DRIFTING AND MERGING: THE DYNAMICS OF LLM DEGENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

An LLM can continue generating long after its output has stopped making meaningful progress. What determines whether generation continues to develop the text or becomes generic, incoherent, or repetitive? Since Holtzman et al. (2020), much research has focused on decoding and fine-tuning methods to reduce these problems, but how they develop during generation remains poorly understood. We organize these forms of text degeneration through two autoregressive effects: *drifting* moves the distribution of generated text away from the natural-language distribution, while *merging* makes predictions from different histories increasingly similar. To test this framework quantitatively, we focus on persistent repetition, whose onset is relatively easy to identify. We call this onset time the *collapse time*. We define and measure drifting and merging times, τ_{drift} and τ_{merge}, from prompts alone and use them to predict collapse time. Across 16 models, five corpora, and five attention temperatures, a regression using these two timescales explains 93.8% of the variance in mean log collapse time on the fitting data. Guided by this analysis, we propose a simple fine-tuning objective that increases τ_{merge}. This increases Llama-3.1-8B's geometric-mean collapse time to $5.52\times$ its original value with zero additional data.

1 INTRODUCTION

Coding, reasoning, and agent workflows can now involve tens or hundreds of thousands of generated tokens, often across multiple model calls. Across these long sequences, outputs must do more than read fluently in isolation: they must remain consistent with earlier requirements and decisions. As generation continues, it can lose this consistency, become generic or incoherent, or get stuck in a repetitive loop. These forms of *neural text degeneration* (Holtzman et al., 2020) make it difficult to sustain reliable generation over long tasks. Existing methods address degeneration by modifying decoding, training objectives, or data preparation (Meister et al., 2023; Welleck et al., 2019; Xu et al., 2022; Li et al., 2023a). Recent work also improves long-form generation through decoding interventions (He et al., 2026) and training that interleaves planning, writing, and reflection (Sun et al., 2026). To study the dynamics behind these failures, we focus on uninterrupted text generation: what changes as the model repeatedly conditions on its own outputs, and how do those changes affect long-term stability?

A natural starting point is predictive (cross-entropy) loss, which measures how well a model predicts the next token in corpus text. During generation, each generated token becomes part of the history used for later predictions. Understanding degeneration therefore requires examining how the generation process evolves as the model repeatedly extends its own history. As generation continues, the distribution of generated text can drift away from the natural-language distribution. We call this process *drifting*.

Drifting, however, does not describe whether different histories remain predictively distinguishable. To examine how these distinctions evolve, we consider a text ensemble generated from different natural-language histories, rather than a single trajectory. We call the fading of predictive distinctions within this ensemble *merging*: predictions from different histories become increasingly similar, weakening the histories' influence on what comes next, even when the output remains fluent. Geometrically, merging can be viewed as a contraction of the ensemble's effective "volume" in

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.

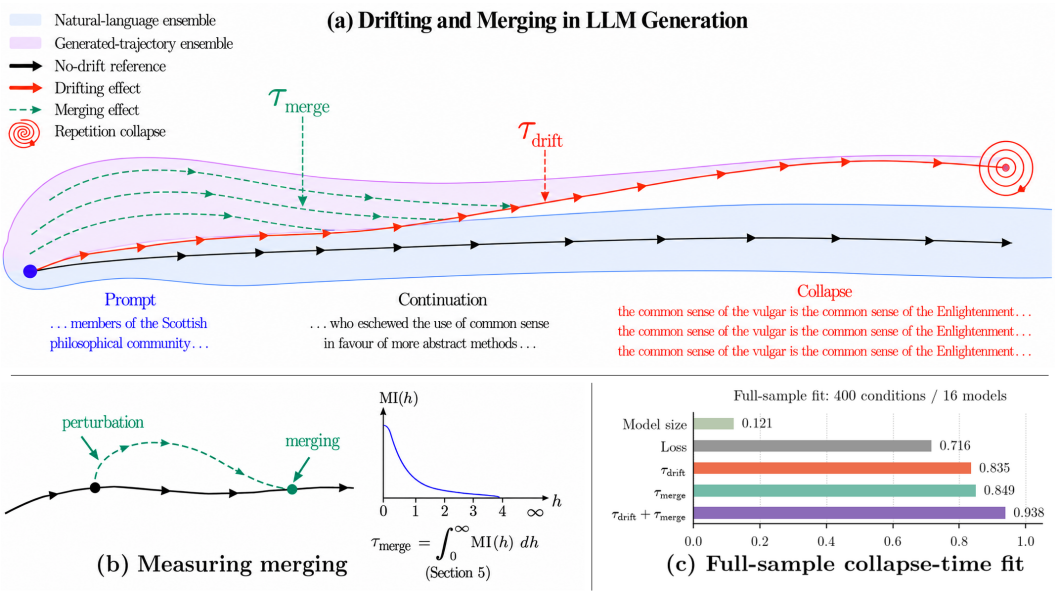


Figure 1: Drifting, merging, and collapse in autoregressive generation. **(a)** Proposed mechanism and an example of repetitive collapse. **(b)** Merging measured by entropy-normalized mutual-information decay after a token perturbation. **(c)** Full-sample R^2 for predicting mean log collapse time across 400 conditions; the joint predictor labeled $\tau_{\text{drift}} + \tau_{\text{merge}}$ gives $R^2 = 0.938$. Labels name the timescales; drifting enters the regression logarithmically and merging linearly.

predictive-state space, as illustrated in Figure 1(a). In Section 3, we show how drifting and merging in a toy ODE can produce patterns similar to different forms of text degeneration.

To test this framework quantitatively, we focus on *repetitive collapse*, a particularly conspicuous form of degeneration in which generated text becomes trapped in a loop of repeated words or phrases. Its identifiable onset makes it well suited for connecting the two effects to an observable outcome. We define the *collapse time* as the generated-token position at which that loop begins (Section 4.1). Our target is modeling of the geometric-mean collapse time across prompts for each model, corpus, and generation setting.

We characterize the two effects through a drifting time τ_{drift} and a merging time τ_{merge}: longer times indicate slower drifting from natural language and longer retention of distinctions between histories, respectively. Both are estimated from prompts, without generating the long continuations whose collapse times we seek to predict. We estimate τ_{drift} from the predictive loss in the prompts. To estimate τ_{merge}, we perturb a token in the prompts and measure how long its effect on predictions persists along subsequent tokens in the original prompt, as illustrated in Figure 1(b).

The two timescales represent two sources of constraints that prevent repetition collapse: the constraint from natural-language structure and the constraint from the evolving context. A generation enters repetition loop if both constraints are lost, producing non-language and generic text. Losing either alone need not produce a loop: text can become incoherent without repeating, or remain fluent but generic. We model persistent repetition as arising when both constraints weaken, and relate their timescales to collapse time in Section 5.

We test this prediction across 16 models, five corpora, and five attention temperatures (i.e., 400 conditions). Attention temperature controls the sharpness of attention weights and defaults to 1.0. Measurements suggest that longer τ_{drift} and τ_{merge} are strongly associated with later collapse. As shown in Figure 1(c), a simple regression using $\log \tau_{\text{drift}}$ and τ_{merge} explains 93.8% of the variance in mean log collapse time across the 400 conditions used for fitting.

Results of held-out experiments across attention temperatures and model sizes (Section 5) support the complementary roles of drifting and merging. First, moderately lowering attention temperature can delay collapse even while increasing prompt loss (i.e., shorter τ_{drift}), a pattern that drifting

alone does not explain. Second, LongWriter models (Bai et al., 2025), post-trained for long-form generation, collapse later than their base models despite higher prompt loss under raw continuation. In both settings, longer merging times τ_{merge} help account for the delayed collapse.

These results motivate a new LLM fine-tuning objective: increasing τ_{merge} through training to preserve predictive distinctions between histories for longer and delay repetitive collapse. We fine-tune Qwen3-4B and Llama-3.1-8B with this objective while limiting increases in prompt loss and deviation from the base model. Without additional training data, fine-tuning extends Llama-3.1-8B's geometric-mean collapse time by up to $5.52\times$, with little change in prompt loss. Under greedy decoding, the fine-tuned 8B model collapses later than an unmodified 300B model (Section 6). These results suggest that information persistence offers a promising new direction for shaping the long-term behavior of LLM generation while maintaining predictive quality.

2 Related Work

Neural text degeneration and repetition. Prior work has proposed several explanations for why generated text degenerates or enters repetition loops. Holtzman et al. (2020) reveals that greedy and beam decoding can produce repetition even when a model assigns high likelihood to natural-language data. Repetition can also take the form of recurring code structures rather than exact token loops (Dong et al., 2025). Proposed explanations include transitions that concentrate probability on the same tokens (Fu et al., 2021), repetition that reinforces itself (Xu et al., 2022), and repeated training material (Li et al., 2023a). Repeated-subsequence statistics also show differences in the long-range structure of generated and natural text (Tanaka-Ishii, 2026). We study how drifting and merging jointly predict repetitive collapse time and test whether increasing information persistence delays collapse.

Long-horizon generation and interventions. Research on long-horizon generation has proposed practical methods for sustaining long-form output (Bai et al., 2025). During generation, a model must repeatedly use its own outputs as context. Exposure-bias studies examine this shift from reference to model-generated histories (Arora et al., 2022) and whether models recover from earlier deviations (He et al., 2021). Practical methods include unlikelihood training (Welleck et al., 2019), probability truncation (Holtzman et al., 2020), long-output fine-tuning (Bai et al., 2025), and explicit planning (Yang et al., 2023). Recent methods combine planning with feedback for constrained long-form output (Zhu et al., 2026), or train models to interleave planning, writing, and reflection (Sun et al., 2026). Other work studies output-length instability through attention traces and decoding interventions (He et al., 2026). We focus on the underlying dynamics of text generation: how stability changes as a model continues to generate under a fixed decoding rule.

Dynamics and information persistence. Dynamical studies and memory measures provide tools for studying how generation changes over time. Prior work examines temperature-dependent transitions and dimensional collapse in generated text (Nakaishi et al., 2024; Du & Tanaka-Ishii, 2025). Recent work links mode collapse to internal trajectories becoming confined to a low-dimensional space and reduces it by regulating the value cache (Du & Tanaka-Ishii, 2026). Lagged token mutual information measures dependence between observed tokens (Lin & Tegmark, 2017), while Fisher memory and information response measure how perturbations affect later states (Ganguli et al., 2008; Auconi et al., 2021). We use token perturbations to measure how long predictive differences between histories persist. Combining this local measure of merging with drifting lets us predict repetitive collapse time from prompts. Appendix H provides an extended review and compares information-persistence measures.

3 A Dynamical View of LLM Degeneration

Prior work has explained repetitive degeneration through the tendency of probability self-reinforcement in LLMs: as a pattern repeats in the context, a language model can assign higher probability to repeating it again (Xu et al., 2022). However, explaining how repetition reinforces itself does not by itself explain how generation first becomes trapped in a repetitive loop. We approach this question through the long-term dynamics of generation. Rather than tracking only the

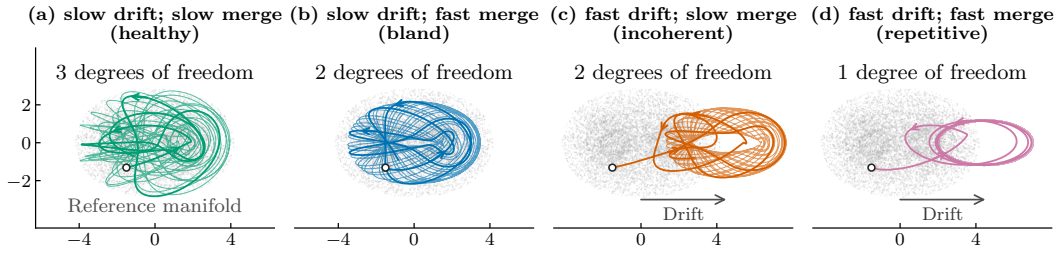


Figure 2: Drifting and merging remove different sources of variation in a dynamical toy model; together they make motion approach a periodic loop. Gray points sample the reference distribution, and each colored curve follows one trajectory from a shared initial state (open circle). Thicker segments highlight early motion, with arrows along the curves indicating forward time.

probabilities of repeated patterns, we study how an ensemble of predictive states evolves, focusing on drifting and merging. A simple dynamical system lets us examine how these two processes separately and jointly change long-term behavior.

We use a Stuart–Landau-type toy ODE with three oscillatory components to demonstrate the core idea. One component persists throughout. Of the other two, one decays as the state drifts away from a reference manifold; the other decays under strong merging, which contracts differences between initial states in that component. Figure 2 compares four settings of the drifting and merging controls, with the same equations, initial state, and reference distribution. Figure 2(a) shows the case without drifting and with weak merging, in which three independent frequencies survive. In (b), strong merging alone leaves two frequencies near the reference manifold; in (c), drifting alone also leaves two, but the state drifts away from the manifold. Both retain nonperiodic motion, despite losing different oscillatory components. When drifting and strong merging act together, only one frequency survives and motion approaches a periodic loop (d). Appendix B provides the equations and more analysis.

These regimes offer an analogy for blandness, incoherence, and repetition in text generation (Holtzman et al., 2020). Merging reduces predictive distinctions between histories, diminishing context-dependent variation. Text can then remain fluent yet become generic and less relevant to its context, suggesting a connection to blandness. Drifting, by contrast, moves generation away from natural-language structure, weakening the constraints that support semantic coherence. Text may then continue to vary without remaining coherent. The periodic motion under their combined action provides an analogy for repetition. Appendix A illustrates these behaviors in generated text. For text generation, blandness and incoherence need not be final outcomes: they may be intermediate stages as further drifting or merging leads toward repetition.

We next study how the rates of drifting and merging relate to the onset of degeneration. Blandness and incoherence often develop gradually, without a clear onset. We therefore focus on repetitive collapse, whose onset is easier to identify. The next section defines collapse time and prompt-based measures of drifting and merging.

4 Measuring Collapse, Drifting, and Merging

4.1 Collapse time

For quantitative analysis, we focus on collapse time—the onset of repetitive collapse—under greedy decoding. This strategy selects the most likely token at each step. Early studies found it prone to repetition (Holtzman et al., 2020), yet it is increasingly used in reasoning tasks. We use greedy decoding throughout the primary experiments to remove token-sampling randomness and isolate variation in collapse time across initial histories. Non-greedy decoding does not necessarily prevent degeneration, and we also observe similar trends under token sampling.

We define collapse time as the generated-token position at which persistent repetition begins, using a sliding-window version of the REP-4 metric (Welleck et al., 2019) to locate its onset. Our target is the behavior of a trajectory ensemble generated from the same LLM, not the exact collapse time of an individual rollout. For each model and generation setting, we therefore summarize collapse

times over a given set of prompts by their mean in log time, corresponding to the geometric mean in token units:

$$y = \frac{1}{N}\sum_{i=1}^{N} \log t_{\text{collapse}}^{(i)}, \quad t_{\text{collapse}}^{(i)} = \text{Onset}\left[\text{REP}_4^{(i)}\right], \quad \text{REP}_4^{(i)}(t; W) = \sum_{u=t-W+1}^{t} \frac{z_{i,u}}{W}. \quad (1)$$

Here y is the mean log collapse time over N prompts, and $t_{\text{collapse}}^{(i)}$ is the collapse time of the rollout from prompt i. Because we used greedy decoding, only one rollout is generated per prompt. The rolling repeated 4-gram score REP-4 averages over $W = 64$ positions, with $z_{i,u} = 1$ if the 4-gram ending at position u has appeared earlier in the generated text, and zero otherwise; prompt tokens are excluded. Appendix C specifies the confirmation and onset-localization procedure; rollouts without confirmed repetition are assigned the generation horizon.

4.2 Drifting time from predictive loss

We use prompt loss to construct a timescale τ_{drift} that characterizes how quickly generated text drifts from the natural-language distribution. For each prompt i, let ℓ_i be the next-token prediction loss on the prompt, normalized by decoded bytes for comparison across tokenizers (i.e., in nats per byte). In computing τ_{drift}, we use only prompt tokens, without generating the long continuations whose collapse times we seek to predict. Section 5.1 specifies which prompt tokens are used and how the losses are aggregated for each model. The connection to drifting comes from the accumulation of excess loss with text length: a larger excess loss implies a faster decrease in the relative probability of reference trajectories. Allowing a fitted offset b_c for each prompt corpus c, we define the mean excess loss $\Delta\ell_c$ and its inverse τ_{drift} for $\mathcal{L}_{\text{prompt}} > b_c$, as follows:

$$\mathcal{L}_{\text{prompt}} = \frac{1}{N}\sum_{i=1}^{N} \ell_i, \qquad \Delta\ell_c = \mathcal{L}_{\text{prompt}} - b_c, \qquad \tau_{\text{drift}} = \frac{1}{\Delta\ell_c}. \quad (2)$$

A larger τ_{drift} indicates slower drifting under this approximation. The offset b_c is fitted jointly with the collapse-time relation on the fitting conditions, and provides a proxy for the corpus entropy rate (see Appendix D for clarification).

4.3 Merging time from information persistence

Merging describes how predictions from different histories become less distinguishable over time. We probe this by replacing one prompt token with model-predicted alternatives and comparing the later predictions (Figure 1(b)). Each replacement gives a different history. We keep all subsequent prompt tokens unchanged, so differences in predictions reflect the replacement rather than different generated text. Faster convergence of these predictions indicates faster local merging.

Let Y_s denote the replacement token at position s, with probabilities given by the model. For each replacement, we compute the predictive distribution of Y_{s+h} at h tokens later ($h = 1, 2, \ldots$). If these distributions coincide, the later prediction contains no information about the replacement. Mutual information measures how much of this information remains. We normalize it by the initial token entropy and integrate the response to summarize how long predictive differences persist:

$$\tau_{\text{merge}} \equiv \int_0^\infty R_{\text{pert}}(h)\, \mathrm{d}h, \qquad R_{\text{pert}}(h) \equiv \mathbb{E}_{x,s}\left[\frac{I(Y_s; Y_{s+h} \mid x_{<s})}{H(Y_s \mid x_{<s})}\right], \qquad R_{\text{pert}}(0) = 1. \quad (3)$$

Here, x is a prompt. The average $\mathbb{E}_{x,s}$ gives equal weight to prompts and weights positions within each prompt by their initial token entropy. The information quantities refer to the replacement procedure above: the prefix $x_{<s}$ determines the replacement probabilities, and subsequent prompt tokens remain fixed. The response $R_{\text{pert}}(h)$ measures the fraction of information about the replacement remaining after h tokens. Its integral expresses persistence in token units; for example, an exponential response $e^{-h/\tau}$ has area τ.

In practice, we use the eight most probable replacement tokens at each sampled position and renormalize their probabilities. We compute mutual information through an equivalent formula based on the generalized Jensen–Shannon divergence of the predictive distributions, weighted by the replacement probabilities (Lin, 1991). After normalization and averaging, we integrate the response over

eight token lags using the trapezoidal rule. Below, τ_{merge} denotes this finite-horizon measure of local predictive persistence. Appendix E gives computation details. The next section relates the two prompt-based timescales to repetitive collapse time.

5 Modeling and Predicting Collapse Time

We model collapse as a first-passage event into persistent repetition. Its rate depends on how often opportunities for repetition arise and how likely each is to produce a persistent loop. Drifting creates opportunities by moving generation away from natural-language structure. Contextual constraints, such as a topic to develop or an unresolved point, can still guide generation away from repetition; merging weakens these constraints. Combining opportunity frequency with collapse probability gives the effective rate

$$r \propto \tau_{\text{drift}}^{-\beta_{\text{drift}}} \exp(-\beta_{\text{merge}} \tau_{\text{merge}}), \tag{4}$$

where the fitted coefficients control the dependence on the two timescales. The power-law factor represents fewer opportunities under slower drifting; the exponential factor models the probability that no effective contextual constraint remains, which decreases with longer persistence. Appendix F gives the derivation and its assumptions. Assuming a constant collapse rate within each combination of model, corpus, and attention temperature gives an exponential distribution of collapse times T, with $\mathbb{E}[\log T] = -\log r + \text{const}$. We use this relation to model the measured mean log collapse time:

$$y_{m,c,a} = \alpha + \beta_{\text{drift}} \log \tau_{\text{drift}}^{(m,c,a)} + \beta_{\text{merge}} \tau_{\text{merge}}^{(m,c,a)} + \epsilon_{m,c,a}, \tag{5}$$

where m, c, a index model, corpus, and attention temperature; $y_{m,c,a}$ is the mean log collapse time defined in Equation (1), and $\epsilon_{m,c,a}$ is the residual. Attention temperature a is not an input; instead, we study whether the effects of attention temperature is captured through the two effects.

5.1 Joint prediction

Experimental setup. We evaluate 16 base models on five corpora spanning code, mathematics, law, philosophy, and literature at $T_{\text{att}} \in \{0.5, 0.7, 1, 1.4, 2\}$, for a total of 400 conditions. Each corpus contains 60 prompts; we thus acquired 24,000 greedy rollouts. Every prompt has approximately 512 tokens: 384 context tokens and 128 tokens for measuring loss and persistence. From each prompt, we yield one rollout of up to 3,584 tokens; we average log collapse times over the prompts in each condition (Appendix G).

Attention temperature changes context use while keeping the greedy decoding rule fixed: values below one sharpen attention weights, values above one flatten them, and one recovers standard attention. We use the same temperature in every head and layer; Appendix G gives the scaling formula. Mixtral-8×22B (141B parameters) and ERNIE-4.5-300B-A47B are additional held-out models evaluated only at $T_{\text{att}} = 1$, outside these 400 conditions.

We compare our joint predictor in Equation (5) with four baseline predictors: log total parameter count (including all mixture-of-experts parameters), byte-normalized prompt loss, $\log \tau_{\text{drift}}$, and τ_{merge}. Corpus offsets and regression coefficients are fitted jointly on the fitting conditions. Attention temperature is not an input to any predictor.

Results. Prompt loss predicts collapse time better than model size ($R^2 = 0.716$ versus 0.121; Figure 1(c)), but misses important changes under attention-temperature adjustments and post-training. Figure 3(a) shows that lowering T_{att} from 1 to 0.7 delays collapse despite higher loss for all five displayed models on SEP-60; raising it to 1.4 produces the opposite pattern. Figure 3(b) shows a similar contrast: both LongWriter models (Bai et al., 2025) collapse later than their base models despite higher loss. Thus, lower prompt loss alone does not consistently imply later collapse.

Using $\log \tau_{\text{drift}}$, which incorporates fitted corpus offsets, gives $R^2 = 0.835$. Adding τ_{merge} raises this to 0.938 (Figure 1(c)). Figure 3(c) compares the joint predictions with observed collapse times across the 400 conditions, spanning more than two orders of magnitude. These results use the same conditions for fitting and evaluation. The following held-out tests examine whether the relation predicts collapse time beyond those conditions; Appendix G.3 provides fitted offsets and sensitivity checks.

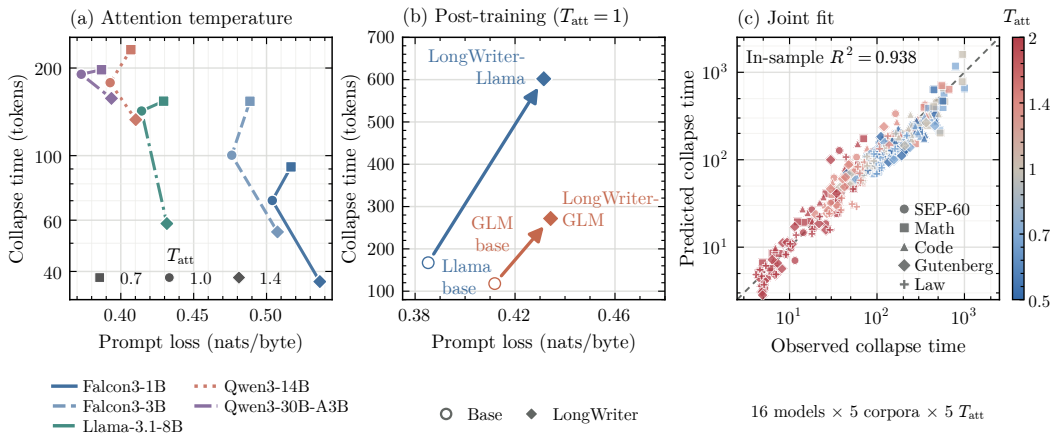


Figure 3: Prompt loss alone misses collapse-time variation captured by the joint predictor. Higher loss accompanies later collapse under **(a)** attention-temperature changes on SEP-60 and **(b)** LongWriter post-training at $T_{att} = 1$ (arrows: base to post-trained). **(c)** Observed versus predicted geometric-mean REP-4 collapse time across 400 conditions; the dashed line denotes equality. The joint fit achieves in-sample $R^2 = 0.938$ in mean log time. All panels use greedy decoding.

Table 1: Held-out prediction with parameters fitted at $T_{att} = 1$ and held fixed. **(a)** Pearson correlation of changes in mean log collapse time; the 1.0 column instead compares in-sample values. **(b)** Signed percentage error in geometric-mean collapse time over five corpora; MAE averages absolute percentage errors across models. Indented rows omit corpus offsets. Bold and underlining mark the best and second-best held-out results.

(a) Attention temperatures: Pearson's r

Predictor	1.0	0.5	0.7	1.4	2.0
Loss	0.838	0.149	-0.090	<u>0.586</u>	0.803
τ_{drift}	0.960	0.556	0.311	0.541	**0.880**
τ_{merge}	0.840	0.683	0.458	0.546	0.628
$\tau_{drift} + \tau_{merge}$	0.964	**0.750**	**0.511**	**0.597**	<u>0.861</u>
- $b_c = 0$	0.940	<u>0.734</u>	<u>0.481</u>	0.570	0.784

(b) Held-out models ($T_{att} = 1$): error (%)

Predictor	70B	72B	141B	300B	MAE
Loss	-42.7	-39.6	-37.2	-54.9	43.6
Size	-29.2	-18.9	**15.4**	<u>-11.4</u>	18.7
τ_{drift}	<u>-14.8</u>	<u>-10.9</u>	-24.2	-26.2	19.1
τ_{merge}	-16.9	-20.8	<u>-16.6</u>	**-6.6**	<u>15.2</u>
$\tau_{drift} + \tau_{merge}$	**-12.7**	**-10.5**	-20.2	-15.2	**14.7**
- $b_c = 0$	-19.8	-18.9	-21.9	-20.5	20.2

5.2 Prediction at held-out attention temperatures

Lowering attention temperature to 0.7 often delays collapse, whereas reducing it further to 0.5 can reverse this benefit in some model families. Raising it to 1.4 or 2.0 generally causes earlier collapse. To test whether the predictor captures these changes, we fit all coefficients and corpus offsets on the 16 models × 5 corpora = 80 conditions at $T_{att} = 1$ and hold them fixed for the other temperatures. Within each model–corpus pair, we compute predicted and observed changes in mean log collapse time relative to their respective values at $T_{att} = 1$.

The joint predictor achieves correlations of 0.750, 0.511, and 0.597 at held-out attention temperatures of 0.5, 0.7, and 1.4, respectively, outperforming all baseline predictors at each temperature (Table 1(a)). For comparison, τ_{drift} alone achieves correlations of 0.556, 0.311, and 0.541, respectively. Although changing attention temperature alters the generation dynamics, the relation fitted only at $T_{att} = 1$ still predicts changes in collapse time through the two measured timescales, without using temperature as an input.

At the two lower temperatures, τ_{merge} alone predicts changes in collapse time better than τ_{drift} alone, while combining them improves prediction further. At $T_{att} = 2.0$, the joint predictor achieves a correlation of 0.861, slightly below τ_{drift} alone (0.880) but above all other predictors. This suggests that drifting contributes more predictive information at this temperature. Overall, considering both effects gives the best predictive performance across the tested held-out temperatures, ranking first at three temperatures and second at the fourth without refitting.

5.3 PREDICTION FOR HELD-OUT LARGER MODELS

To test prediction for larger models, we refit all parameters using only 14 models no larger than 30B at $T_{\text{att}} = 1$, giving 70 conditions across five corpora. We then hold these parameters fixed and predict collapse times for Llama-3.1-70B, Qwen2.5-72B, Mixtral-8×22B (141B), and ERNIE-4.5-300B-A47B. For each held-out model, we compare the geometric means of predicted and observed collapse times over the five corpora.

As seen in Table 1(b), the joint predictor achieves the lowest mean absolute percentage error across the four models at 14.7%, compared with 19.1% for τ_{drift} alone and 43.6% for prompt loss. The model-size predictor is more accurate at 141B, while τ_{merge} alone is more accurate at 300B. The joint predictor underestimates collapse time for all four models. One possible explanation is that larger models exhibit subtler forms of repetition that REP-4 misses, so collapse is recorded later.

Corpus offset b_c. Table 1(b) also shows that removing corpus offsets raises the mean absolute percentage error from 14.7% to 20.2%, supporting their use for prediction on held-out larger models. The fitted offsets depend on the prompts and conditions used for fitting. Appendix G.3 reports their values, sensitivity checks, and bootstrap results.

6 INCREASING INFORMATION PERSISTENCE TO DELAY COLLAPSE

Fine-tuning objective. The drifting–merging relation suggests a direct intervention: train models to increase τ_{merge} while limiting increases in prompt loss. We fine-tune Qwen3-4B-Base and Llama-3.1-8B with LoRA on the five evaluation domains. We first extend each training prompt using the fixed base model. We then replace a token in the generated text with an alternative predicted by that model and train to increase the persistence of the resulting predictive differences. The entropy-based computation in Section 4.3 motivates a differentiable persistence objective, combined with language modeling and a KL penalty that limits changes from the base model:

$$\mathcal{L}_{\text{train}} = \mathcal{L}_{\text{LM}} + D_{\text{KL}}(p_{\text{base}} \| p_\theta) - \lambda \tau_{\text{merge}}. \tag{6}$$

The LM and KL terms preserve predictive quality. For training, we approximate the persistence term using the original and perturbed predictions; the calculation is specified in Appendix G.4. At evaluation, we used a parameter $s_{\text{LoRA}} < 1$ to scale the learned weight update, controlling the strength of fine-tuning without retraining. We measure collapse time under greedy raw continuation at $T_{\text{att}} = 1$. Evaluation prompts are excluded from training. Appendix G.4 gives the data splits, training parameters, and evaluation details.

Results. As seen in Table 2, fine-tuning delays collapse in both model families. Qwen3-4B's geometric-mean collapse time increases by $2.89\times$ at $s_{\text{LoRA}} = 0.40$, with later onset on every corpus, and by $4.96\times$ at 0.55. Llama-3.1-8B's collapse time increases by $5.52\times$ at 0.40. The Qwen model at 0.55 and the fine-tuned Llama model both collapse later than all reference models shown in the table under raw continuation. We use 0.40 as the primary Qwen setting because it changes prompt loss less.

The improvement accompanies longer information persistence, not lower prompt loss. For Qwen at $s_{\text{LoRA}} = 0.40$, τ_{merge} rises from 1.823 to 1.997 tokens while prompt loss changes only from 0.4260 to

Table 2: Fine-tuning and collapse under greedy raw continuation at $T_{\text{att}} = 1$. τ_{drift} (bytes) and τ_{merge} (tokens) are arithmetic means over corpora; t_{collapse} (tokens) is the geometric mean over prompts and corpora. Parentheses give timescale differences and collapse-time ratios relative to the displayed base.

Model / s_{LoRA}	τ_{drift} ↑	τ_{merge} ↑	t_{collapse} ↑
Base models (no fine-tuning)			
Qwen3-4B	3.29	1.82	130.4
Llama-3.1-8B	3.29	1.74	152.4
GLM4-9B	3.10	1.75	118.0
Qwen2.5-72B	4.64	1.89	293.3
Mixtral-8x22B	3.84	1.86	271.5
ERNIE-4.5-300B	5.43	1.88	475.6
LongWriter(Bai et al., 2025) Long-form SFT			
GLM4-9B	2.89 (−0.21)	1.89 (+0.14)	271.8 (2.3×)
Llama-3.1-8B	2.86 (−0.43)	2.00 (+0.26)	601.9 (3.9×)
Our method Eq. (6)			
Qwen3-4B (0.40)	3.28 (−0.01)	2.00 (+0.17)	**376.9** (2.9×)
Qwen3-4B (0.55)	3.25 (−0.04)	2.12 (+0.29)	**647.4** (5.0×)
Llama-3.1-8B (0.40)	3.27 (−0.02)	1.90 (+0.16)	**841.4** (5.5×)

0.4278 nats/byte. Separate Qwen controls using LM loss alone or with the KL term yield smaller gains than persistence training (Appendix G.4). Table 2 also shows that LongWriter increases τ_{merge} and delays collapse, but reduces τ_{drift} more than our fine-tuning. This comparison concerns raw continuation, not LongWriter's native long-writing instructions. Appendix G.4 gives the evaluation details.

Figure 4 compares code generated by Llama-3.1-8B before and after fine-tuning. The base Llama model repeatedly renames cache helpers while reusing essentially the same implementation. After fine-tuning, it constructs embeddings (Embed.), attention blocks, feed-forward layers (FFN), and an unfinished encoder (Enc.), with later components using earlier ones. The generated code still contains implementation errors. Appendix A.4 provides the unedited excerpts.

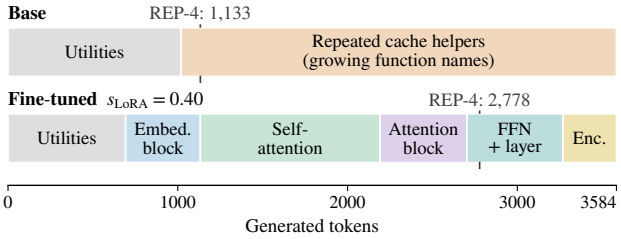


Figure 4: Llama-3.1-8B code continuation before and after fine-tuning ($s_{\text{LoRA}} = 0.40$), with the same prompt and greedy decoding at $T_{\text{att}} = 1$. Colors mark manually annotated code regions; markers show REP-4 onsets.

Appendix A shows that an overly large s_{LoRA} can lengthen τ_{merge} while producing severely incoherent text, including letter strings and mixed-language fragments. Longer persistence alone therefore does not ensure coherent generation.

7 Discussion and Limitations

Optimizing information persistence over a few token positions delays repetitive collapse over much longer continuations, although the objective does not directly optimize collapse time. How these local changes delay collapse remains unclear. Measuring drifting and merging during free generation could reveal whether the increase in persistence measured on prompts persists as context grows, and how it affects entry into repetition.

Our experiments mainly concern base models under uninterrupted greedy continuation. Applicability to models post-trained for specific response formats requires further study under their intended prompting and generation settings. The fixed-timescale model does not track how drifting and merging change as context grows. Our persistence measure also does not distinguish useful context from irrelevant token differences. Finally, our preliminary fine-tuning experiments demonstrate in-domain gains. Whether these gains generalize requires evaluation on new domains with fixed hyperparameters and broader measures of coherence and task performance.

8 Conclusion

Information persistence helps predict repetitive collapse time beyond predictive loss alone. The drifting–merging framework connects drifting from natural-language structure with the fading influence of history, helping explain why similar predictive losses can accompany different repetition behavior. Local prompt-based measurements of persistence add predictive information about repetitive collapse, and optimizing persistence alongside language-modeling constraints offers a way to delay it under greedy continuation. Persistence alone, however, does not ensure coherence, nor does delayed repetition establish factual accuracy or task success. Further implications and limitations are discussed in Appendix I.

AI use statement

In this work, we used generative AI tools to assist with experiments, literature searches and manuscript polishing. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

The appendices document the toy-model equations and analysis (Appendix B), collapse detection and persistence estimation (Appendices C and E), and the collapse-time derivation (Appendix F). Experimental settings, fitting protocols, and fine-tuning details are provided in Appendix G. We plan to release the code publicly following the public release of the paper.

References

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Cheung. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 700–710, 2022. doi: 10.18653/v1/2022.findings-acl.58.

Andrea Auconi, Benjamin M. Friedrich, and Andrea Giansanti. Fluctuation-response theorem for Kullback–Leibler divergences to quantify causation. *EPL*, 135(2):28002, 2021. doi: 10.1209/0295-5075/135/28002.

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longwriter: Unleashing 10,000+ word generation from long context LLMs. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=kQ5s9Yh0WI.

Sourya Basu, Govardana Sachitanandam Ramachandran, Nitish Shirish Keskar, and Lav R. Varshney. Mirostat: A neural text decoding algorithm that directly controls perplexity. *arXiv preprint arXiv:2007.14966*, 2020. URL https://arxiv.org/abs/2007.14966.

Ting-Rui Chiang and Yun-Nung Chen. Relating neural text degeneration to exposure bias. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 228–239. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.blackboxnlp-1.16. URL https://aclanthology.org/2021.blackboxnlp-1.16/.

Yuntian Deng, Volodymyr Kuleshov, and Alexander Rush. Model criticism for long-form text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11887–11912. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.815. URL https://aclanthology.org/2022.emnlp-main.815/.

Yihong Dong, Yuchen Liu, Xue Jiang, Bin Gu, Zhi Jin, and Ge Li. Rethinking repetition problems of LLMs in code generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 965–985. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.48. URL https://aclanthology.org/2025.acl-long.48/.

Xin Du and Kumiko Tanaka-Ishii. Correlation dimension of autoregressive large language models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 109050–109077, 2025.

Xin Du and Kumiko Tanaka-Ishii. Escaping mode collapse in LLM generation via geometric regulation. *arXiv preprint arXiv:2605.00435*, 2026. URL https://arxiv.org/abs/2605.00435. Accepted to ICML 2026.

Stephen G. Eick, William A. Massey, and Ward Whitt. The physics of the $M_t/G/\infty$ queue. *Operations Research*, 41(4):731–742, 1993. doi: 10.1287/opre.41.4.731.

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, 2018. doi: 10.18653/v1/P18-1082.

Matthew Finlayson, John Hewitt, Alexander Koller, Swabha Swayamdipta, and Ashish Sabharwal. Closing the curious case of neural text degeneration. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/34899013589ef41aea4d7b2f0ef310c1-Abstract-Conference.html.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 12848–12856, 2021. doi: 10.1609/aaai.v35i14.17520.

Surya Ganguli, Dongsung Huh, and Haim Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105(48):18970–18975, 2008. doi: 10.1073/pnas.0804451105.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. Exposure bias versus self-recovery: Are distortions really incremental for autoregressive text generation? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5087–5102. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.415. URL https://aclanthology.org/2021.emnlp-main.415/.

Zhitao He, Haolin Yang, Rui Min, Zeyu Qin, and Yi R. Fung. On stable long-form generation: Benchmarking and mitigating length volatility. *arXiv preprint arXiv:2605.01357*, 2026. URL https://arxiv.org/abs/2605.01357.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B. Brown, Prafulla Dhariwal, Scott Gray, Chris Hallacy, Benjamin Mann, Alec Radford, Aditya Ramesh, Nick Ryder, Daniel M. Ziegler, John Schulman, Dario Amodei, and Sam McCandlish. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020. URL https://arxiv.org/abs/2010.14701.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030, 2022. doi: 10.52202/068431-2176. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c1e2faff6f588870935f114ebe04a3e5-Abstract-Conference.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.

Zhe Hu, Hou Pong Chan, Jiachen Liu, Xinyan Xiao, Hua Wu, and Lifu Huang. PLANET: Dynamic content planning in autoregressive transformers for long-form text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2288–2305. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.acl-long.163. URL https://aclanthology.org/2022.acl-long.163/.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. URL https://arxiv.org/abs/2001.08361.

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. URL https://arxiv.org/abs/1909.05858.

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 284–294, 2018. doi: 10.18653/v1/P18-1027. URL https://aclanthology.org/P18-1027/.

Huayang Li, Tian Lan, Zihao Fu, Deng Cai, Lemao Liu, Nigel Collier, Taro Watanabe, and Yixuan Su. Repetition in repetition out: Towards understanding neural text degeneration from the data perspective. In *Advances in Neural Information Processing Systems*, volume 36, pp. 72888–72903, 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e6c2e85db1f1039177c4495ccd399ac4-Abstract-Conference.html.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B. Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12286–12312. Association for Computational Linguistics, 2023b. doi: 10.18653/v1/2023.acl-long.687. URL https://aclanthology.org/2023.acl-long.687/.

Henry W. Lin and Max Tegmark. Critical behavior in physics and probabilistic formal languages. *Entropy*, 19(7):299, 2017. doi: 10.3390/e19070299.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. doi: 10.1109/18.61115.

Hong Liu, Sang Michael Xie, Zhiyuan Li, and Tengyu Ma. Same pre-training loss, better downstream: Implicit bias matters for language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 22188–22214, 2023.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121, 2023. doi: 10.1162/tacl_a_00536.

Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016. URL https://arxiv.org/abs/1611.02163.

Kai Nakaishi, Yoshihiko Nishikawa, and Koji Hukushima. Phase transition in large language models and the criticality of natural languages. *arXiv preprint arXiv:2406.05335*, 2024. doi: 10.48550/arXiv.2406.05335. URL https://arxiv.org/abs/2406.05335.

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

Chau Minh Pham, Simeng Sun, and Mohit Iyyer. Suri: Multi-constraint instruction following for long-form text generation. *arXiv preprint arXiv:2406.19371*, 2024. URL https://arxiv.org/abs/2406.19371.

Bowen Ping, Jiali Zeng, Fandong Meng, Shuo Wang, Jie Zhou, and Shanghang Zhang. LongDPO: Unlock better long-form generation abilities for LLMs via critique-augmented stepwise information. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 7613–7632. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.395. URL https://aclanthology.org/2025.findings-acl.395/.

Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. Observational scaling laws and the predictability of language model performance. In *Advances in Neural Information Processing Systems*, volume 37, pp. 15841–15892, 2024.

Lucas Rudelt, Daniel González Marx, Michael Wibral, and Viola Priesemann. Embedding optimization reveals long-lasting history dependence in neural spiking activity. *PLOS Computational Biology*, 17(6):e1008927, 2021. doi: 10.1371/journal.pcbi.1008927.

Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Claude E. Shannon. Prediction and entropy of printed english. *Bell System Technical Journal*, 30 (1):50–64, 1951. doi: 10.1002/j.1538-7305.1951.tb01366.x.

Alan D. Sokal. Monte carlo methods in statistical mechanics: Foundations and new algorithms. In *Functional Integration: Basics and Applications*, pp. 131–192. Springer, 1997. doi: 10.1007/978-1-4899-0319-8_6.

Zechen Sun, Yuyang Sun, Zecheng Tang, Juntao Li, Wenpeng Hu, Wenliang Chen, Zhunchen Luo, Guotong Geng, and Min Zhang. IS-CoT: Breaking the long-form generation collapse via interleaved structural thinking. *arXiv preprint arXiv:2606.09709*, 2026. URL https://arxiv.org/abs/2606.09709.

Shuntaro Takahashi and Kumiko Tanaka-Ishii. Evaluating computational language models with scaling properties of natural language. *Computational Linguistics*, 45(3):481–513, 2019. doi: 10.1162/coli_a_00355.

Bowen Tan, Zichao Yang, Maruan Al-Shedivat, Eric Xing, and Zhiting Hu. Progressive generation of long text with pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4313–4324. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.341. URL https://aclanthology.org/2021.naacl-main.341/.

Kumiko Tanaka-Ishii. Repeated sequences reveal gaps between large language models and natural language. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8367–8382. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.379. URL https://aclanthology.org/2026.acl-long.379/.

Petar Veličković, Christos Perivolaropoulos, Federico Barbero, and Razvan Pascanu. Softmax is not enough (for sharp size generalisation). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 61190–61211, 2025. URL https://proceedings.mlr.press/v267/velickovic25a.html.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*, 2019. URL https://arxiv.org/abs/1908.04319.

Gian Wiher, Clara Meister, and Ryan Cotterell. On decoding strategies for neural text generators. *Transactions of the Association for Computational Linguistics*, 10:997–1012, 2022. doi: 10.1162/tacl_a_00502.

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, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/148c0aeea1c5da82f4fa86a09d4190da-Abstract-Conference.html.

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. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.66. URL https://aclanthology.org/2023.emnlp-main.66/.

Kevin Yang, Dan Klein, Nanyun Peng, and Yuandong Tian. DOC: Improving long story coherence with detailed outline control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.190. URL https://aclanthology.org/2023.acl-long.190/.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023a. URL https://proceedings.neurips.cc/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. How language model hallucinations can snowball. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59670–59684, 2024. URL https://proceedings.mlr.press/v235/zhang24ay.html.

James Xu Zhao, Jimmy Z. J. Liu, Bryan Hooi, and See-Kiong Ng. How does response length affect long-form factuality. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3102–3125. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.161. URL https://aclanthology.org/2025.findings-acl.161/.

Wenhong Zhu, Hongkun Hao, and Rui Wang. Penalty decoding: Well suppress the self-reinforcement effect in open-ended text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1218–1228, 2023. doi: 10.18653/v1/2023.emnlp-main.78.

Yifan Zhu, Guanting Chen, Bing Wei, and Haoran Luo. HiFlow: Hierarchical feedback-driven optimization for constrained long-form text generation. *arXiv preprint arXiv:2603.04996*, 2026. URL https://arxiv.org/abs/2603.04996.

A EXAMPLES OF TEXT DEGENERATION

These greedy continuations illustrate repetition, blandness (little substantive progression), and incoherence, complementing the schematic in Figure 2. Incoherence ranges from broken semantic connections to a breakdown of word and sentence structure. We selected base-model conditions by their measured timescales, then inspected the generated passages. The base-model runs use 512-token SEP-60 prompts. Qwen and Mistral have 3,584-token rollouts; the saved Pythia and full-strength adapter examples have 1,536-token rollouts. Table 3 reports condition means, with base-model drifting times using the main fit's fixed SEP-60 offset; timescale comparisons are relative to the same-corpus panel. Blue text gives the final 36 words of the actual prompt, black text the generated excerpt, and ellipses mark omissions.

Table 3: Conditions for the qualitative examples, ordered by their presentation. τ_{drift} is in decoded bytes and τ_{merge} in model tokens. The tuned row uses Qwen3-4B-Base with the 150-step adapter at scale 1.0; its eight-token merging estimate fixes candidates and weights from the base model and pools responses across prompts, unlike the base-model panel. The dash denotes an unavailable byte-normalized drifting time.

Examples	Model	Corpus	T_{att}	τ_{drift}	τ_{merge}
Exact loop	Qwen3-0.6B-Base	SEP-60	2.0	1.112	0.769
Template / bland	Mistral-Small-24B	SEP-60	1.0	4.084	1.715
Incoherent 1/2	Pythia-2.8B	SEP-60	0.5	2.568	1.972
Incoherent 3	Qwen3-4B + adapter	Law	1.0	—	3.244

A.1 REPETITIVE

For an *Idealism* prompt, Qwen generates:

Repetitive 1: exact loop *Idealism*

Prompt (tail) ... reality, or even exhaustive of reality, and although the existence of something independent of the mind is conceded, everything that we can know about this mind-independent "reality" is held to be so permeated by the creative

...

and the same, and the same, and the same, and the same, and the same, and the same, and the same, and the same, and the same, and the same

The phrase persists through the rollout, with REP-4 onset at token 4, illustrating the condition with short drifting and merging times.

Repetition can also preserve a template while changing a few words. Later in the Mistral continuation shown in Bland 1, consecutive sections reuse the same opening:

Repetitive 2: template repetition *Idealism*

Prompt (tail) ... reality, and although the existence of something independent of the mind is conceded, everything that we can know about this mind-independent "reality" is held to be so permeated by the creative, formative, or constructive activities of

...

4. Idealism as a Moral Doctrine The third way in which the term "idealism" is used is as a label for a moral doctrine.

...

5. Idealism as an Aesthetic Doctrine The fourth way in which the term "idealism" is used is as a label for an aesthetic doctrine.

...

6. Idealism as a Political Doctrine The fifth way in which the term "idealism" is used is as a label for a political doctrine.

The ordinals and doctrine names change, but the sentence frame remains fixed; intervening paragraphs likewise recycle explanations and qualifications. This template repetition escapes REP-4 detection throughout the 3,584-token rollout.

A.2 BLAND

The prompt from *Idealism* contrasts conceptions of idealism and the positions to which they are opposed. Mistral continues:

Bland 1 *Idealism*

Prompt (tail) ... reality, and although the existence of something independent of the mind is conceded, everything that we can know about this mind-independent "reality" is held to be so permeated by the creative, formative, or constructive activities of

...

In any case, it is clear that the term "idealism" is used in a variety of ways, and that it is not always clear what is meant by it. This is not to say that the term "idealism" is meaningless, but rather that it is used in different ways, and that it is not always clear which of these ways is meant in a given context. It is also clear that the term "idealism" is often used in a polemical way, and that this is one of the reasons why it is not always clear what is meant by it.

The passage remains grammatical and on topic, but repeatedly reformulates the term's ambiguity without developing the distinction. It precedes the template repetition shown above.

For *Chance versus Randomness*, the same model offers generic hedging:

Bland 2 *Chance versus Randomness*

Prompt (tail) ... are incorrect. Below, some of the theoretical principles that philosophers have extracted from commonplace beliefs about chance will be outlined. Two such constraints have been widely accepted since the early days of the philosophy of probability.

...

These constraints are not necessarily mutually exclusive, and it is possible that some combination of them may be jointly sufficient to pick out a unique kind of probability, namely chance. However, it is also possible that no combination of these constraints is jointly sufficient to pick out a unique kind of probability, and that chance is a kind of probability that is not reducible to any other kind of probability.

The passage presents opposite possibilities without giving a criterion for deciding between them.

A.3 INCOHERENT

For another *Idealism* prompt, Pythia generates:

Incoherent 1 *Idealism*

Prompt (tail) ... or even exhaustive of reality, and although the existence of something independent of the mind is conceded, everything that we can know about this mind-independent "reality" is held to be so permeated by the creative, form

...

Indeed, the term "idealism" is used in a very different way, namely as a term of praise, to characterize the philosophical outlook of the ancient Greeks and Romans. The term "idealism" is used in this way because of the fact that the ancient Greeks and Romans were not idealists, but rather they were realists.

The sentences are individually readable, but the second explains the attribution by denying it: semantic connection fails while word and sentence structure remain intact.

In a separate *Normativity in Metaethics* continuation, the same condition produces an assertion followed immediately by its denial:

Incoherent 2 *Normativity in Metaethics*

Prompt (tail) ... will be true about some of the questions we consider here. For example, take the question of the relationship between the normative and the non-normative. The case of the Open Question Argument directly illustrates that what

...

Consider the case of the "Normative Explanandum" in terms of the "Normative Explanans" – i.e., the claim that some action is N. This is a claim that can be non-normative for a speaker, but normative for a hearer. But it is not the case that some action is non-normative for a speaker, but normative for a hearer.

There is no intervening argument or qualification connecting the two claims.

The next example shows more severe incoherence, extending to word and sentence structure. A Qwen3-4B adapter trained with the persistence objective for 150 steps produces this breakdown at full adapter strength (1.0), compared with the reduced strength 0.40 used in the main case study. In the Law corpus, the generated text breaks down mid-sentence while discussing constitutional interpretation:

Incoherent 3 *Law: judgment 1305*

Prompt (tail) ... true that such things would never be, but one is concerned to know if such a doing would be possible." "The Constitution gives so many assurances in Part III that it would be difficult to think

...

```
The words 'shall not be' omitted hereunder sccouldn t beemal 1
123emeta 1l/ 1em s tdel e s o 1 1/ 2 1bracket s t ype sh a xt ight
s h a oundt s o n ounde
```

The adapter raises the Law merging estimate from 1.742 to 3.244 tokens, yet the excerpt loses recognizable word and sentence structure. The rollout later meets REP-4 at token 800; the displayed passage illustrates the incoherence preceding that repetition.

A.4 GENERATION AFTER FINE-TUNING FOR INFORMATION PERSISTENCE

For the code case in Figure 4, Llama-3.1-8B and its fine-tuned adapter ($s_{\mathrm{LoRA}} = 0.40$) continue the same 512-token prompt under raw greedy decoding at $T_{\mathrm{att}} = 1$, with recorded REP-4 onsets of 1,133 and 2,778, respectively, within 3,584 generated tokens. This pair was selected after inspecting

saved outputs, not sampled at random. The unedited excerpts are not matched token windows; [. . .] marks omissions outside the listings.

Base model *Code*

```
[...]
def _get_encoder_decoder_cache_mask_from_cache(cache):
[...]
    cache_mask = []
    for i in range(len(cache.encoder_outputs)):
        cache_mask.append(_expand_mask(cache.decoder_outputs[i].
    new_ones(cache.decoder_outputs[i].shape[1], dtype=torch.bool)))
    return cache_mask
[...]
def _get_encoder_decoder_cache_mask_from_cache_with_encoder(
    encoder_outputs, cache):
[...]
    cache_mask = []
    for i in range(len(cache.encoder_outputs)):
        cache_mask.append(_expand_mask(encoder_outputs[i].new_ones(
    encoder_outputs[i].shape[1], dtype=torch.bool)))
    return cache_mask
[...]
```

τ_{merge} **fine-tuning** $s_{\text{LoRA}} = 0.40$

```
[...]
class XLMLayer(nn.Module):
    def __init__(self, config, is_decoder=False, is_chunked=False,
    chunk_size=1):
        super().__init__()
        self.chunk_size_feed_forward = chunk_size
        self.is_chunked = is_chunked
        self.is_decoder = is_decoder
        self.self = XLMAttention(config, is_decoder=is_decoder,
    is_chunked=is_chunked, chunk_size=chunk_size)
        self.intermediate = XLMIntermediate(config)
        self.output = XLMOutput(config)
[...]
        attention_output = self_outputs[0]
        if self.is_chunked:
            attention_output = [o.chunk(self.chunk_size_feed_forward
    , dim=0) for o in attention_output]
        intermediate_output = self.intermediate(attention_output)
        if self.is_chunked:
            intermediate_output = [o.chunk(self.
    chunk_size_feed_forward, dim=0) for o in intermediate_output]
        layer_output = self.output(intermediate_output,
    attention_output)
        if self.is_chunked:
            layer_output = [o.chunk(self.chunk_size_feed_forward,
    dim=0) for o in layer_output]
        outputs = (layer_output,) + self_outputs[1:]  # add
    attentions if we output them
        return outputs
[...]
```

The base output repeats cache-helper code with increasingly long names, whereas `XLMLayer` in the tuned output instantiates and calls previously defined components through `self.intermediate` and `self.output`. The example shows continued construction of connected components, not

correct code. The tuned output contains implementation errors and an unfinished encoder; its REP-4 onset does not mark a clear semantic loop.

B A STUART–LANDAU-TYPE TOY MODEL

This appendix specifies the noise-free model used in Section 3. Drifting and merging suppress different oscillations, reducing the number of independent surviving frequencies. All four conditions use the same equations, initial state, and reference distribution; only the two control parameters differ.

B.1 SIX REAL STATE VARIABLES AND TWO CONTROLS

The state is $\xi = (s, \theta, x, y, a, b)$, where $s \in [0, 1]$ records the extent of drifting from the reference, $\theta \in \mathbb{R}/2\pi\mathbb{Z}$ is a carrier phase, and $(x, y), (a, b) \in \mathbb{R}^2$ are two oscillator pairs. Drifting damps the first pair; merging contracts differences in the second, which serves as a proxy for history-dependent predictive distinctions. The autonomous equations are

$$\dot{s} = d(1 - s), \qquad s(0) = 0, \tag{7}$$

$$\dot{\theta} = \omega_0, \tag{8}$$

$$\begin{aligned} \dot{x} &= (1 - 2s - x^2 - y^2)x - \omega y, \\ \dot{y} &= (1 - 2s - x^2 - y^2)y + \omega x, \end{aligned} \tag{9}$$

$$\begin{aligned} \dot{a} &= (\alpha - \kappa - a^2 - b^2)a - \nu b, \\ \dot{b} &= (\alpha - \kappa - a^2 - b^2)b + \nu a. \end{aligned} \tag{10}$$

The controls are the drift rate $d \geq 0$ and damping parameter κ. We compare $d = 0$ with $d = 0.05$, and weak merging $\kappa_{\mathrm{w}} = 0.2$ with strong merging $\kappa_{\mathrm{s}} = 1.4$, keeping

$$(\omega_0, \omega, \nu) = \chi(1, \sqrt{2}, \sqrt{3}), \qquad \chi = \tfrac{1}{4}, \qquad \alpha = 0.9. \tag{11}$$

All conditions start at $s(0) = 0$, radii $r_u(0) = 1$ and $r_v(0) = \sqrt{0.7}$, and the same three phases. The multiplier χ reduces overplotting by slowing rotations without changing their ratios or the drifting and radial dynamics.

Writing $u = x + \mathrm{i}y$ and $v = a + \mathrm{i}b$ gives

$$\dot{u} = (1 - 2s + \mathrm{i}\omega - |u|^2)u, \qquad \dot{v} = (\alpha - \kappa + \mathrm{i}\nu - |v|^2)v. \tag{12}$$

Each oscillator has the Stuart–Landau form $\dot{z} = (\lambda + \mathrm{i}\Omega - |z|^2)z$, with polar equations

$$\dot{r} = (\lambda - r^2)r, \qquad \dot{\psi} = \Omega. \tag{13}$$

For constant $\lambda > 0$, the origin is unstable and nonzero amplitudes approach the radially stable value $\sqrt{\lambda}$; for $\lambda < 0$, the origin is globally exponentially stable. At $\lambda = 0$, nonzero amplitudes decay algebraically to zero. The phase equation applies when $r > 0$. The carrier retains unit amplitude.

B.2 REFERENCE DISTRIBUTION AND DRIFT TIMESCALE

Let $m_{\mathbb{S}^1}$ be uniform angular measure and σ_r uniform measure on the radius-r circle in $\mathbb{R}^2$. The fixed reference law is

$$\mu_{\mathrm{ref}} = \delta_0 \otimes m_{\mathbb{S}^1} \otimes \sigma_1 \otimes \sigma_{\sqrt{0.7}}, \tag{14}$$

with factors ordered as $(s, \theta, (x, y), (a, b))$. It is invariant at $d = 0$, $\kappa = \kappa_{\mathrm{w}}$: both radii are at equilibrium and rotations preserve uniform phase measures. At $d = 0$, either merging setting preserves the (s, θ, x, y) marginal, but strong merging changes the (a, b) law. Thus zero drift need not preserve the full joint reference distribution.

The drifting coordinate satisfies

$$s(t) = 1 - e^{-dt}, \qquad \tau_{\mathrm{drift}}^{\mathrm{toy}} = d^{-1} \quad (d > 0). \tag{15}$$

Smaller positive d slows drifting without changing its endpoint; $d = 0.05$ gives timescale 20. For the trajectory empirical measure $\widehat{\mu}_T = T^{-1} \int_0^T \delta_{\xi(t)}\, dt$ and observable $h(\xi) = s$,

$$\int h\, d\widehat{\mu}_T - \int h\, d\mu_{\mathrm{ref}} = 1 - \frac{1 - e^{-dT}}{dT}, \qquad d > 0. \tag{16}$$

This discrepancy grows from zero toward one, measuring drifting in the s observable independently of the plotted projection. It is not a distance between the full distributions and does not detect changes due solely to merging.

B.3 Merging and contraction timescale

Consider two toy states representing different histories, sharing (s, θ, x, y) but differing in $c = (a, b)^{\mathsf{T}}$ and $\widetilde{c}$. Their dynamics are $\dot{c} = (\alpha - \kappa)c + \nu J c - \|c\|^2 c$, where $J = \left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$. For $\Delta = c - \widetilde{c}$, skew-symmetry of J gives

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\|\Delta\|^2 &= (\alpha - \kappa)\|\Delta\|^2 - \Delta^{\mathsf{T}}(\|c\|^2 c - \|\widetilde{c}\|^2 \widetilde{c}) \\ &\le (\alpha - \kappa)\|\Delta\|^2. \end{aligned} \tag{17}$$

The inequality follows from convexity of $\|c\|^4/4$. For $\kappa > \alpha$,

$$\|c(t) - \widetilde{c}(t)\| \le e^{-(\kappa - \alpha)t}\|c(0) - \widetilde{c}(0)\|, \qquad \tau_{\mathrm{merge}}^{\mathrm{toy}} = (\kappa - \alpha)^{-1}. \tag{18}$$

Under strong merging, this bound gives an exponential contraction rate of 0.5 and timescale 2; distinct states approach one another asymptotically. The Hopf threshold of the v subsystem is $\kappa = \alpha$: below it, nonzero v approaches a sustained oscillation; above it, v decays to zero. At κ_{w}, histories on the equilibrium circle retain their phase difference and distance, so the finite contraction timescale does not apply. These toy timescales describe the controls, rather than numerically identifying the measured proxies τ_{drift} and τ_{merge}.

B.4 Drift-induced damping and degrees of freedom

Drifting changes the growth rate of u:

$$\lambda_u(t) = 1 - 2s(t) = -1 + 2e^{-dt}. \tag{19}$$

At $d = 0$ it remains $+1$. For $d > 0$, it crosses zero at $t_* = (\log 2)/d$ and tends to -1, so u vanishes asymptotically. With s held fixed, the u subsystem has a Hopf threshold at $s = 1/2$. Along the drifting trajectory, the amplitude already decreases before t_* from the specified initial radius. Crossing t_* makes the linear growth rate negative; it does not mark the start of decay or finite-time extinction. The Hopf threshold describes the fixed-s subsystem, not a bifurcation of the full system along its trajectory.

We summarize the evolving state by the drifting coordinate D and squared amplitudes S, C, normalizing the latter to their reference values:

$$D = s, \qquad S = x^2 + y^2, \qquad C = \frac{a^2 + b^2}{\alpha - \kappa_{\mathrm{w}}}. \tag{20}$$

Table 4 lists their limits and the surviving frequencies in the main figure's panel order. Strong merging removes v in both drift settings; drifting additionally removes u, leaving only the carrier when both act together.

The chosen frequencies are rationally independent, so the three- and two-frequency limits are quasiperiodic; the one-frequency limit is periodic with period $2\pi/\omega_0 = 8\pi$. In the full state space, the limiting motion densely explores a phase torus whose dimension equals the DoF count. It counts surviving phase directions, rather than state variables or displayed coordinates. A finite trajectory segment is still a curve, and the observation map below is not an embedding of the full state. The incoherent analogue is non-reference quasiperiodic motion, not chaos.

Table 4: Long-time limits and degrees of freedom (DoF), counted as independent surviving frequencies. Regime names refer to analogies with text generation.

Drift / merging	$(D_\infty, S_\infty, C_\infty)$	Frequencies	Analogue	DoF
Zero / weak	$(0, 1, 1)$	ω_0, ω, ν	Healthy	3
Zero / strong	$(0, 1, 0)$	ω_0, ω	Bland	2
On / weak	$(1, 0, 1)$	ω_0, ν	Incoherent	2
On / strong	$(1, 0, 0)$	ω_0	Repetitive	1

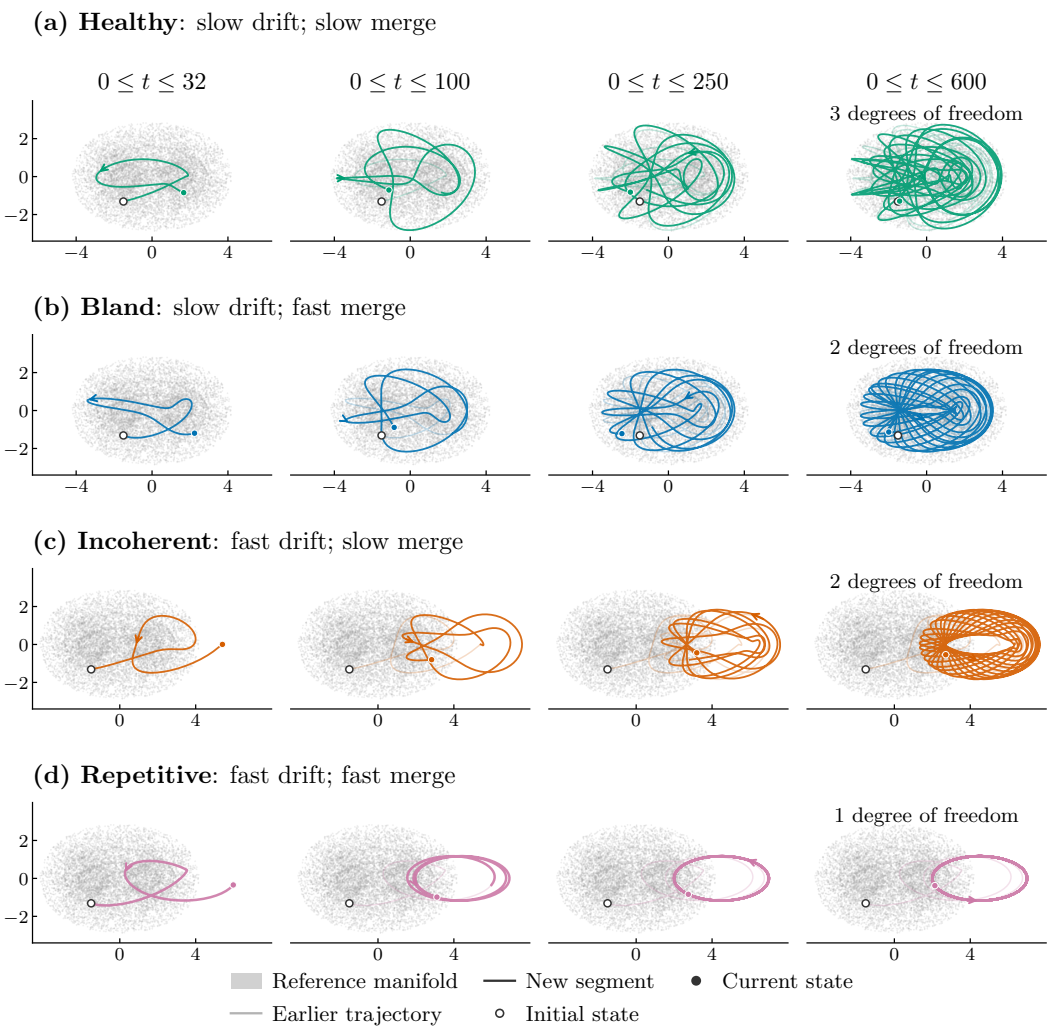


Figure 5: Time evolution under four drifting–merging conditions. Columns extend the trajectory from the shared initial state to the indicated time. Darker segments show the newly added interval; paler segments retain earlier motion. Open and filled circles mark the initial and current states, respectively. Gray points show the projected reference distribution; the final column reports the long-time degrees of freedom.

B.5 Observation map and plotted coordinates

All figures use a fixed nonlinear observation map followed by an orthographic projection. Define

$$\rho = R + x + \beta a, \qquad z = y + \beta b, \qquad G(\xi) = (\rho \cos\theta, \rho \sin\theta, z)^\mathsf{T}, \tag{21}$$

with $R = 2.5$ and $\beta = 0.8$. For viewing azimuth $\phi = -55^\circ$ and elevation $\eta = 28^\circ$, add displacement Ase_{h}, where $A = 4.5$ and $e_{\mathrm{h}} = (-\sin\phi, \cos\phi, 0)^{\mathsf{T}}$. The projected coordinates are

$$H(\xi) = \rho \sin(\theta - \phi) + As,$$
$$V(\xi) = -\rho \sin\eta \cos(\theta - \phi) + z\cos\eta. \tag{22}$$

The carrier generates circulation, the oscillators shape the cross-section, and s produces displacement. These display parameters do not alter the dynamics. Gray reference samples are the pushforward of Equation (14). Since the projection is not invertible, overlap does not imply equal underlying distributions; drifting is measured by Equation (16).

Figure 2 shows $t \in [0, 600)$, with open circles marking the shared initial state, thicker curves highlighting $t \in [0, 100]$, and arrows along curves indicating forward time. All panels use the same map and axis scales, with horizontal windows centered separately for zero-drift and drifting conditions. Separate arrows labeled "Drift" indicate displacement, not trajectory paths.

B.6 Numerical implementation

We integrate the six real equations with DOP853, relative tolerance 10^{-10}, absolute tolerance 10^{-12}, maximum step 0.25, horizon 900, and output spacing 0.05. Initial phases and $16{,}000$ reference samples come from the stored seed-271828 NumPy realization, retaining the carrier, structural, and first content phases. The initial state and reference samples are shared across conditions. Rendering applies Equation (22) to the saved states without resimulation.

B.7 Time evolution

Figure 5 follows the same trajectories as Figure 2, with unchanged dynamics and angular velocities. Each row shows one condition over progressively longer intervals $[0, 32]$, $[0, 100]$, $[0, 250]$, and $[0, 600]$. The first interval shows early motion; longer later intervals reveal the quasiperiodic or periodic form. In each column, the newly added segment is emphasized and previous segments remain pale, retaining the path from the initial state. Drifting displaces the trajectory from the reference, while merging reduces its sustained variation; together they make the trajectory approach a periodic loop. All panels use the same reference and projection, with fixed coordinate scales and horizontal windows centered separately for the two drift settings.

C REP-4 Collapse Detection

Equation (1) marks a generated 4-gram as repeated if it has appeared earlier in the generated text, excluding the prompt; overlapping occurrences count. Positions are one-based, the first three repetition indicators are zero, and rolling scores use only complete windows. The detector first confirms persistence, then searches backward to locate onset:

1. Find the first run of 256 consecutive endpoints satisfying $\mathrm{REP}_4(t; 64) > 0.7$, and let s be its first endpoint.
2. Within $[s - 48, s]$, find the earliest endpoint a satisfying $\mathrm{REP}_4(a; 16) > 0.7$.
3. Report the one-based left edge of this shorter window,

$$t^{\mathrm{REP4}}_{\mathrm{collapse}} = a - 15. \tag{23}$$

The long persistence requirement excludes brief repetitions, while the shorter window localizes when the episode began. If no episode is confirmed within the rollout, its horizon is assigned as the operational collapse time. This includes episodes that begin near the horizon but cannot yet be confirmed. All regression outcomes use this same convention.

D Predictive Loss and the Drifting Timescale

Let P_c be a stationary reference distribution over byte sequences in corpus c and Q_θ the model distribution. Assuming the entropy and KL rates are finite and the model assigns positive probability

on the reference support, the cross-entropy rate in nats per byte decomposes as

$$\ell_c(Q_\theta) = h_c + d_c(Q_\theta), \qquad d_c(Q_\theta) = \lim_{B\to\infty} \frac{1}{B} D_{\mathrm{KL}}(P_c(X_{1:B}) \,\|\, Q_\theta(X_{1:B})), \tag{24}$$

where h_c is the reference entropy rate. Assume also that the normalized log likelihood ratio converges to $d_c(Q_\theta)$ for P_c-almost every reference sequence. Then its length-B prefix x satisfies

$$\log \frac{Q_\theta(x)}{P_c(x)} = -B\big(\ell_c(Q_\theta) - h_c\big) + o(B). \tag{25}$$

Thus excess loss determines how quickly reference text loses relative probability under the model. For $d_c(Q_\theta) > 0$, let B_{pass} be the first byte length at which $\log(P_c/Q_\theta)$ reaches a threshold $A > 0$. When A is large enough that fluctuations and overshoot are small relative to A, this gives the characteristic passage length below. Replacing the population excess loss with the positive prompt-based adjusted loss then gives

$$B_{\mathrm{pass}} \approx \frac{A}{d_c(Q_\theta)} \approx \frac{A}{\mathcal{L}_{\mathrm{prompt}} - b_c} = A\tau_{\mathrm{drift}}. \tag{26}$$

Subtracting a corpus-specific offset is motivated by the distinction between the reference corpus's intrinsic uncertainty and the model's excess loss. We fit b_c with the coefficients of the joint predictor in place of the unknown entropy rate, requiring $\mathcal{L}_{\mathrm{prompt}} - b_c > 0$ wherever the predictor is applied; b_c is a predictive calibration parameter, not an independent entropy estimate. This construction motivates the inverse-loss timescale and the opportunity frequency in Section 5. The connection to repetitive collapse is a modeling assumption: the likelihood-ratio calculation follows reference text, whereas collapse occurs during free generation.

E Information Persistence Estimation

E.1 Finite-horizon calculation

For the main prediction experiments, at each selected prompt position s, retain the model's top $K = 8$ candidate tokens in $\mathcal{C}_s$ and renormalize their probabilities to weights w_k. Each candidate replaces the original token while subsequent prompt tokens remain fixed (Figure 6, top). For $h \geq 1$, let $p^{(k)}_{s,h} = Q_\theta(\cdot \mid x_{<s}, k, x_{s+1:s+h-1})$ be the full-vocabulary distribution predicting position $s + h$ on branch k. The normalized generalized Jensen–Shannon divergence estimates the information response:

$$\widehat{R}_{\mathrm{pert},s}(h) = \frac{H\Big(\sum_{k\in\mathcal{C}_s} w_k p^{(k)}_{s,h}\Big) - \sum_{k\in\mathcal{C}_s} w_k H\Big(p^{(k)}_{s,h}\Big)}{H(w)}, \qquad \widehat{R}_{\mathrm{pert},s}(0) = 1. \tag{27}$$

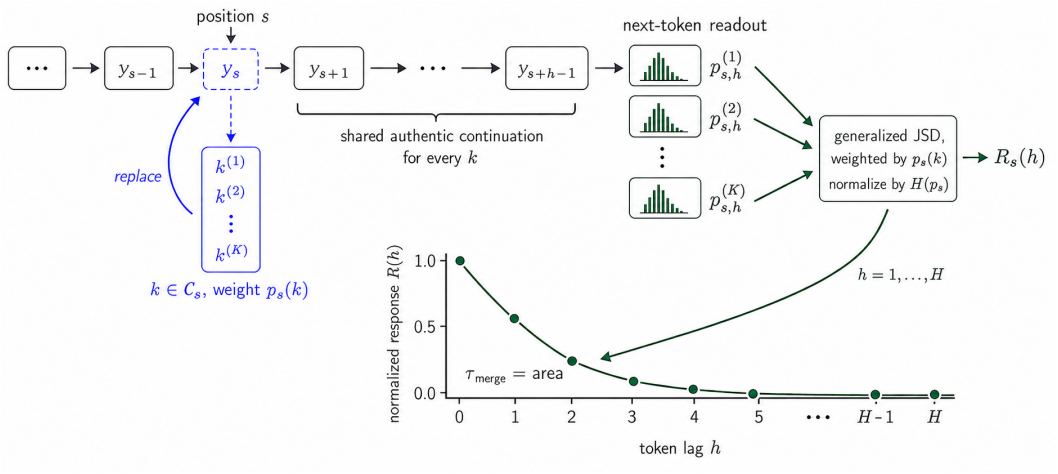
For each fixed prompt and position, define the intervention law by $\Pr(Y_s = k, Y_{s+h} = v) = w_k p^{(k)}_{s,h}(v)$. The numerator is exactly $I(Y_s; Y_{s+h})$ under this law (Lin, 1991), rather than under free generation or posterior conditioning on the fixed suffix. Dividing by $H(w) > 0$ normalizes it to $[0, 1]$; at $h = 0$, the readout is Y_s itself, which gives response one. Zero-entropy positions carry no weight. Positions are weighted by candidate entropy within each prompt i, then prompts equally:

$$\widehat{R}_{\mathrm{pert}}(h) = \frac{1}{N}\sum_{i=1}^{N} \frac{\sum_s H(w_{is})\widehat{R}_{\mathrm{pert},is}(h)}{\sum_s H(w_{is})}. \tag{28}$$

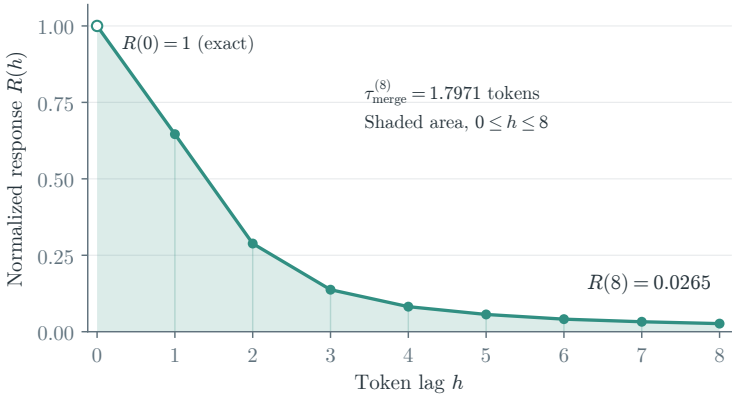
With unit token spacing and linear interpolation between measured lags, the eight-lag proxy is the trapezoidal integral, with $H = 8$:

$$\widehat{\tau}^{(H)}_{\mathrm{merge}} = \frac{1}{2}\widehat{R}_{\mathrm{pert}}(0) + \sum_{h=1}^{H-1} \widehat{R}_{\mathrm{pert}}(h) + \frac{1}{2}\widehat{R}_{\mathrm{pert}}(H). \tag{29}$$

Reported τ_{merge} values denote $\widehat{\tau}^{(8)}_{\mathrm{merge}}$, not an extrapolation to infinite lag. The pooling implementation omits prompts with nonpositive or nonfinite total candidate entropy (N counts retained prompts) and clips the pooled response to $[0, 1]$ for numerical stability, without imposing monotone decay. For fixed candidate sets, the entropy-based calculation is differentiable in the branch probabilities and motivates the persistence objective in Section 6.



(a)



(b)

Figure 6: Estimating merging time from token perturbations. (a) Token alternatives induce different predictions along shared prompt text. (b) Measured response for Llama-3.1-8B on SEP-60 at $T_{\text{att}} = 1$. Shading gives the eight-lag integral; lines connect measurements rather than a fitted decay model.

E.2 An empirical response curve

Figure 6 (bottom) shows the response for Llama-3.1-8B on 60 SEP-60 prompts at $T_{\text{att}} = 1$. It falls rapidly at early lags, from $R_{\text{pert}}(1) = 0.646$ to $R_{\text{pert}}(4) = 0.082$, followed by a slower tail reaching $R_{\text{pert}}(8) = 0.027$. The shaded area gives $\tau_{\text{merge}} = 1.797$ tokens, the value used for this condition in the main analysis. The nonzero endpoint also illustrates why the finite-horizon proxy need not capture all longer-lag persistence.

F Derivation of the Collapse-Time Relation

The model in Section 5 combines collapse opportunities with the probability that no contextual constraint prevents a loop. To motivate the latter factor, use a continuous approximation to token time and suppose contextual constraints become active as a homogeneous Poisson process of rate λ per token. Their lifetimes are independent copies of $L > 0$ with finite mean, independent of the arrival process. At stationarity, the active constraint count satisfies (Eick et al., 1993)

$$N \sim \text{Poisson}\left(\lambda \int_0^\infty \Pr(L > h)\, \mathrm{d}h\right) = \text{Poisson}(\lambda \mathbb{E}[L]), \qquad \Pr(N = 0) = e^{-\lambda \mathbb{E}[L]}. \tag{30}$$

To connect lifetime to information persistence, idealize each constraint as revealing an initial token choice while active and returning a distinct erasure symbol afterward, with lifetime independent of

the token choice. Its normalized information response is then its survival probability:

$$R_{\text{pert}}(h) = \Pr(L > h), \qquad \tau_{\text{merge}} = \int_0^\infty R_{\text{pert}}(h)\,\mathrm{d}h = \mathbb{E}[L], \qquad \Pr(N = 0) = e^{-\lambda \tau_{\text{merge}}}. \tag{31}$$

If any active constraint prevents a loop and an opportunity produces collapse when none remains, this model gives an exponentially decreasing collapse probability per opportunity. Using the measured finite-horizon proxy in place of $\mathbb{E}[L]$ assumes that it tracks constraint lifetimes across conditions; the fitted β_{merge} combines the arrival rate and an approximate proportionality between measured persistence and mean lifetime.

For the opportunity frequency, suppose each fixed increment of drifting creates an opportunity. Their frequency then scales as $1/\tau_{\text{drift}}$ in byte units; the fitted exponent β_{drift} generalizes this to a power law. The joint predictor uses τ_{drift} in decoded bytes and τ_{merge} in model tokens. Fixed unit factors enter the coefficients; treating the same scaling as an opportunity rate per token additionally assumes that variation in bytes per generated token can be neglected or absorbed by the empirical calibration.

If opportunities form a homogeneous Poisson process independent of the constraints, and their constraint states can be approximated as independent stationary draws, independent thinning gives a collapse rate equal to the opportunity rate times $\Pr(N = 0)$. Combining this factor with the drifting dependence yields Equation (4). Without this independence approximation, stationary occupancy alone does not imply a constant first-passage hazard. Under the approximation, the uncensored continuous collapse time T is exponential, and substituting $u = rT$ gives

$$\mathbb{E}[\log T] = -\log r + \underbrace{\int_0^\infty e^{-u} \log u\,\mathrm{d}u}_{-\gamma}, \qquad \gamma \text{ is Euler's constant.} \tag{32}$$

Absorbing this constant into the intercept gives the mean log-time form of Equation (5). More generally, it holds for $T = Z/r$ when $Z > 0$ has a condition-independent distribution and $\mathbb{E}[|\log Z|] < \infty$. Applying this form to the reported REP-4 times is an empirical approximation: finite-horizon assignment and confirmation can alter mean log time and are not included in this first-passage derivation. The constraint and opportunity processes supply a mechanistic hypothesis, rather than separately measured components of the model's trajectories.

G Experimental Details and Supplementary Results

G.1 Models, corpora, and evaluation

The main experiment uses 16 base models from six families: Falcon3 (1B, 3B, 10B), Llama (3.2-1B, 3.1-8B, 3.1-70B), Pythia (2.8B, 12B), Mistral (24B), Qwen (0.6B, 4B, 8B, 14B, 30B-A3B, 72B), and GLM4-9B. The five corpora are code, Gutenberg literature, law, mathematics, and SEP-60 philosophical prose. Crossing models and corpora with five attention temperatures gives 400 conditions and approximately 24,000 greedy rollouts. Prompts contain 512 model tokens; rollouts allow up to 3,584 generated tokens. GLM uses the matched subset described below; the other models use 60 prompts per corpus. Prompt loss is aggregated by averaging per-prompt byte-normalized losses or by dividing total NLL by total decoded bytes, according to the model's evaluation implementation. GLM uses the latter.

The held-out temperature experiment fits the 80 default-temperature conditions and evaluates the other 320. The held-out model experiment instead fits 14 models no larger than 30B at the default temperature (70 conditions) and evaluates Llama-70B, Qwen-72B, Mixtral-8×22B (141B nominal), and ERNIE-4.5-300B-A47B (20 conditions). The two external models use the same 60 documents per corpus as the larger-model evaluation and do not enter the 400-condition fit. Each held-out experiment refits all parameters on its own fitting conditions and then holds them fixed for evaluation.

Attention-temperature scaling. Prior work uses attention-temperature scaling for long-context language modeling and attention-based retrieval and reasoning (Peng et al., 2024; Veličković et al., 2025). We use it to vary contextual dependence while keeping the greedy decoding rule unchanged. For an attention head with causal mask M, we apply

$$\text{Attention}_{T_{\text{att}}}(Q, K, V) = \text{softmax}\Big(QK^\top/(T_{\text{att}}\sqrt{d_k}) + M\Big)\,V. \tag{33}$$

The same temperature is used in every head and layer: values below one sharpen attention, values above one flatten it, and one recovers standard attention.

G.2 Matched GLM and LongWriter comparison

GLM4-9B and LongWriter–GLM use the same 32 prompts per corpus at all five attention temperatures, giving 1,600 rollouts across the two checkpoints. Both use greedy raw continuation and the REP-4 detector in Appendix C. Prompt loss pools scored NLL and decoded bytes; merging time uses the entropy-weighted eight-lag response on the same prompts. The drifting values in Table 2 use the fixed default-temperature offsets below. GLM base enters the main regression; LongWriter remains outside it.

Table 5: Matched GLM4-9B and LongWriter–GLM across attention temperatures. Merging time is averaged over five corpora; collapse time is the geometric mean over 160 prompts per checkpoint and temperature. All times are in model tokens. Ratios compare LongWriter with the base model.

T_{att}	Base τ_{merge}	LongWriter τ_{merge}	Base time	LongWriter time	Ratio
0.5	1.814	1.953	147.6	284.7	1.9×
0.7	1.793	1.920	142.4	312.0	2.2×
1.0	1.753	1.891	118.0	271.8	2.3×
1.4	1.646	1.838	65.7	174.6	2.7×
2.0	1.052	1.679	10.3	55.7	5.4×

LongWriter has longer merging times and collapses later at every temperature. Its prompt loss is higher at the first four temperatures but lower at $T_{\text{att}} = 2$. Thus post-training delays repetition across these settings, while the reversal of the usual loss–collapse relationship occurs only over part of the temperature range. The comparison measures the combined effect of post-training, not an isolated change in merging.

G.3 Corpus offsets and sensitivity

Table 6 combines the default-temperature corpus offsets with their variability under model resampling. The offsets increase broadly from code to expository prose and literature, consistent with differences in predictive uncertainty. They are fitted adjustments rather than independent entropy estimates; in particular, Code's boundary value does not imply zero entropy. The pooled 400-condition fit and the smaller-model fit each estimate their own offsets using only their training conditions.

Each bootstrap replicate samples 16 models with replacement, retaining all five corpus conditions for each sampled model, and refits coefficients and offsets with 30 optimization starts. The training-only constraint $0 \le b_c \le 0.9 \min_{\text{fit},c} \mathcal{L}_{\text{prompt}}$ is recomputed for each sample. The table reports marginal 2.5th–97.5th percentiles. These intervals describe variability with model composition, conditional on the observed corpora and outcomes, rather than prompt-level uncertainty.

Additional checks. Using a stricter training cutoff of eight models no larger than 8B gives held-out $R^2 = 0.800$ on 70B/72B and 0.226 on 141B/300B, compared with 0.850 and 0.475 under the main 30B cutoff. Removing one training model at a time from the default-temperature 8B fit yields defined predictions with 70B/72B R^2 between 0.743 and 0.828. Offsets are more sensitive to changing the temperature coverage: fitting all five temperatures under the 8B cutoff produces one nonpositive adjusted loss among 50 held-out 70B/72B conditions. Training-set positivity therefore does not guarantee valid extrapolation.

Alternative feature transformations can also fit the data closely: untransformed drifting and merging times achieve essentially the same full-sample fit as the proposed relation and a better absolute-level fit at unseen temperatures, whereas logarithmic drifting performs better in the small-to-large comparison. The data support using drifting and merging times together, but do not identify a unique functional form.

Table 6: Corpus offsets from the joint predictor fitted on 16 models at $T_{\text{att}} = 1$, with 1,000 model-cluster bootstrap replicates. Values and percentile intervals are in nats per byte. Code's fitted offset reaches the nonnegativity boundary.

Corpus	Fitted b_c	95% percentile interval
Code	0.0000	[0.0000, 0.0466]
Mathematics	0.0542	[0.0407, 0.0949]
Law	0.0674	[0.0432, 0.1162]
SEP-60	0.0939	[0.0523, 0.1361]
Gutenberg	0.1656	[0.0318, 0.2255]

G.4 Fine-tuning setup and evaluation

Training setup. For each of Qwen3-4B-Base and Llama-3.1-8B, we split the 60 prompts per corpus into 30 training and 30 evaluation prompts. The 150 training prompts are pooled across the five corpora to train one shared adapter per model. Each training prompt is extended by greedy generation from the frozen base model. At selected continuation positions, we separately replace the token with the highest-ranked alternative among that model's top candidates, keeping subsequent tokens fixed. The LM and KL terms use reference prompt tokens; the persistence term compares clean and perturbed predictions. For training, we approximate persistence using equally weighted binary JSD between the current model's original and perturbed predictions, divided by $\ln 2$. We sum responses from lag one to the training horizon, with half weight on the last lag, and average over four replacement positions. Gradients pass through the perturbed prediction only; the LM and KL terms each retain unit weight. We use AdamW for 150 steps at learning rate 10^{-4} and rank-eight LoRA updates to the query, key, value, and output projections. The persistence weight and training horizon are $(\lambda, H) = (0.4, 8)$ for Qwen and $(0.2, 32)$ for Llama. Evaluation uses the eight-lag estimator in Equation (29).

Evaluation. Table 2 evaluates the Qwen3-4B and Llama-3.1-8B base/adapter pairs on 30 prompts per corpus, held out from gradient-based training. Qwen2.5-72B, Mixtral-8×22B, and ERNIE-4.5-300B-A47B use the same split. LongWriter–GLM and its base use 32 matched prompts per corpus; LongWriter–Llama uses 60, so its changes relative to the displayed Llama base are descriptive rather than paired.

Adapter strength s_{LoRA} multiplies the learned update after standard LoRA scaling: $W = W_{\text{base}} + s_{\text{LoRA}} \Delta W_{\text{LoRA}}$. One applies the full update; smaller values attenuate it toward the base model without retraining. We use $s_{\text{LoRA}} \in \{0.40, 0.55\}$ for Qwen and 0.40 for Llama. This multiplier is distinct from the objective weight λ. Some evaluation prompts are used to select training settings and update strengths, but not for gradient updates.

Metric aggregation. Collapse time is the geometric mean across evaluated prompts and corpora. Drifting time is the arithmetic mean over corpora of $1/(\mathcal{L}^{(c)}_{\text{prompt}} - b_c)$, using the fixed default-temperature offsets in Table 6; merging time is also averaged arithmetically. Per-corpus loss pools total scored NLL and bytes. Loss/response sample counts per corpus are 8/8 for the Qwen4 and Llama8 base/adapter pairs, 30/30 for Qwen72, and 30/8 for Mixtral and ERNIE. GLM base and LongWriter–GLM use 32 prompts for all metrics; LongWriter–Llama uses 60. For the Qwen and Llama pairs, replacement candidates and their weights come from the frozen base model, and responses are pooled by candidate entropy across positions and prompts before integration. Timescale differences are computed before rounding, and collapse-time ratios use the displayed values.

Decoding condition. The table uses greedy raw continuation at $T_{\text{att}} = 1$, with the same generation horizon as the main panel. Rollouts without confirmed repetition are assigned that horizon, as in Appendix C. A separate diagnostic gives LongWriter later REP-4 onset with its native long-writing instruction. That setting is excluded because the instruction and wrapper change the generation policy; the table compares raw continuation, not general long-writing quality.

LM and KL controls. In a separate Qwen evaluation with a 1,536-token generation horizon, LM-only and LM+KL training increase geometric-mean collapse time by $1.21\times$ and $1.02\times$, respectively,

compared with $2.47\times$ for persistence training. The controls use 300 training steps and the persistence model uses 150; this comparison is separate from the longer-generation results in Table 2.

H Extended Related Work

H.1 Mode Collapse and Text Degeneration

Mode collapse is a distribution-level failure: a generator covers too narrow a range of outcomes even when individual samples appear plausible. The problem is most familiar from generative adversarial networks, where training methods aim to prevent different inputs from producing the same few outputs (Goodfellow et al., 2014; Salimans et al., 2016; Metz et al., 2016). Precision–recall evaluations likewise separate sample quality from coverage of the data distribution (Sajjadi et al., 2018). We use the same broad idea of diversity loss, but study trajectories that unfold over time rather than a set of independent samples.

The corresponding literature in language modeling studies *neural text degeneration*. Greedy and beam decoding can produce repetitive text even when a model assigns high likelihood to natural-language data (Holtzman et al., 2020). Degeneration includes exact token loops, repeated syntactic templates, and restatements that make no semantic progress (Dong et al., 2025; Du & Tanaka-Ishii, 2026). These failures are usually measured through output quality, repetition frequency, or the presence of a loop. Our work instead asks how the diversity of the trajectory ensemble changes before a persistent loop appears.

Existing explanations identify several mechanisms behind repetition. Fu et al. (2021) derive a "high-inflow" effect in a Markov generation model: many tokens can point with high probability to the same next token, making a return more likely. Other accounts emphasize self-reinforcement after a string has begun to repeat (Xu et al., 2022) or repeated material in the training corpus (Li et al., 2023a). These mechanisms explain why a loop forms or becomes difficult to leave. We ask a complementary quantitative question: how do prompt-based measurements predict the typical time to persistent repetition across a set of rollouts?

H.2 Long-Horizon Autoregressive Generation

Long-horizon generation has received growing attention as language models are used for long-form writing, extended reasoning, and multi-step tasks (Pham et al., 2024; Bai et al., 2025; Yao et al., 2023b). Long-context understanding and long-form generation are distinct abilities. A model may retrieve information from a long input yet fail to produce a comparably long and coherent document. LongWriter reports that models with context windows beyond 100,000 tokens often stop after about 2,000 generated words (Bai et al., 2025). Long generation poses a separate challenge because every generated token becomes part of the context for all later predictions.

Repeated conditioning on model outputs creates failures that context length and next-token loss alone do not fully characterize. Models with similar predictive loss can behave differently once decoding begins (Liu et al., 2023; Wiher et al., 2022). Exposure-bias research studies the shift from natural-language histories during training to model-generated histories at inference time (Chiang & Chen, 2021; Arora et al., 2022). Early deviations can compound, but models can also recover rather than accumulate errors monotonically (He et al., 2021). Over longer outputs, generation may become repetitive, converge to a fixed template, stop early, or remain locally fluent while losing global structure (Deng et al., 2022). Factual errors can also snowball (Zhang et al., 2024), and factual precision often declines with response length (Zhao et al., 2025). These studies show how model-generated histories affect output quality. We focus on predicting the onset of persistent repetition, rather than treating it as a measure of every form of degeneration.

One line of work targets degeneration directly through training or decoding. Training methods use unlikelihood objectives or synthetic repetition data (Welleck et al., 2019; Xu et al., 2022). Decoding methods use probability truncation (Fan et al., 2018; Holtzman et al., 2020; Finlayson et al., 2024), repetition penalties (Keskar et al., 2019; Zhu et al., 2023), locally typical sampling (Meister et al., 2023), contrastive decoding (Li et al., 2023b), or comparisons with earlier predictive distributions (Xu et al., 2023). Specialized methods target structural repetition in code (Dong et al., 2025) or regulate the value cache during generation (Du & Tanaka-Ishii, 2026).

Table 7: Related measures of dependence and information persistence. Rows summarize the cited formulations, not limits on their possible extensions. τ_{merge} combines model-supported token replacements, a predictive-distribution readout, and a finite-lag integral. MI denotes mutual information; JSD denotes Jensen–Shannon divergence; KL denotes Kullback–Leibler divergence.

Measure	Variation	Readout	Statistic	Lag summary
Observational dependence				
Lagged token MI (Lin & Tegmark, 2017)	Observed sequence	Token pair	MI	Lag curve
Neural information timescale (Rudelt et al., 2021)	Observed history	Past and current spiking	MI	Timescale
Integrated autocorrelation (Sokal, 1997)	Observed trajectory	Scalar observable	Correlation	Integrated time
Perturbation-based measures				
Context ablation (Khandelwal et al., 2018)	Remove/shuffle context	Next-token loss	Cross-entropy	Loss vs. context range
Fisher memory (Ganguli et al., 2008)	Infinitesimal input change	State distribution	Fisher information	Curve and total capacity
Information response (Auconi et al., 2021)	Infinitesimal state change	Later state distribution	KL ratio	Response curve
τ_{merge} (ours)	Model-supported token replacement	Predictive distribution	Entropy-normalized MI	Finite-lag integral

Another line of work improves long-form output by adding data or structure. Long-output supervised fine-tuning and preference learning teach models to produce longer responses (Pham et al., 2024; Bai et al., 2025; Ping et al., 2025). Planning methods use coarse-to-fine representations, content plans, or detailed outlines (Tan et al., 2021; Hu et al., 2022; Yang et al., 2023). Other systems decompose a long task or add reasoning, tools, feedback, branching, and backtracking (Bai et al., 2025; Yao et al., 2023b;a). These methods change training, decoding, or trajectory structure to improve the generated output. Our goal is complementary: to relate drifting and merging to collapse time under uninterrupted generation with a fixed decoding rule. The joint predictor uses prompt-based measurements to predict mean log collapse time across rollouts.

H.3 Language Complexity and Autoregressive Dynamics

Natural language has statistical structure far beyond the next token. Entropy rate measures its irreducible uncertainty (Shannon, 1951), while scaling relations describe vocabulary growth, fluctuations, and long-range dependence across text. These scaling properties have been used to evaluate whether language models reproduce the global structure of natural language (Takahashi & Tanaka-Ishii, 2019). More recently, repeated-subsequence statistics have revealed systematic differences between the long-range organization of natural-language and LLM-generated texts, even when both are locally fluent (Tanaka-Ishii, 2026).

Generation-time studies ask how this structure changes when a model repeatedly conditions on its own outputs. Studies of sampling temperature examine repetitive and incoherent regimes (Basu et al., 2020; Nakaishi et al., 2024). Correlation-dimension analyses measure long-range structure in autoregressive trajectories and link collapsed outputs to lower dimension and persistent directions in the value cache (Du & Tanaka-Ishii, 2025; 2026). These studies provide dynamical descriptions of degeneration and motivate our focus on drifting and merging. We test whether measurements of these two processes on prompts jointly predict collapse time.

Scaling laws offer a complementary way to relate models, data, and training. Next-token predictive loss follows simple relations with model size, data, and compute (Kaplan et al., 2020; Henighan et al., 2020; Hoffmann et al., 2022). Related work uses observational scaling relations to predict broader model performance (Ruan et al., 2024). These laws mainly concern prediction loss or task performance. Our joint predictor instead combines drifting and merging to predict repetitive collapse time.

Several existing measures quantify how dependence or information persists over time. Lagged token mutual information, neural information timescales, and integrated autocorrelation are observational (Lin & Tegmark, 2017; Rudelt et al., 2021; Sokal, 1997). Perturbation-based alternatives include context ablation, Fisher memory, and information response (Khandelwal et al., 2018; Ganguli et al., 2008; Auconi et al., 2021). Our merging measurement changes one model-supported token, holds the intervening text fixed, and reads the complete later predictive distribution. Section 4.3 and Table 7 give the detailed design.

Table 7 distinguishes these measures by their perturbations, readouts, and summaries over time. Observational measures capture dependence in recorded sequences, rather than the response to a controlled token change; autocorrelation further depends on the chosen scalar observable. Context ablation measures how removing or shuffling context changes next-token loss. Fisher memory measures the sensitivity of a noisy system's state distribution to infinitesimal changes in past inputs; its memory curve also yields an integrated capacity. Information response normalizes the KL change in a later state distribution by the KL cost of the perturbation in the small-perturbation limit. Its fluctuation-response formulation can also express this response through unperturbed statistics under the stated assumptions. Thus, perturbation-based definitions need not require experimental interventions.

Different from the perturbation-based approach of Fisher memory and information response (Ganguli et al., 2008; Auconi et al., 2021), our method uses finite token replacements and later predictive distributions rather than infinitesimal changes in a continuous state or input. Relative to context ablation (Khandelwal et al., 2018), we examine how predictive differences persist over time rather than how removing context changes prediction loss. We connect this persistence to drifting to predict repetitive collapse time and test persistence-targeted fine-tuning.

I Extended Discussion and Limitations

Beyond repetitive collapse. The examples in Appendix A help distinguish repetition from other forms of degeneration. Table 3 shows that the selected bland condition has longer τ_{drift} and shorter τ_{merge} than the base-model incoherent condition. This is consistent with retaining language structure while losing context-dependent variation. The overly strong fine-tuning update shows the opposite risk: longer τ_{merge} can accompany a breakdown of word and sentence structure. These selected examples support the qualitative interpretation of drifting and merging, but do not establish onset criteria for blandness or incoherence.

Implications for long-horizon generation. The attention-temperature and post-training results show that prompt loss alone does not account for variation in collapse time. Measuring drifting and merging during free generation could test whether their changes precede repetition and whether their relative importance changes as context grows.

Limitations and scope. The current persistence measure does not distinguish useful context from irrelevant token differences. Studying which differences are retained is therefore important when using persistence as a training objective. The quantitative analysis also focuses on uninterrupted greedy generation. Sampling, instruction-guided generation, and interactive feedback can change both processes; applying the framework to these settings requires testing the relation again rather than assuming the fitted coefficients remain valid.