removes the need to choose a repetition period, because the matching occurrence need not be the most recent one. Section 4.1 compares the full same-token history with two restricted alternatives.

2.3 FROM A PER-STEP SCORE TO AN ALERT

A single high signal value can also occur in successful reasoning, whereas the hypothesis predicts high values over many steps of a loop. We therefore aggregate the per-step signal x_t with one of two

standard statistics, a moving average over w steps or a one-sided cumulative sum (CUSUM) with drift δ (Page, 1954):

$$y_t = \frac{1}{w} \sum_{i=t-w+1}^{t} x_i \qquad (t \geq w-1), \tag{2}$$

$$y_t = \max\bigl(0,\, y_{t-1} + x_t - \delta\bigr), \qquad y_{-1} = 0. \tag{3}$$

The moving average responds to a high recent mean, whereas CUSUM lets a sustained excess above δ accumulate. Either detector alerts when y_t first exceeds a threshold τ. The window length w or drift δ, together with the alert threshold τ, is selected on the labeled validation split (Section 3).

3 Experimental Design

The evaluation takes up three questions in turn: whether candidate recurrence distinguishes the labeled outcomes at a fixed generation length (Section 4.1), whether it supports alerts before the textual-repetition criterion is met (Section 4.2), and what happens when those alerts trigger an intervention (Section 5).

Models and problems. We evaluate 25 open-weight reasoning models from eight families: DeepSeek-R1-Distill (Guo et al., 2025), Gemma 4 (Gemma Team, 2026), gpt-oss (OpenAI, 2025), Ministral 3 (Liu et al., 2026), Nemotron 3 Nano (NVIDIA, 2025; 2026), Phi-4 (Abdin et al., 2025; Xu et al., 2025), Qwen3 (Yang et al., 2025), and Qwen3.5 (Qwen Team, 2026). Each model answers the same 1,948 problems: 500 competition mathematics problems from MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), 448 graduate-level science questions from GPQA main (Rein et al., 2024), and 1,000 multi-domain questions sampled from MMLU-Pro (Wang et al., 2024). We decode greedily, always emitting the most probable token, with thinking enabled, at most 32,768 generated tokens, and the top-50 candidates recorded at every step. Greedy decoding provides frequent loop instances (Pipis et al., 2026); we separately evaluate transfer to each model's recommended sampling settings. Appendix B lists model identities, prompts, and decoding details.

Detection metrics. A detector can err in two ways: it can alert on successful reasoning or fail to alert on a loop. The *false-positive rate* (FPR) measures the first error as the fraction of PASS generations that receive any alert. The *early-detection rate* (EDR) measures timeliness as the fraction of LOOP generations first alerted before t_{loop}. *Potential token savings* (PTS) combines timing and misses: a LOOP generation of n tokens first alerted at step T (counted from zero) would save $n - (T+1)$ tokens if stopped there, a missed loop saves none, and PTS is the mean over LOOP generations. PTS assumes immediate stopping; Section 5 measures the outcomes of actual interventions. All three metrics are averaged over models with equal weight (macro averages). The criterion itself serves as the reference detector, alerting at t_{loop}: its false alerts are the PASS generations that repeat as much as loops, and its savings are the tokens a loop generates after meeting it.

Validation and test splits. For every model, the even-indexed problems form the validation split and the odd-indexed problems the test split, so both splits cover all models. The detector's signal, window, and threshold are chosen on the validation split, and the test split is used only for evaluation. The detector used from Section 4.2 onward combines the signal that separates loops best on the validation split in Section 4.1 with a moving average. Over its window lengths and thresholds, we select the configuration with the highest macro validation EDR whose macro validation FPR and PTS are both no worse than the reference's (search in Appendix A), then freeze it for the test split.

4 Candidate Recurrence and Online Detection

4.1 Signal Separation at Fixed Lengths

We first locate the information that distinguishes loops, comparing the concentration of the current prediction with the recurring probabilities, ranks, and identities of its candidates.

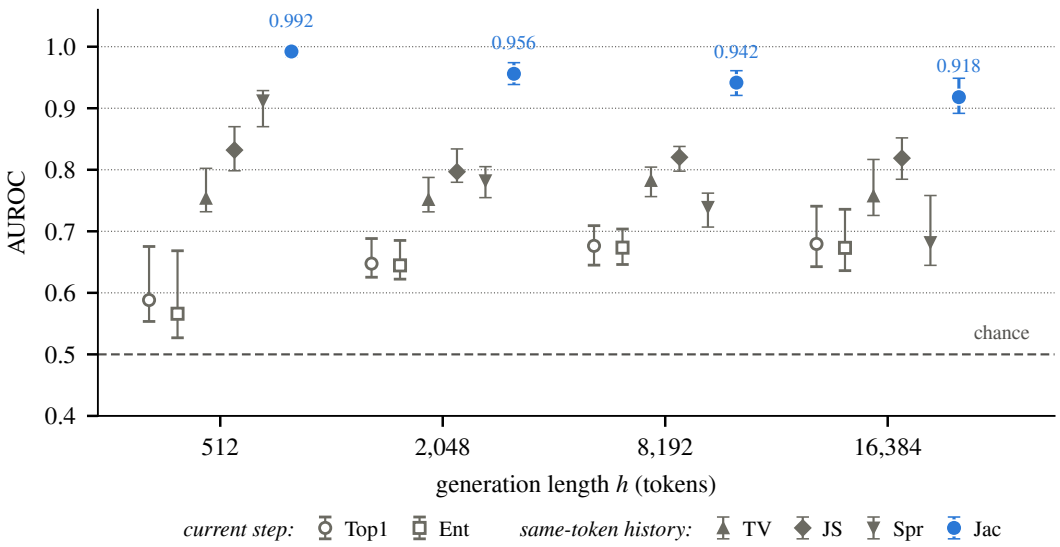


Figure 2: Single-step signal separation at matched generation lengths. LOOP generations have met the repetition criterion by the scoring step and are compared with still-running PASS generations at the same length. Points show macro AUROC with 95% problem-level bootstrap intervals; the dashed line marks chance.

Fixed-length comparison. Over all 1,948 problems, we compare two groups at $h = 512$, 2,048, 8,192, and 16,384 generated tokens: LOOP generations that have already met the repetition criterion, and still-running PASS generations that will answer correctly. Each generation is scored by its signal value x_h at that single step, without temporal aggregation; we call this a snapshot comparison. Comparing at a common length avoids contrasting long failures with short completed answers. The area under the ROC curve (AUROC) measures how well a signal ranks loops above PASS generations, from .5 at chance to 1 for perfect separation. We compute it within each model–benchmark pair that contains both groups, average over benchmarks and then over models, and report 95% problem-level bootstrap intervals. The eligible generations change with h, so the four lengths are separate snapshots rather than one cohort followed over time, and all of them observe loops after the criterion has been met; earliness is tested in Section 4.2.

Membership gives the strongest snapshot separation. Jac separates best at all four lengths, and the two current-step signals, Top1 and Ent, separate worst (Figure 2); on the test split alone, Jac again ranks first, with Top1 and Ent last (Appendix C). The history signals that compare probabilities or ranks fall between them. Which candidates recur thus distinguishes loops better than how their probabilities or ranks recur, and far better than the concentration of the current prediction.

Full-history matching improves snapshot separation. Comparing each step only with the most recent occurrence of its token lowers Jac's AUROC at every length, and comparing adjacent steps without matching the token lowers it further (Table 3 in Appendix C).

Candidate sets add information beyond token counts. Counting how often the emitted token has occurred before, without looking at candidates, nearly matches Jac at 512 tokens but falls far behind it at 8,192 tokens and beyond, and every paired difference favors Jac (Table 3); the gap is largest in the long generations that hold most of the wasted tokens.

Separation persists among generations that meet the criterion. Because the loop label itself requires repeated text, we also compare loops with PASS generations that have met the same criterion by the snapshot step and still answer correctly. No text counter with a threshold up to the criterion can separate these groups, because it alerts on all of them. Jac still separates them at 8,192 and 16,384 tokens, with bootstrap intervals above chance; at 2,048 tokens, too few such PASS generations exist

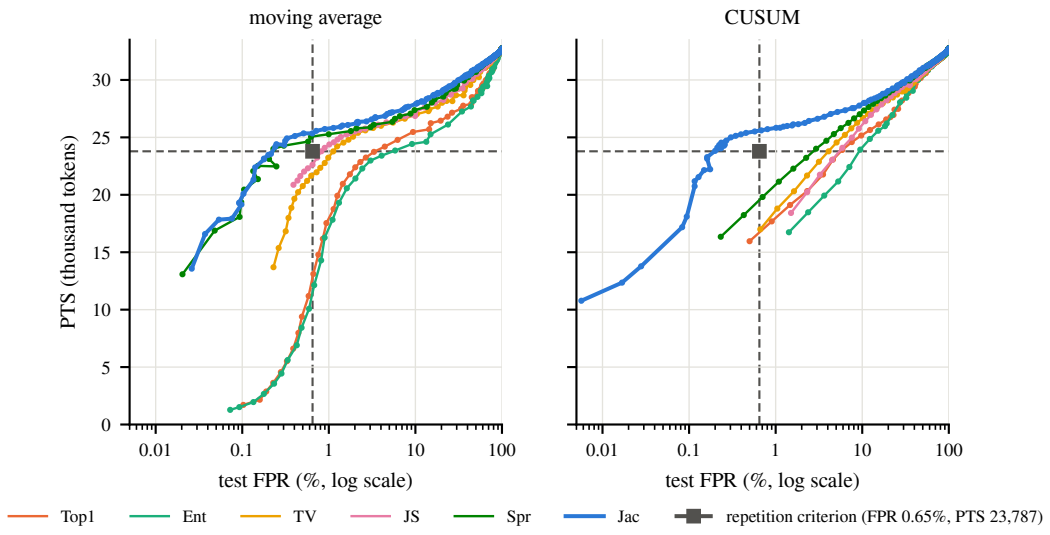


Figure 3: Online detection trade-offs. Test-split FPR and PTS for moving-average (left) and CUSUM (right) detectors. Each curve connects configurations on the validation FPR–PTS frontier for one signal and statistic, evaluated on the test split. The square and dashed lines mark the textual-repetition criterion.

to support a conclusion (Appendix C). Candidate recurrence therefore carries information beyond meeting the criterion.

4.2 Online detection

Separation at a single step does not guarantee useful decisions during generation: a detector must accumulate evidence over time, while PASS generations also produce occasional high signal values. We therefore evaluate the first alerts that each detector raises on the unmodified test-split generations.

Membership leads online. Over most of the FPR range, the Jac curves lie above those of Top1, Ent, TV, and JS under both statistics (Figure 3). Spr comes close under moving averages but not under CUSUM, so Jac's advantage over ranks at single steps carries over to online detection only in part. Curves show configurations on each signal–statistic pair's validation FPR–PTS frontier, evaluated on the test split (Appendix A). Both statistics include Jac configurations with a lower FPR and higher PTS than waiting for the criterion.

One shared configuration alerts earlier at a similar FPR. The selection rule of Section 3 yields the *candidate-set detector* used from here on: a Jac moving average with $w = 30$ and $\tau \approx 0.913$ (Appendix A), shared across all 25 models. On the test split, it alerts before the criterion on 85.3% of loops at an FPR of 0.73%, close to the criterion's own 0.65%, and stopping at its first alerts would save about 1,600 more tokens per loop than waiting for the criterion (Appendix D). A text counter alerts earlier as its threshold falls, but none of the thresholds we evaluate matches both figures: reaching these savings requires more than twice this FPR, and reaching this FPR requires the criterion itself (per-model results in Appendix D). Text counters with spans of 10 to 50 tokens and counts of 10 to 30 are likewise matched by Jac configurations with no higher FPR and no lower PTS (Figure 6 in Appendix D).

Comparison with a released hidden-state detector. Word Salad Chopper (WSC; Xie et al., 2025) flags repetitive, low-value chunks of reasoning with a classifier that reads hidden states at paragraph boundaries; its target overlaps with our loop label but is not defined by it. On the two models with released classifiers, DeepSeek-R1-Distill-Qwen-7B and -Llama-8B, every WSC point across the thresholds we evaluate is matched by a candidate-set point with no higher FPR and no lower PTS (Figure 4); at its published settings, WSC raises more false alerts than the candidate-set detector on

Table 1: Policy outcomes on nine models, pooled over all 17,532 paired generations (test split alone in Table 6), with 95% problem-level bootstrap intervals (±). Bold marks the best intervention in each column.

Policy	PASS (%) ↑	LOOP (%) ↓	Retention (%) ↑	Tokens (M) ↓
No intervention	58.61 ±1.46	23.11 ±0.77	100.00 ±0.00	182.4 ±5.2
Always-on penalties				
Repetition (1.1)	67.70 ±1.46	5.52 ±0.34	90.04 ±0.72	94.6 ±3.5
LZ (0.15)	**70.53** ±1.48	2.54 ±0.25	91.35 ±0.72	86.3 ±3.8
Force-close policies				
Budget (4,096 tokens)	66.11 ±1.48	4.99 ±0.37	95.75 ±0.52	**74.6** ±2.4
Text counter ($K = 10$)	69.47 ±1.47	2.29 ±0.23	99.62 ±0.12	81.3 ±3.3
Ours	69.80 ±1.47	**2.01** ±0.22	**99.89** ±0.06	81.2 ±3.4

both models, and its higher EDR and PTS on Qwen-7B come with a much higher FPR (Table 5). We apply the released classifiers with the published chopping rule, without fitting, and treat the first chunk each would cut as an alert (Appendix D).

Smaller candidate sets preserve most of the trade-off. Some interfaces report fewer than 50 candidates; with 20, 10, or 5, the Jac trade-off curves nearly coincide with those at 50. The curves diverge only below about 1% FPR, where narrower sets save slightly fewer tokens, and every width keeps a configuration with a lower FPR and higher PTS than the criterion (Figure 5 in Appendix D).

Lower early-detection rates under sampling. Under each model's recommended sampling settings and without re-selection, the detector's FPR falls to 0.22%, but its EDR falls too, to 52.4%: many sampled loops are detected late or not at all. Loops are also rare under sampling, occurring in 0.71% of generations versus 18.1% under greedy decoding (Appendix D).

5 ACTING ON ALERTS

Stopping a loop avoids its remaining tokens but leaves the problem without an answer. We therefore test a second use of the alerts: whether the reasoning generated before an alert can still produce an answer.

Action and comparisons. At an alert, *force-close* appends the model's end-of-thinking delimiter and lets generation continue, prompting the model to answer from the reasoning so far; this is the delimiter insertion used in budget forcing (Muennighoff et al., 2025); we call each insertion an *action*. The candidate-set detector is the frozen configuration selected in Section 3. We compare it with four baselines. A text counter ($K = 10$, chosen on the validation split) provides a text-only trigger; it resets its counts after each action, whereas the candidate-set detector keeps its history. A fixed token *budget* triggers force-close once at 4,096 tokens. Two always-on penalties modify every step instead of acting selectively: a repetition penalty lowers the score of every token already present in the prompt or output (Keskar et al., 2019), and a Lempel–Ziv (LZ) penalty lowers the scores of tokens that extend a compressible repeat (Ginart et al., 2026), evaluated in its authors' released implementation (Appendix E); their strengths are given in parentheses in Table 1. The comparison covers nine models from three families on all 1,948 problems. Force-close generations are paired with their unmodified counterparts problem by problem (Appendices B and E). Under the candidate-set policy, a generation without an alert is unchanged.

Outcome measures. An intervention can turn an incorrect or unfinished outcome into a correct answer, but it can also turn a correct answer into an incorrect one. Overall accuracy conflates these effects, so we report them separately: the PASS rate for final success, the share of loops and the total tokens for remaining waste, and *retention*, the fraction of answers correct without intervention that remain so. Measured run times are reported in Appendix E.3.

Loop reduction and answer recovery. Candidate-set force-close reduces loops by 91.3% and generated tokens by 55.5% (Table 1). The savings are large because the waste is concentrated: on these nine models, loops make up 23% of generations but consume 73% of their tokens. About half of the interrupted loops end with a correct answer, and most of the rest with an incorrect one (Appendix E.2). The cost to successful reasoning is small: only 11 of the 10,275 answers that were correct without intervention become incorrect. High retention reflects selective triggering: only 38 of these correct generations receive an alert, and 27 of them remain correct (Appendix E).

Answer recovery and retention. The LZ penalty reaches the highest overall PASS rate, and both penalties recover more incorrect or unfinished outcomes than candidate-set force-close. They also turn many more correct answers incorrect, on every model (Appendix E). The budget uses the fewest tokens but has the lowest retention of the force-close policies (Table 1), because a fixed cutoff also interrupts reasoning that would have answered correctly. At a similar token count, candidate-set force-close keeps more correct answers and leaves fewer loops than text-counter force-close, with paired intervals that exclude zero (Appendix E.2).

6 RELATED WORK

Repetition as a decoding phenomenon. Maximization-based decoding, such as greedy and beam search, tends to produce generic and repetitive text, which sampling from high-probability tokens reduces (Fan et al., 2018; Holtzman et al., 2020). Other work changes the training objective (Welleck et al., 2020; Xu et al., 2022) or penalizes previously generated tokens during decoding (Keskar et al., 2019; Ginart et al., 2026). Pipis et al. (2026) analyze repetition loops in reasoning models, examining the roles of learning errors and decoding temperature. We adopt their textual-repetition criterion but ask different questions: which observable in the model's predictions distinguishes unfinished repetitive generations from successful ones, and whether it supports earlier alerts.

Observables and the timing of intervention. Look-back decoding (Xu et al., 2023) compares current and past next-token distributions by KL divergence and adjusts decoding to avoid repetitive phrases and topic drift. Our comparison also looks back over the output history, but it matches earlier occurrences of the same emitted token, measures the recurrence of candidate identities, and leaves token selection unchanged. Other methods act on repetition throughout decoding: the LZ penalty (Ginart et al., 2026) changes next-token scores to suppress compressible repetition, CUSUM-based controllers combine token uncertainty with repetition features (Ettifouri et al., 2026a;b), and LoopGuard (Xu et al., 2026a) prunes repetitive spans from the key–value cache. Internal representations offer another source of evidence. Duan et al. (2026) accumulate the scores of a linear probe on sentence-averaged hidden states with CUSUM, and WSC (Xie et al., 2025) identifies repetitive chunks with a learned classifier on hidden states before truncating and regenerating the reasoning. Several of these methods also aggregate evidence over time; what distinguishes ours is the observable, the identities of candidates at recurring tokens.

Stopping for sufficiency versus detecting failure. Early-exit methods reduce inference cost by estimating when further reasoning is unlikely to change the answer. DEER (Yang et al., 2026) and Tikhonov et al. (2026) use confidence in intermediate answers, and Xu et al. (2026b) use changes in the predictive entropy of prompted intermediate answers. Answer Consistency (Liu & Wang, 2025) and Certaindex (Fu et al., 2025) use the stability of answers, ThinkBrake (Song et al., 2026) uses the log-probability margin of the end-of-thinking delimiter at sentence boundaries, and confidence-based monitoring filters or stops low-confidence reasoning (Fu et al., 2026). Our target is narrower: generation that repeats itself and remains unfinished within the token limit. The downstream action comes from budget forcing (Muennighoff et al., 2025), and our experiments evaluate candidate recurrence as one trigger for it.

7 DISCUSSION

Candidate identities versus probability mass. Candidate identities retain information that concentration scores discard. Under greedy decoding, two steps that emit the same token share the top-ranked candidate. When both predictions place most of their probability mass on that token,

probability-based similarities can remain high even if the alternatives differ. Jaccard overlap weights candidate identities equally, which makes it sensitive to those differences.

Limits of the evidence. The data are observational. High overlap could reflect repeated contexts, concentrated predictions, or other regularities of prolonged repetition, and we do not identify what causes a loop or show that the model revisits an internal state. Maximizing over the full history also gives frequent tokens more chances to match; the token-count and repetitive-success controls address the simplest text-only explanations. Finally, snapshot comparisons after the criterion and alerts before the twentieth occurrence cannot show that a loop is predictable before its repetition begins.

From an observable to a deployment policy. Under recommended sampling, loops are rare and the same configuration detects many of them late, which makes calibration and the cost of false alerts central to any deployment. Intervention is evaluated under greedy decoding on nine models from three families, with each penalty at a single strength, and delimiter insertion fails repeatedly on some loops. The detector also needs access to candidate lists and keeps a history that grows with the generation. Appendix F lists further limitations.

8 Conclusion

Textual-repetition loops leave a regularity in the identities of next-token candidates. Comparing candidate sets at recurring tokens reveals information beyond the model's confidence and a simple token count. Aggregated over time, this information yields alerts that precede the textual-repetition criterion under greedy decoding, and force-closing reasoning at these alerts reduces waste while changing very few answers that were already correct.

Ethics Statement

This work studies repetition failures in language model inference and interventions to reduce them. All experiments use public benchmarks and publicly released models, and involve no human subjects or personal data. The detector observes output candidates; the intervention attempts to end reasoning by inserting a delimiter. The resulting answers retain the risks of the underlying models, and intervention can change otherwise correct outputs.

Reproducibility Statement

Sections 2 and 3 and Appendices A to E describe the signal definitions, detector selection procedure, evaluation setup, and intervention protocol. The exact numerical threshold is given in Appendix A. Generation records, signal traces, and intervention outcomes are linked by model and problem identifiers. Code will be released upon publication.

AI Use Statement

We used generative AI tools to aid and polish writing; for retrieval and discovery of related work; for research ideation and execution, including methodological feedback, software implementation, analysis, and interpretation of experimental results; and to draft and edit portions of the manuscript. We did not use generative AI to generate synthetic datasets.

All AI-assisted outputs were reviewed by the authors. Reported numerical results were recomputed from the underlying experimental records, and manuscript claims were checked against the corresponding results. The authors take responsibility for the final content of this work, including all text, claims, code, and artifacts produced with the assistance of generative AI.

References

Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, Piero Kauffmann, Yash

Lara, Caio César Teodoro Mendes, Arindam Mitra, Besmira Nushi, Dimitris Papailiopoulos, Olli Saarikivi, Shital Shah, Vaishnavi Shrivastava, Vibhav Vineet, Yue Wu, Safoora Yousefi, and Guoqing Zheng. Phi-4-reasoning technical report. *arXiv preprint arXiv:2504.21318*, 2025.

Zenghao Duan, Liang Pang, Zihao Wei, Wenbin Duan, Yuxin Tian, Shicheng Xu, Jingcheng Deng, Zhiyi Yin, and Xueqi Cheng. Circular reasoning: Understanding self-reinforcing loops in large reasoning models. *arXiv preprint arXiv:2601.05693*, 2026.

El Hassane Ettifouri, Ayoub Belfatmi, Mahaman Sanoussi Yahaya Alassan, and Walid Dahhane. Calibrated e-CUSUM decoding for quantized reasoning models: Why token log-probability is the wrong observable for decoding monitors. *arXiv preprint arXiv:2607.11317*, 2026a.

El Hassane Ettifouri, Ayoub Belfatmi, Mahaman Sanoussi Yahaya Alassan, and Walid Dahhane. CUSUM-shaped inference-time monitoring and targeted re-decoding for quantized small language model reasoning. *arXiv preprint arXiv:2607.20129*, 2026b.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1082. URL https://aclanthology.org/P18-1082/.

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, and Hao Zhang. Efficiently scaling LLM reasoning with certaindex. In *Advances in Neural Information Processing Systems*, volume 38, pp. 141987–142025. Curran Associates, Inc., 2025. doi: 10.52202/085713-4751. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/d037fd021c9aace128b8ce25001cdb6c-Paper-Conference.pdf.

Yichao Fu, Xuewei Wang, Hao Zhang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/982572853408542621855 00e59bc0f28-Abstract-Conference.html.

Gemma Team. Gemma 4 technical report. arXiv preprint arXiv:2607.02770, 2026.

Tony A. Ginart, Naveen Kodali, Jason Lee, Caiming Xiong, Silvio Savarese, and John Emmons. LZ penalty: An information-theoretic repetition penalty for autoregressive language models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=vNzPB4YCHj.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

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. URL https://openreview.net/forum?id=rygGQyrFvH.

Paul Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50, 1912. doi: 10.1111/j.1469-8137.1912.tb05611.x.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Xin Liu and Lu Wang. Answer convergence as a signal for early stopping in reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17896–17907, 2025. URL https://aclanthology.org/2025.emnlp-main.904/.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1025.

NVIDIA. Nemotron 3 Nano: Open, efficient mixture-of-experts hybrid Mamba-Transformer model for Agentic reasoning, 2025. URL https://arxiv.org/abs/2512.20848. Technical report.

NVIDIA. NVIDIA-Nemotron-3-Nano-4B-BF16. Hugging Face model card, 2026. URL https://huggingface.co/nvidia/NVIDIA-Nemotron-3-Nano-4B-BF16.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. arXiv preprint arXiv:2508.10925, 2025.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1–2):100–115, 1954. doi: 10.1093/biomet/41.1-2.100.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 31932–31951, 2024. URL https://openreview.net/forum?id=wHBfxhZu1u.

Charilaos Pipis, Shivam Garg, Vasilis Kontonis, Vaishnavi Shrivastava, Akshay Krishnamurthy, and Dimitris Papailiopoulos. Wait, wait, wait... why do reasoning models loop? In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=oZWE7mSqlk. Spotlight. arXiv:2512.12895.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, pp. 10131–10165, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Sangjun Song, Minjae Oh, Seungkyu Lee, Sungmin Jo, and Yohan Jo. ThinkBrake: Efficient reasoning via log-probability margin guided decoding. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21765–21790, 2026. URL https://aclanthology.org/2026.findings-acl.1095/.

Pavel Tikhonov, Ivan Oseledets, and Elena Tutubalina. Confidence leaps in LLM reasoning: Early stopping and cross-model transfer. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 602–616, 2026. URL https://aclanthology.org/2026.eacl-short.43/.

vLLM Project. [Reasoning] Break repeating reasoning loops by forcing the reasoning end sequence. GitHub pull request #52677, vllm-project/vllm, 2026. URL https://github.com/vllm-project/vllm/pull/52677. Opened 2026-08-18; not merged as of 2026-09-25.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024. doi: 10.52202/079017-3018. Datasets and Benchmarks Track.

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. URL https://iclr.cc/virtual_2020/poster_SJeYe0NtvH.html.

Wenya Xie, Shaochen Zhong, Hoang Anh Duy Le, Zhaozhuo Xu, Jianwen Xie, and Zirui Liu. Word salad chopper: Reasoning models waste a ton of decoding budget on useless repetitions, self-knowingly. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33588–33598. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1705. URL https://aclanthology.org/2025.emnlp-main.1705/.

Dongjie Xu, Hao Wu, Weijie Shi, Yue Cui, Yuanjun Liu, Jiawei Li, Haolun Ma, An Liu, Jia Zhu, and Jiajie Xu. LoopGuard: Breaking self-reinforcing attention loops via dynamic KV cache intervention. *arXiv preprint arXiv:2604.10044*, 2026a.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, Shuohang Wang, Weijian Xu, Jianfeng Gao, and Weizhu Chen. Phi-4-mini-reasoning: Exploring the limits of small reasoning language models in math. *arXiv preprint arXiv:2504.21233*, 2025.

Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 3082–3095. Curran Associates, Inc., 2022. doi: 10.52202/068431-0223.

Nan Xu, Chunting Zhou, Asli Celikyilmaz, and Xuezhe Ma. Look-back decoding for open-ended text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1039–1050, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.66. URL https://aclanthology.org/2023.emnlp-main.66/.

Ting Xu, Xu He, Yupu Lu, Jiankai Sun, Dong Li, Wai Lam, and Jianye Hao. Unveiling the entropy dynamics of chain-of-thought reasoning. In *Forty-third International Conference on Machine Learning*, 2026b. arXiv:2606.02020.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8df90a1440ce782d1f5607b7a38f2531-Abstract-Conference.html.

A Signals, Detector Search, and Selection

Signal computation. In TV and JS, a candidate absent from one of the two lists receives zero probability; in Spr, $\rho_{t,t'}$ is computed on the ranks of the candidates shared by the two lists and set to zero when fewer than two are shared.

Configurations. For the curves of Section 4.2, each signal has 1,536 configurations: moving averages with $w \in \{2, 3, 5, 8, 12, 20, 30, 50, 80, 120, 200, 300\}$ and 73 thresholds, and CUSUM with drift $\delta \in \{0.05, 0.1, 0.15, 0.2, 0.3, 0.4, \ldots, 0.9\}$ and 60 thresholds. The moving-average thresholds are the 43 values $11/600, 24/600, \ldots, 557/600$ (0.018 to 0.928) followed by 30 values approaching 1 geometrically, up to $1 - 10^{-6}$; the CUSUM thresholds are 60 geometrically spaced values from 0.01 to 1,000.

Selection rule. The selected detector comes from a separate search over Jac moving averages with 34 window lengths (2, 3, 4, 5, 6, 8, 10, 12, 15, 20, 25, 30, 40, 50, 65, 80, 100, 120, 150, 200, 250, 300, 375, 470, 590, 740, 920, 1,150, 1,440, 1,800, 2,250, 2,800, 3,500, and 4,400 tokens); for each window, the thresholds considered are the largest values the statistic reaches in each successful validation generation, the only values at which the validation FPR changes. On the validation split, the repetition criterion has a macro FPR of 0.53% and a macro PTS of 24,064 tokens; among configurations that match or improve on both, we choose the one with the highest macro validation EDR. The selected configuration has $k = 50$, $w = 30$, and $\tau = 0.91329193$; on the validation split its macro FPR is 0.53%, its macro EDR 85.8%, and its macro PTS 25,575 tokens. One model has no validation loops and contributes to FPR only.

Curves in Figure 3. For each signal and statistic, we compute macro validation FPR and PTS for every configuration and discard a configuration if another one of the same signal and statistic is at least as good on both measures and better on one; the configurations that remain are the ones not dominated, and of configurations with identical FPR and PTS only one is kept. The figure shows the kept configurations at their test-split values, joined in order of validation FPR. PTS splits into two parts: the tokens a loop generates after the repetition criterion is met, which do not depend on the detector, and how far the first alert precedes the criterion. Formally, within a model, let T_i^{eff} be the first-alert step of loop i, or $n_i - 1$ if it is missed; then PTS $= \frac{1}{L}\sum_i (n_i - 1 - t_{\text{loop},i}) + \frac{1}{L}\sum_i (t_{\text{loop},i} - T_i^{\text{eff}})$ over its L loops. The first term does not depend on the detector, so on the same generations a difference in PTS equals a difference in mean token lead relative to the repetition criterion.

B Models, Prompts, and Decoding

Table 2 lists the 25 models and their outcome classes under greedy decoding. Math problems ask for the final answer in `\boxed{}`, and multiple-choice questions carry a short instruction on the answer format. The 1,000 MMLU-Pro questions are sampled from its test split with seed 42. Answers are taken from `\boxed{}`, with standard fallbacks, and checked with math-verify for MATH-500 and by option letter for GPQA and MMLU-Pro; unfinished generations are not graded, and interventions are graded the same way. The recorded candidates and log-probabilities are those of the model's next-token distribution before temperature scaling and top-p truncation. Decoding is greedy, always emitting the most probable token, with at most 32,768 generated tokens and a context of 36,864 tokens. For DeepSeek-R1-Distill, Phi-4, Qwen3, and Qwen3.5, whose reasoning closes with `</think>`, a generation that stops before 32,768 tokens without the tag is also unfinished; this applies to twelve generations (DeepSeek-R1-Distill-Qwen-7B, Phi-4-mini-reasoning, and Phi-4-reasoning). Phi-4-reasoning reaches this context length with the YaRN extension (factor 1.5; Peng et al., 2024). For the nine intervention models, generation is reproducible token for token: a problem yields the same output however many other problems are generated alongside it. An intervened generation therefore matches its counterpart without intervention up to its first intervention.

C Additional Signal Results

Separation at fixed lengths. The 95% intervals of the Jac AUROC in Figure 2 are [.984, .996], [.939, .974], [.921, .961], and [.892, .949] at $h = 512$, 2,048, 8,192, and 16,384; intervals here and

Table 2: Models and outcome classes under greedy decoding (1,948 problems each).

Model	PASS	FAIL	LOOP	OPEN
DeepSeek-R1-Distill				
Qwen-7B	1,139	549	260	0
Qwen-14B	1,397	421	129	1
Qwen-32B	1,425	318	205	0
Llama-8B	1,066	487	393	2
Gemma 4				
E2B	1,241	696	11	0
E4B	1,422	526	0	0
12B	745	31	1,148	24
26B-A4B	1,104	91	752	1
31B	1,710	214	24	0
gpt-oss				
20b	1,489	409	48	2
Ministral 3				
3B	952	155	818	23
8B	1,308	252	322	66
14B	1,243	199	491	15
Nemotron 3 Nano				
4B	1,238	428	282	0
30B-A3B	1,365	234	309	40
Phi-4				
mini-reasoning	1,224	422	300	2
reasoning	521	284	1,134	9
Qwen3				
4B	1,359	455	133	1
8B	1,449	418	80	1
14B	1,524	413	9	2
32B	1,569	354	22	3
Qwen3.5				
4B	776	50	1,093	29
9B	1,340	169	424	15
27B	1,576	163	197	12
35B-A3B	1,565	146	232	5
Total	31,747	7,884	8,816	253

below come from resampling problems 2,000 times (bootstrap), and each replicate keeps a resampled problem's outcomes across all models and policies. On the validation and test splits separately, Jac has the highest AUROC at every length, and on the test split Top1 and Ent have the lowest.

First and second controls. Table 3 scores the generations of Figure 2 at the same step with four variants that differ only in which earlier steps they compare. *Jac, all earlier* is the signal of Eq. (1): the largest Jaccard overlap between the current candidate set and that of any step in the same-token history $\mathcal{H}(t)$. *Jac, most recent* uses only the latest step in $\mathcal{H}(t)$. *Jac, preceding step* uses step $t-1$, whatever token it emitted. *Earlier count* is $|\mathcal{H}(t)|$, the number of earlier steps that emitted the same token, and reads no candidates. The two same-token variants of Jac are zero when $\mathcal{H}(t)$ is empty. AUROC is computed and averaged as in Figure 2; the second row counts the models that contribute at each length, and each difference to *Jac, all earlier* is computed on the same bootstrap replicates.

Third control. Against PASS generations that had met the 30-token, 20-occurrence criterion by the snapshot step (16, 52, and 43 generations at $h = 2{,}048$, 8,192, and 16,384; one at 512), Jac has AUROC .605 [.294, .933], .847 [.743, .896], and .832 [.748, .884]; at the two longest lengths the whole interval lies above .5, the value of chance.

Table 3: Full history improves Jac separation. AUROC at the lengths of Figure 2; bold marks the highest at each length. All alternatives have lower AUROC than full-history Jac; paired 95% bootstrap intervals for the differences exclude zero.

h	512	2,048	8,192	16,384
Models (of 25)	9	22	24	22
Jac, all earlier	**.992**	**.956**	**.942**	**.918**
Jac, most recent	.884	.845	.812	.805
Jac, preceding step	.624	.603	.497	.587
Earlier count	.974	.874	.720	.706

D Additional Detection Results

Selected configuration. On the test split, stopping at the first alerts of the selected configuration would avoid 25,392 tokens per loop on average, 1,605 more than waiting for the repetition criterion. A text counter precedes the criterion on every loop whenever $K < 20$; saving as many tokens as the selected configuration requires $K \leq 12$ and an FPR of at least 1.90%, and an FPR at or below 0.73% requires $K = 20$, the criterion itself. Per model, test FPR ranges from 0% (nine models) to 5.2% (Qwen3-4B), and EDR from 53.0% (Qwen3.5-35B-A3B) to 100% over the 24 models with test loops. Table 4 lists every model.

Text counters. Figure 6 scores text counters that alert when some span of n tokens has occurred K times, for $n \in \{10, 20, 30, 50\}$ and $K \in \{10, 20, 30\}$, on the same labels as Figure 3; the counter with $n = 30$ and $K = 20$ is the repetition criterion. Every counter is matched by a Jac configuration with no higher test FPR and no lower PTS.

WSC. We use the classifiers released by the authors for DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B, unchanged. At every chunk boundary (a token containing a blank line), the classifier reads the penultimate-layer hidden state and scores the chunk; the published rule cuts when two consecutive chunks score above 0.5, and requires five consecutive chunks instead of two when the current chunk has fewer than 10 tokens. We run each stored greedy generation through the model to obtain the hidden states the classifier would read online, and take the last token of the first chunk it would cut as the alert. FPR, EDR, and PTS are defined as in Section 3 and computed on the test split. Figure 4 varies only the threshold of the published rule; for each model it keeps the thresholds not dominated on the validation split and shows them at their test values.

Candidate width. Figure 5 computes Jac from only the k most probable candidates and plots curves as in Figure 3. Every width has a configuration with lower FPR and higher PTS than the repetition criterion; under moving averages with five candidates, there is only one.

Recommended sampling. Sampling uses each model's recommended decoding parameters from its model card or official evaluation settings, with seed 42; the same-token history uses the token actually sampled at each step. Of the 48,700 sampled generations, 344 are LOOP; the test split holds 172 of them, from 19 models. Without re-selecting its configuration, the selected detector alerts on 142 of these loops, 89 of them before the criterion, and raises 37 false alerts on successful test generations; its macro PTS over the 19 models is 17,771 tokens.

E Additional Intervention Results

E.1 Policies

The nine intervention models are the four DeepSeek-R1-Distill, three Ministral 3, and two Phi-4 models of Table 2. Force-close appends a line break, `</think>`, and two line breaks for the DeepSeek-R1-Distill and Phi-4 models, and the same with `[/THINK]` for the Ministral 3 models. The detector has no notion of reasoning and answer phases, so an alert raised during the answer inserts the delimiter again. The repetition penalty (strength 1.1) lowers the score of every token already

Table 4: Selected configuration on the test split, per model. EDR and PTS are undefined for a model without test loops (–) and are averaged over the other 24 models.

Model	PASS	FPR (%)	LOOP	EDR (%)	PTS (tokens)
DeepSeek-R1-Distill					
Qwen-7B	554	0.90	129	87.6	25,904
Qwen-14B	702	1.00	64	85.9	23,813
Qwen-32B	696	0.14	105	94.3	26,628
Llama-8B	516	0.00	199	93.5	25,679
Gemma 4					
E2B	617	0.16	6	100.0	27,995
E4B	695	0.00	0	–	–
12B	370	1.89	577	83.9	24,027
26B-A4B	566	0.18	366	67.8	23,075
31B	867	0.23	10	60.0	24,618
gpt-oss					
20b	739	2.84	23	73.9	26,401
Ministral 3					
3B	479	0.42	410	91.5	27,747
8B	649	0.46	164	95.7	26,098
14B	617	0.32	255	98.4	25,683
Nemotron 3 Nano					
4B	605	0.66	144	95.8	28,434
30B-A3B	667	0.30	166	87.3	22,320
Phi-4					
mini-reasoning	621	0.00	149	94.6	24,854
reasoning	267	1.50	564	81.4	29,066
Qwen3					
4B	672	5.21	77	96.1	27,549
8B	725	2.07	42	95.2	26,185
14B	764	0.00	6	100.0	23,205
32B	788	0.00	12	75.0	26,012
Qwen3.5					
4B	386	0.00	541	87.1	24,489
9B	685	0.00	206	80.6	23,450
27B	803	0.00	93	67.7	23,718
35B-A3B	785	0.00	117	53.0	22,459
Macro average		0.73		85.3	25,392

Table 5: Released hidden-state detector and the candidate-set detector on the test split.

Detector	FPR (%) ↓	EDR (%) ↑	PTS (tokens) ↑
DeepSeek-R1-Distill-Qwen-7B			
WSC	22.2	91.5	26,476
Ours	0.9	87.6	25,904
DeepSeek-R1-Distill-Llama-8B			
WSC	6.0	83.9	24,728
Ours	0.0	93.5	25,679

present in the prompt or the output. The LZ penalty (strength 0.15) is evaluated in the code released by its authors, which omits the term of the published equation for extending an already matched repeat. The text counter counts overlapping occurrences of 30-token spans in model-generated text, alerts when a span completes its Kth occurrence, and resets its counts after each delimiter; inserting the delimiter when a text-repetition check fires has also been proposed for vLLM (vLLM Project, 2026). We use $K = 10$, the smallest of 5, 10, and 15 with a validation FPR of at most 5%.

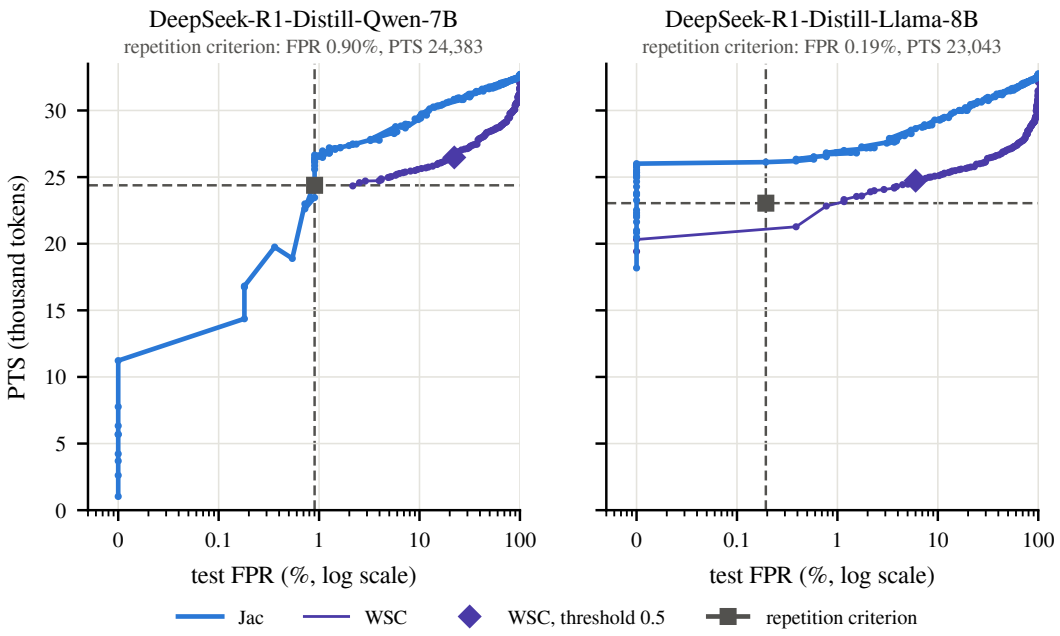


Figure 4: WSC and the candidate-set detector per model: FPR versus PTS on the test split. Diamonds mark WSC at its published threshold of 0.5. An FPR of 0 is drawn at the left edge; dashed lines mark the repetition criterion.

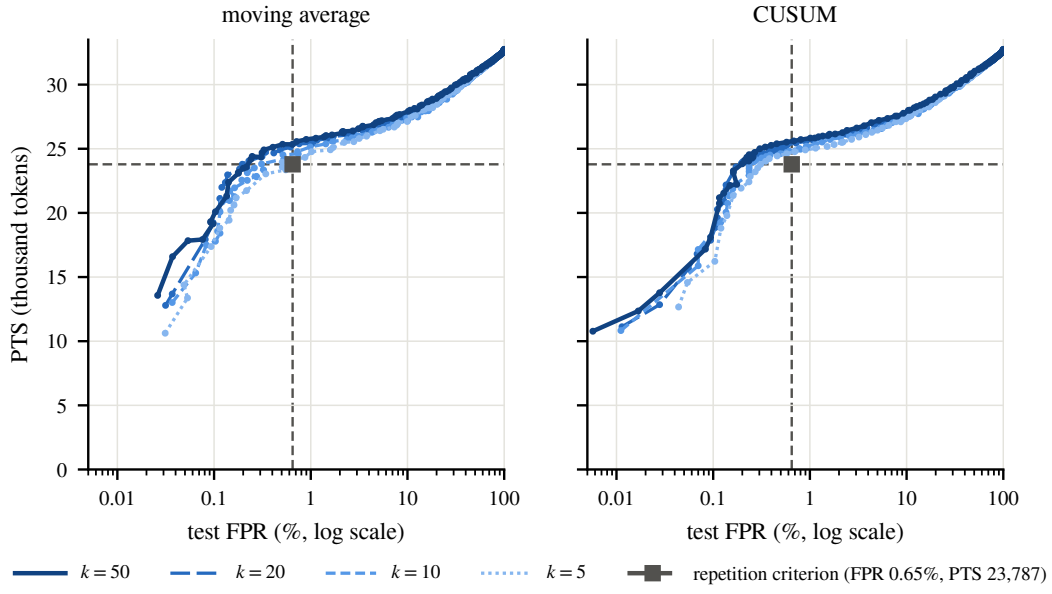


Figure 5: Effect of the number of recorded candidates on Jac's test-split FPR–PTS trade-off. Curves use configurations selected on the validation split; dashed lines mark the repetition criterion.

E.2 Outcomes

Per model, candidate-set force-close reduces loops by 77.0–99.3% (Phi-4-reasoning: 1,134 to 261) and tokens by 36.9–70.8%, and loses at most six answers that are correct without intervention (retention 99.47–100%). Pooled over the nine models, the 95% problem-level bootstrap interval for retention is 99.83–99.95%; on the test split alone, loop and token reductions are nearly unchanged and retention is 99.90%. Of the 38 generations that are correct without intervention and receive an

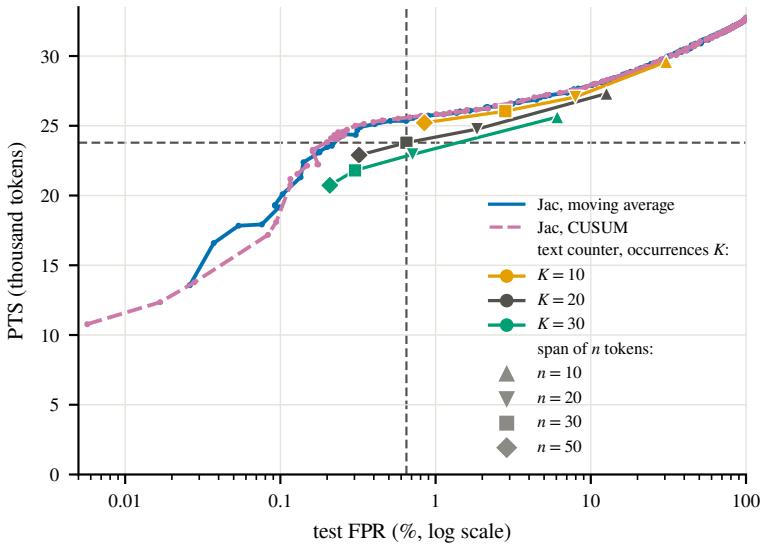


Figure 6: Text counters with span length n and count K: plotted with the Jac curves of Figure 3 on the test split. Dashed lines mark the repetition criterion ($n = 30$, $K = 20$).

alert, 27 still answer correctly. Of the 4,052 loops without intervention, 1,958 become PASS, 1,741 become FAIL, 352 remain LOOP, and one ends unfinished without a loop, while one generation that does not loop without intervention loops under it; over all generations, PASS rises from 10,275 to 12,238 and FAIL from 3,087 to 4,844.

Per model, retention is 66.0–95.3% under the repetition penalty and 77.2–95.7% under LZ, and both penalties lose more previously correct answers than candidate-set force-close on every model. Against the text counter, candidate-set force-close has fewer loops on five of the nine models and fewer tokens on two; its lower pooled loop count comes mainly from Phi-4-reasoning (261 against 295 loops). Paired over problems, ours minus the counter is $+0.33$ [$+0.13$, $+0.54$] in PASS (%), -0.27 [-0.44, -0.11] in LOOP (%), $+0.27$ [$+0.17$, $+0.39$] in retention (%), and -0.09 [-0.97, $+0.79$] in tokens (M), with 95% intervals from resampling the problems 2,000 times, as for Table 1. On the test split alone (Table 6), the best policy in each column is unchanged, and ours minus the counter is $+0.41$ [$+0.13$, $+0.72$] in PASS (%), -0.31 [-0.54, -0.08] in LOOP (%), $+0.35$ [$+0.18$, $+0.54$] in retention (%), and $+0.05$ [-0.58, $+0.64$] in tokens (M). On DeepSeek-R1-Distill-Qwen-7B, DeepSeek-R1-Distill-Llama-8B, and Ministral-3-8B, one-time budgets of 4,096, 8,192, and 16,384 tokens lose 148, 58, and 19 correct answers and generate 19.1M, 23.5M, and 33.7M tokens, against 8 answers and 26.0M tokens for candidate-set force-close.

Table 6: Policy outcomes on the test split alone (8,766 paired generations), as in Table 1.

Policy	PASS (%) ↑	LOOP (%) ↓	Retention (%) ↑	Tokens (M) ↓
No intervention	58.19 ±2.03	23.26 ±1.16	100.00 ±0.00	91.6 ±3.8
Always-on penalties				
Repetition (1.1)	67.16 ±1.98	5.54 ±0.48	89.88 ±1.04	47.5 ±2.5
LZ (0.15)	**70.37** ±2.05	2.76 ±0.37	91.32 ±0.98	43.7 ±2.7
Force-close policies				
Budget (4,096 tokens)	65.42 ±2.03	5.19 ±0.52	95.35 ±0.79	**37.8** ±1.7
Text counter ($K = 10$)	68.95 ±2.09	2.40 ±0.33	99.55 ±0.19	41.0 ±2.3
Ours	69.36 ±2.05	**2.09** ±0.31	**99.90** ±0.10	41.0 ±2.4

E.3 Repeated actions and run time

Inserted delimiter tokens enter the moving average as zeros and are not added to the history; neither is reset after an action, and no alert is raised while a delimiter is being inserted. Candidate-set force-close issues 92,630 actions, and 3,533 of the 4,160 generations with an action receive exactly one. The 257 generations with more than 100 actions (up to 903 each) account for 8.4M tokens both without and with intervention, and 253 of them remain loops; most savings come from the one-action group, whose tokens fall from 114.6M to 21.1M.

Table 7 gives the time to generate all 1,948 problems of each of the nine models, with one NVIDIA B300 GPU per model, vLLM 0.28.0 (Kwon et al., 2023), batch-invariant decoding, and all problems of a model submitted at once, including model loading. These times depend on the hardware and serving configuration; the token counts of Table 1 do not.

Table 7: Total generation time on nine models (GPU-hours, including model loading).

Policy	Time (h)
No intervention	49.6
Repetition penalty (1.1)	18.9
LZ penalty (0.15)	22.6
Ours	14.4

F Scope and Limitations

1. **Model scale.** The largest model studied has 35B parameters; behavior at larger scales is untested.
2. **Interface requirements.** The detector needs candidate identities at every step, including reasoning tokens, and its configuration depends on the number of observed candidates; force-close additionally requires appending tokens. Services that hide candidates cannot support the detector, and services that disallow token insertion cannot support force-close.