WHEN DOES DECODING ORDER AFFECT COVERAGE IN DIFFUSION LANGUAGE MODELS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Unlike an autoregressive language model, a diffusion language model can choose which positions to fill next. Prior work reports that forcing diffusion language models to decode from left to right can improve repeated-sampling coverage. For dLLMs, coverage (Pass@k) matters because it shows how effectively repeated sampling finds successful solutions for test-time scaling and reinforcement-learning exploration. We ask whether left-to-right position order causes the gain. Our central insight is that changing order also changes which tokens commit together, what context they can use, when the decoder resolves uncertain choices, and how sampling temperature affects token choices. We hold model weights and the decoding loop fixed across three open diffusion language models. The left-to-right advantage is strongest with one token per step at low temperature, varies across model–benchmark pairs, and can disappear or reverse as width or temperature increases. At width two, left-to-right decoding commits neighboring tokens together, so the right-hand token cannot use its newly sampled predecessor. Avoiding these adjacent co-commits recovers much of the LLaDA/HumanEval coverage loss, with a compatible but less certain result on Dream. We also observe positive coverage changes without fixed left-to-right priority. Confidence-order decoding at a higher temperature reaches comparable coverage, while committing selected positions earlier yields weaker positive estimates. When we keep the number of early commitments roughly fixed, moving them later lowers coverage and raises single-sample accuracy. Thus, decoding order combines several decoder choices rather than changing position order alone.

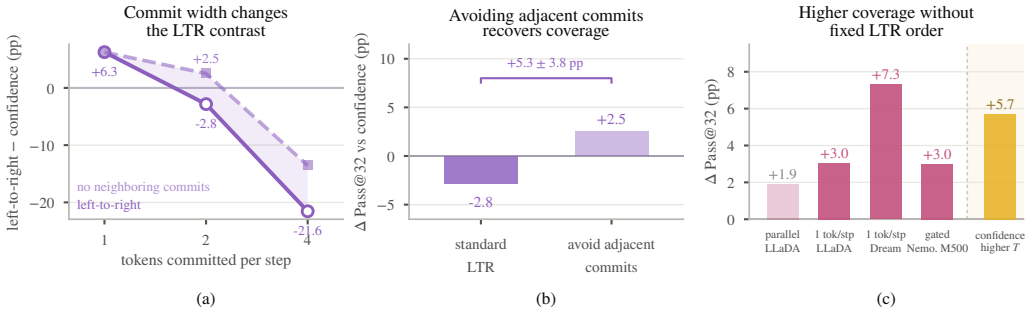

Figure 1: What looks like an order effect can come from width, local context, timing, or temperature. (a) On LLaDA/HumanEval, the left-to-right (LTR) contrast shrinks and reverses as commit width increases. (b) At width two, avoiding adjacent co-commits recovers 5.3 ± 3.8 percentage points of Pass@32, although the control also changes some pairwise order. (c) Positive coverage estimates can arise without fixed LTR priority, through candidate-first commitment or confidence-order decoding at a higher sampling temperature. Every bar's interval is tabulated in Tables 1, 13 and 15.

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.

1 INTRODUCTION

Unlike an autoregressive language model, a diffusion language model (dLLM) does not have to generate the next token on the right. At every decoding step, it can choose which still-masked positions to fill. Most practical samplers use *confidence order*: they first commit positions where the model is most confident in the sampled token. Researchers choose this rule at inference time and can change it without retraining the model. Prior work studies how position selection affects reasoning accuracy (Kim et al., 2025; Ye et al., 2025a) and speed (Ben-Hamu et al., 2025; Wu et al., 2026). Here we study its effect on coverage under repeated sampling.

Pass@k asks whether at least one of k independently sampled completions succeeds (Chen et al., 2021). It therefore matters when systems spend test-time compute on generating several attempts (Brown et al., 2024; Snell et al., 2025). Ni et al. (2026) report that forcing a dLLM to fill positions in left-to-right (LTR) order can improve Pass@k over arbitrary-order decoding. If LTR position order itself causes better coverage, arbitrary-order generation may have a genuine limitation, which could justify training models to behave more autoregressively.

The comparison, however, changes more than where the decoder starts. Consider four masked positions. If the decoder commits one token per step, LTR fills position 1, reruns the model, and lets position 2 use the token just sampled at position 1. If it commits two tokens per step, LTR selects positions 1 and 2 from the same masked state. The decoder chooses both before either sampled token enters the context. Confidence order might instead select positions 1 and 4, exposing a different pattern of context on the next step. Thus, LTR versus confidence order is not necessarily a pure test of position order.

One token per step. $[\square_1][\square_2][\square_3][\square_4] \to [A][\square_2][\square_3][\square_4] \to [A][B][\square_3][\square_4]$. The second decision sees A.
Two tokens per step. $[\square_1][\square_2][\square_3][\square_4] \to [A][B][\square_3][\square_4]$. The decoder samples A and B from the same old state, so neither can inform the other.

We therefore treat decoding order as a bundle of coupled choices. These choices determine where the decoder prefers to commit, how many positions it commits before rerunning the model, which positions commit simultaneously, what sampled context is then available, how randomly the decoder samples tokens, and when it forces uncertain choices. We use *order effect* only to describe a measured decoder contrast, not to claim that position order alone causes it.

Our experiments follow this ambiguity step by step. We map where the reported LTR advantage appears across models, tasks, commit widths, and temperatures, test the most direct width-two explanation, and ask whether higher coverage requires LTR priority at all. Throughout we hold model weights and the decoding loop fixed, and we evaluate LLaDA-8B, Dream-7B, and Nemotron-3B with a diagnostic rather than a decoder-optimization objective. The lesson is not that order is irrelevant, but that its components must be separated before it motivates a mechanistic claim or a training-time constraint.

Our contributions are:

- We map the boundary of the LTR coverage effect. Across three open dLLMs, it is strongest with one token per step and low temperature, appears with a positive interval in six of ten evaluated model–benchmark pairs, ranging from $+2.8$ to $+9.3$ pp, and shrinks or reverses as width or temperature increases.
- We show why parallel LTR is not a pure order comparison. At width two, it commits adjacent tokens before either can see the other. Avoiding these co-commits recovers much of the LLaDA/HumanEval loss, with a compatible but less certain result on Dream, although the control also changes some pairwise order.
- We show that higher coverage does not require fixed LTR priority. Raising sampling temperature can produce comparable coverage, while earlier commitment yields weaker positive estimates. Holding the promotion count roughly fixed, the timing of commitment still contributes to the measured difference. Targeted selectors do not clearly beat random promotion, and diversity does not distinguish the competitive decoders.

2 BACKGROUND AND DECODING SETUP

2.1 DIFFUSION DECODING AS REPEATED COMMITMENT

At any moment, some output positions contain tokens and the rest remain masked. The model predicts a token distribution at each masked position. A decoder chooses which positions to commit and which tokens to sample there. It fixes those tokens, reruns the model with the new context, and repeats. The decoder's *commit width* is the number of positions it fixes in one iteration.

A masked diffusion LM answers a prompt x by filling a length-L sequence s that initially contains only mask symbols. Decoding proceeds over blocks of size B. The *active block* is the earliest block that still contains a mask. We hold this semi-autoregressive block structure fixed, as deployed dLLM samplers do, so every comparison here isolates order *within* a block. Our confidence-order baseline is therefore not globally arbitrary order; it already fills blocks left to right. Write $\mathcal{M}_B(s)$ for its masked positions. At each step, the model scores every masked position, $p_i(v \mid s) = p_\theta(y_i{=}v \mid x, s)$. We summarize distributional uncertainty with entropy $H_i(s) = -\sum_v p_i(v \mid s) \log p_i(v \mid s)$. A selection policy picks w positions from $\mathcal{M}_B(s)$. The decoder samples their tokens at temperature T and fixes them. The commit width $w = L/\text{steps}$ sets how many positions commit per step. Decoding is sequential at $w{=}1$ and parallel at $w{\geq}2$. Nemotron's gated mode instead commits $\{i \mid c_i(s) \geq \gamma\} \cup \{\arg\max_i c_i(s)\}$.

2.2 POSITION PRIORITY AND COMMIT WIDTH

Each policy assigns position i a priority score $\sigma_i(s)$ and commits the w highest-scoring positions in the active block,

$$C_\sigma(s) \;=\; \operatorname*{arg\,max}_{A \subseteq \mathcal{M}_B(s),\ |A| = \min(w,\, |\mathcal{M}_B(s)|)} \;\sum_{i \in A} \sigma_i(s), \tag{1}$$

so confidence order and forced left-to-right order are $\sigma_i^{\text{conf}}(s) = c_i(s)$ and $\sigma_i^{\text{ar}}(s) = -i$ respectively. Here $c_i(s)$ is the probability the model assigns to the token that the decoder commits at i. It is not necessarily the maximum of the distribution, although the two coincide at $T{=}0$. No model uses top-p or top-k truncation in any run.

Position priority and commit width are different knobs. Priority decides *which* positions the decoder prefers. Width decides *how many* positions it commits before updating its predictions. At width one, priority determines a commitment sequence. At larger widths, it also groups positions that the decoder samples from the same old state. Those positions cannot condition on one another, even if the decoder preserves their overall left-to-right priority. Sampling temperature is a third knob. It controls how randomly the decoder samples the token at each selected position.

2.3 CONFIDENCE SCORES AND SAMPLING TEMPERATURE

Each model retains its official confidence definition. All of them score a position by the probability of the token the decoder just sampled there, so temperature always affects the ranking indirectly by changing which token is scored. The models differ in the distribution doing the scoring. LLaDA and Nemotron read that probability off untempered logits, so temperature leaves the scoring distribution itself unchanged. Dream reads it off tempered logits, so temperature additionally flattens the scoring distribution and pushes its ranking toward uniform. Dream is therefore the harder case for any claim that temperature substitutes for order, and we treat it as such in Section 5.1.

2.4 COVERAGE METRIC AND UNCERTAINTY

For a problem with n samples of which c pass the verifier we use the unbiased estimator

$$\widehat{\text{pass@}k} \;=\; 1 - \binom{n-c}{k} \Big/ \binom{n}{k}, \tag{2}$$

averaged over problems (Chen et al., 2021). Every decoder comparison uses a paired problem-level bootstrap (10^4 resamples) of the difference in Eq. (2). Most tables report percentage-point differences as $\delta \pm h$, with h half the width of the 95% percentile interval; this summarizes the width only, and exclusion and bolding use the actual endpoints, which the core diagnostic tables print. Higher is better throughout.

3 DIAGNOSTIC INTERVENTIONS

3.1 REMOVING ADJACENT CO-COMMITMENT

At width two, standard LTR selects adjacent pairs, $(1, 2)$, then $(3, 4)$, and so on. The decoder samples both tokens before rerunning the model, so the right-hand token cannot use its newly sampled predecessor. We test whether this loss of local context explains part of the coverage drop. Confidence order can also select neighbors, but does not require it.

Our *non-adjacent LTR* control preserves a leftward preference but prevents neighbors from committing in the same step. We call this policy *strided LTR* in the code and artifact. It is identical to LTR at w=1; at w=2, it selects spaced rather than adjacent positions. This is not a pure adjacency intervention. Changing the pairs also changes some pairwise order, and 26% of commits precede their left neighbor (Section B.1). The control therefore tests a concrete local-context explanation without fully isolating it.

3.2 MOVING SELECTED POSITIONS EARLIER

We next test whether coverage can improve without a fixed LTR schedule. Our deliberately simple intervention moves a small set of uncertain positions ahead of the confidence queue and otherwise retains confidence order. We call it *candidate-first* in the paper and `fork_first` in the artifact. It changes when selected positions commit, but does not claim that they are semantic reasoning branches.

The detector flags positions whose top prediction begins one of 26 connective stems and that lie within 32 tokens of the decoded frontier. Candidate-first prioritizes flagged positions by entropy and otherwise follows confidence order; if none are flagged, the two rules select the same positions. The exact set and scoring definitions appear in Section A.3. The detector mostly flags enumerative stems rather than classical inferential connectives (Section B.2), reinforcing its role as an operational perturbation.

3.3 MOVING PROMOTIONS IN TIME

We call a commit a *promotion* when candidate-first selects a position that confidence order would not select at that step. We ask whether moving these promotions in time changes coverage.

Why simple delay is confounded. Our delay rule requires a candidate to remain eligible for d consecutive steps, so d=0 is candidate-first and large d approaches confidence order. It does not isolate timing, because a position can stop meeting the candidate rule during the hold and never be promoted at all, which moves the count as well as the time (Section 5.3).

Approximately count-matched timing control. The second control instead holds candidate positions below all non-candidates for the first S steps of each block, then promotes at most one candidate at the first opportunity at or after step S. The number of blocks containing a candidate then mainly determines the count, while S shifts when promotions occur. We match counts empirically rather than by construction and report the realized counts in Section 5.3.

Alternative selectors replace the dictionary with learned influence, entropy, or random selection. Their thresholds target similar candidate-flag rates but do not equalize realized promotions (Section 5.4 and Table 5).

3.4 OBSERVATIONAL PROBES OF INFLUENCE

Before intervening on timing, we ask whether the commit events each policy selects look capable of redirecting later predictions and final answers. These probes describe events that each policy chooses. They motivate hypotheses and do not identify causal effects.

Within-block influence. Suppose a position had taken its second-choice token rather than its first. At a commit event for position i in state s, let a and b be the two most probable tokens at i and

$R = \mathcal{M}_B(s) \setminus \{i\}$ the rest of the block. We measure

$$I(i,s) \;=\; \frac{1}{|R|} \sum_{j \in R} D_{\mathrm{KL}}\big(p_j(\cdot \mid s^{i \leftarrow a}) \,\big\|\, p_j(\cdot \mid s^{i \leftarrow b})\big)\,, \qquad |R| > 0, \tag{3}$$

where $s^{i \leftarrow v}$ is the state after committing v at i, over 66,560 commit events. Because R covers only the active block, the window shrinks as a block fills and closes entirely at its end, so this measure attenuates for late commits by construction and understates their effect on later blocks. At $w{=}1$, $|R|$ also determines the within-block step, so we compare candidate and ordinary positions at matched $|R|$.

Outcome influence. From $s^{i \leftarrow a}$ and $s^{i \leftarrow b}$ we decode both branches to completion under shared seeds and record whether the final answer or its correctness changes. This probe carries no such window and is much smaller, at 976 branch pairs over 199 problems, since each event costs two full continuations. The two probes ask different questions and need not agree.

4 Experimental setup

Models and benchmarks. We evaluate LLaDA-8B-Instruct, Dream-7B-Instruct, and Nemotron-Labs-Diffusion-3B in bf16. We refer to them as LLaDA, Dream, and Nemotron. Our ports match official outputs at $T{=}0$ and reported greedy GSM8K accuracies of 77.5%, 84%, and 73%. We test GSM8K (500 problems; Cobbe et al., 2021), MATH500 (500; Hendrycks et al., 2021; Lightman et al., 2024), HumanEval (164; Chen et al., 2021) and MBPP (257 sanitized; Austin et al., 2021). Prompts and parsers follow Zhao et al. (2025). HumanEval and MBPP use execution-based verifiers.

Decoding configurations. Unless stated otherwise, we generate $L{=}256$ tokens in blocks of $B{=}32$. The base temperature is $T{=}0.6$. Policy comparisons within a regime vary only position selection. We compare one token per step with a parallel width of two. Four tokens per step is exploratory, because non-adjacent LTR departs further from strict LTR order there. Appendix analyses also use $L{=}512$ or $B{=}512$. For Nemotron, we test confidence-gated parallel decoding with $\gamma{=}0.9$.

Sampling and statistical comparisons. We draw $n{=}32$ samples per problem and condition, with $n{=}128$ for stated follow-ups. The corpus contains roughly 450k generations and represents about 850 GPU hours. We derive seeds by content hashing. We use paired problem-level bootstraps for every comparison. The $n{=}128$ runs reuse their first 32 samples from the exploratory runs, so they add 96 fresh samples rather than constituting independent replications.

Evidence hierarchy. Our primary comparison is the Pass@32 difference between LTR and confidence order at one token per step. Our controlled diagnostic is non-adjacent versus standard LTR at width two. Everything else, including timing, the choice of promoted positions, diversity, and targeted heating, is a follow-up chosen after seeing exploratory results, not an independent replication.

5 Results and discussion

5.1 When does the left-to-right advantage appear?

We first ask whether the reported LTR advantage is stable across models, tasks, commit widths and temperatures. It is not. The clearest positive effect appears with one token per step at low temperature, where LTR improves LLaDA/HumanEval Pass@32 by $+6.3 \pm 3.1$ pp (Table 1). Positive estimates also appear on Dream/HumanEval ($+7.9 \pm 3.8$ pp, $n{=}128$) and Nemotron/MATH500 ($+3.2 \pm 2.4$ pp). Its largest value is on MBPP at $+9.3 \pm 3.9$ pp, where it costs nothing in Pass@1 (-0.1 ± 1.8 pp) instead of trading accuracy for coverage as elsewhere. Six of ten pairs have positive intervals and five survive a Benjamini–Hochberg correction across the grid; the rest show no detectable difference (Table 12).

Increasing commit width weakens the contrast. At two tokens per step, the LTR estimate relative to confidence order is -2.8 ± 2.2 pp on LLaDA/HumanEval and -4.3 ± 4.0 pp on Dream/HumanEval (Fig. 1 and Tables 2 and 11). Candidate-first likewise shows no consistent benefit at the default

Table 1: Three routes to coverage on LLaDA/HumanEval with one token per step. Here T is sampling temperature and n is samples per problem. Confidence-temperature differences use $T{=}0.6$ as baseline, all others confidence order at the same T. Bold values have 95% intervals excluding zero.

condition	T	n	Pass@1	Pass@32	ΔPass@32
Official sampler and its temperature sweep					
confidence order	0.6	128	38.8	57.5	—
confidence order	1.4	128	31.6	63.2	**+5.7** ±2.6
Position-selection rules					
LTR order	0.6	128	35.2	63.7	**+6.3** ±3.1
LTR order	1.4	128	0.1	3.7	**-59.5** ±6.5
candidate-first	0.6	128	32.6	60.5	**+3.0** ±2.9
candidate-first	1.4	128	1.0	17.8	**-45.4** ±6.1

parallel width (Table 14). Matching test-time compute rather than sample count does not change this. At a fixed budget of 8192 forward passes, which buys 32 samples at one token per step and 128 at width four, confidence order holds 57.5 against 58.5 while LTR order falls from 63.7 to 41.5 (Table 3), so the width penalty belongs to LTR order rather than to parallelism.

Temperature provides another route to coverage. On LLaDA/HumanEval with one token per step, the LTR difference peaks at $T{=}0.6$ and shrinks in both directions, to $+4.9 \pm 4.0$ pp at $T{=}0.2$ below it and $+1.2 \pm 2.7$ pp at $T{=}1.0$ above it, so the advantage needs some sampling noise and disappears when there is too much. Without changing position order, confidence order peaks at $T{=}1.4$ with Pass@32 of 63.2, nearly matching LTR's 63.7 at $T{=}0.6$, then falls to 57.9 at $T{=}1.6$ and 39.0 at $T{=}1.8$. LTR itself becomes pathological at $T{=}1.4$, reaching only 3.7. Dream ranks positions on tempered logits while LLaDA does not, so temperature flattens its ranking as well as its sampling, yet it shows the same pattern with $+7.9 \pm 3.8$ pp at $T{=}0.6$, nothing at $T{=}1.0$, and collapse at $T{=}1.4$. At width two, neither LTR nor candidate-first has a positive interval at any tested temperature (Tables 1 and 13).

The two routes are not equivalent in quality, however. LTR at $T{=}0.6$ costs 3.5 pp of Pass@1, while confidence order at $T{=}1.4$ costs 7.2 pp and raises the syntax-error rate from 1.1% to 10.4% (Tables 4 and 10). Raising temperature buys coverage through stochastic exploration that degrades program validity, so LTR dominates it on the joint trade-off even where their coverage matches.

The low-temperature LTR contrast remains our strongest positive result. Within a post-hoc local-family sensitivity analysis, its paired-randomization p-value survives both Benjamini–Hochberg and Holm corrections; this does not adjust for selection of the cell or comparison family (Table 7). The partially overlapping $n{=}128$ follow-up also remains positive on the 96 fresh samples alone, at $+6.1 \pm 3.2$ pp (Table 8).

Take-away. The LTR advantage is strongest with one token per step at low temperature. Increasing width or temperature can reverse the contrast.

5.2 Why does commit width change the result?

The width sweep shows that the LTR effect is not stable, but not why. We next test the most direct width-two explanation. Standard LTR at width two is not merely the same order twice as fast: it always selects neighboring positions together, so the right-hand token cannot use the token sampled immediately to its left. Confidence order can also select neighbors but does not require it. On LLaDA/HumanEval, non-adjacent LTR (strided LTR in the artifact) recovers $+5.3 \pm 3.8$ pp over standard LTR. Its estimate relative to confidence order becomes $+2.5 \pm 3.4$ pp, although that interval includes zero. On Dream, the control recovers $+4.9 \pm 4.9$ pp and differs from confidence order by $+0.6 \pm 4.9$ pp. Both recoveries reach rough parity with confidence order rather than exceeding it. MBPP, with 257 problems, splits the result. Standard LTR does not lose at width two there, at $+0.4 \pm 4.3$ pp, yet the control still gains $+3.9 \pm 4.3$ pp over it and is the only arm to separate from confidence order, at $+4.3 \pm 4.1$ pp (Table 2). The width-two deficit is benchmark-dependent; the recovery's direction is not.

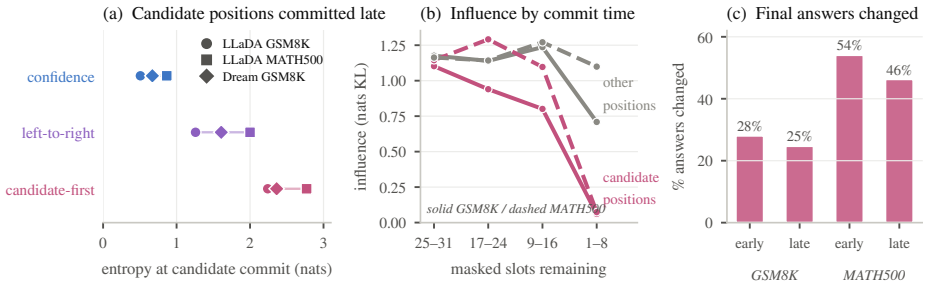

Figure 2: Candidate positions commit at lower entropy under confidence order, and late commits have lower influence in the sense of Eq. (3). These comparisons are observational and do not identify timing.

The control is compound, so we read it as support rather than isolation. It halves the rate at which a position commits without its left neighbor, 51.6% to 25.8%, but also inverts one pair in four and lets a bidirectional model see both sides of a position (Section B.1).

Take-away. Avoiding adjacent co-commits recovers LLaDA coverage, consistent with local context contributing to the width-two deficit. The control also changes some pairwise order, so it does not isolate that effect.

5.3 Can coverage increase without left-to-right priority?

Removing adjacent co-commits explains part of the parallel deficit but not the positive sequential result, so we ask whether LTR priority is needed at all. Our candidate-first rule moves detector-selected positions ahead of the confidence queue without a fixed positional schedule. Under confidence order these positions commit at lower entropy, and later candidates have lower measured influence (Fig. 2 and Table 18); because those probes compare different events, they motivate rather than identify a timing effect.

On LLaDA/HumanEval at n=128 candidate-first gains $+3.0$ pp over confidence order, about half the LTR estimate, and on Dream it gains $+7.3 \pm 3.3$ pp (Table 13); in Nemotron's gated parallel mode it reaches $+3.0 \pm 2.4$ pp (Table 15). This route is less robust than the primary contrast: the LLaDA comparison survives neither correction and gives $+2.8 \pm 3.0$ pp on the 96 fresh samples, an interval including zero (Tables 7 and 8), and it cuts Pass@1 from 38.8 to 32.6.

Candidate-first changes both which positions move early and when they move, so we next hold the count roughly fixed and vary only the timing. The simple delay of Section 3.3 cannot do this: it cuts promotions per sample from 5.09 to 0.06 across $d \in \{0, 8, 16, 24, 32\}$, a factor of 80, so its coverage trend cannot be read as timing (Tables 5 and 9 and Fig. 4).

The count-matched control instead uses S as the earliest promotion step (Section 3.3). Across $S \in \{0, 8, 16, 24\}$ at n=128, the mean promotion step moves from 3.6 to 23.9 while promotions per sample fall only from 2.70 to 2.11, measured on 320 samples per arm (Table 5). Pass@32 relative to confidence order reads $+4.2$, $+4.9$, $+2.7$ and $+1.6$ pp, Pass@1 rises from 34.9 at S=0 to 37.5 at S=24, and distinct-4gram falls from 0.317 to 0.215 (Table 6). The two earliest arms sit within noise of each other, so the decline runs from S=8 to S=24, a difference of $+3.2$ pp with interval $[+0.8, +5.8]$. Moving promotions earlier therefore raises Pass@32 and lowers Pass@1 at a roughly fixed count.

Two residuals bound that reading. Promotions still fall by 22% across the sweep, and the entropy at promotion falls from 3.48 to 1.24 nats, so late promotions land on positions that have largely resolved. Across selectors, counts range from 1.97 to 5.41 while Pass@32 changes by at most 0.8 pp (Tables 5 and 16), though selector and count change together there, so matching counts by construction remains the cleaner design.

Take-away. Positive coverage estimates can arise without fixed LTR priority, although candidate-first is weaker than the primary contrast. With counts approximately matched, earlier promotions raise coverage but lower Pass@1.

5.4 Does it matter which positions move earlier?

Candidate-first raises coverage without LTR order, but is the detector finding especially useful positions? If so, targeted selectors should beat random promotion at a similar number of moved positions. We compare the connective dictionary with learned influence, entropy, and uniformly random selection. The learned predictor reaches ROC AUC 0.730 on held-out influence events against 0.712 for entropy alone, but transfers weakly to outcomes at 0.570 (Section B.4).

Their Pass@32 estimates nonetheless lie between $+2.6$ and $+3.4\,\mathrm{pp}$, with random promotion at $+2.6 \pm 2.0\,\mathrm{pp}$ and no pairwise advantage for the dictionary or the learned rule over entropy (Table 16 and Section B.4). We find no evidence that these particular positions beat random ones. The test is not clean at a fixed count, though: matched flag rates still produce 5.09 realized promotions per sample for the dictionary, 4.34 learned, 1.97 entropy and 5.41 random (Table 5), and entropy reaches the same coverage as random while moving under half as many positions. Isolating which positions matter needs exact count matching.

The negative controls rule out the broader claim that any early commitment helps. Unrestricted entropy-first reduces HumanEval Pass@32 by $40.2\,\mathrm{pp}$, and over-promoting learned-rule positions costs as much as $16.6\,\mathrm{pp}$, so indiscriminate or excessive early commitment is harmful (Tables 14, 16 and 17 and Section B.4).

Take-away. Targeted selectors do not clearly outperform random promotion, but unequal promotion counts prevent a clean test of which positions matter.

5.5 Is the result explained by diversity?

A straightforward explanation is that early commitment and temperature simply produce more varied samples, giving Pass@32 more chances to succeed, so we first compare conditions with similar measured diversity. Distinct-4gram is 0.366 for confidence order at $T{=}1.4$ and 0.361 for candidate-first at $T{=}0.6$, and five other measures differ by at most 0.012 (Section B.3). Yet these near-identical diversities give Pass@32 of 63.2 and 60.5, against 63.7 for LTR, and their Pass@1 costs differ, so matched diversity does not imply matched coverage or sample quality (Table 10).

As a post-hoc descriptive diagnostic, we fit Pass@32 as a quadratic function of distinct-4gram fraction across 20 LLaDA/HumanEval conditions, choosing the form and the 0.70 cutoff after examining them. Across all conditions the fit is strong (in-sample $R^2{=}0.954$; leave-one-out $R^2{=}0.936$), but broad decoding changes and severe failures drive this result (Fig. 5). Among competitive conditions (distinct-4gram below 0.70 and Pass@32 from 52.4 to 63.7), leave-one-out R^2 falls to -0.25; Dream similarly reaches -0.26.

The fit also misses the held-out targeted-heating conditions by 10–12 pp at $T_{\mathrm{hot}}{=}2.0$, one of which nearly matches LTR's diversity but reaches Pass@32 of 48.8 rather than 63.7 (Fig. 5 and Table 17). Distinct-4gram also scores corrupted text highly, and structural diversity loses its association with coverage once we condition on parseability (Section B.3 and Table 4).

Take-away. Diversity tracks broad changes and obvious failures but does not reliably distinguish competitive decoders, so it is not a sufficient explanation.

5.6 What remains unidentified?

Generality limits our conclusions. Positive one-token-per-step LTR intervals appear in six of ten pairs; timing, temperature and diversity analyses focus on LLaDA/HumanEval, and the outcome probe lacks problem-clustered uncertainty. All four benchmarks also have natural left-to-right dependency structure, whereas arbitrary order is known to beat LTR order on constraint-satisfaction and planning

tasks (Kim et al., 2025), so we bound these conclusions to the LTR-friendly regime where the advantage was claimed.

Post-selection also limits the evidence. We chose several follow-ups after exploration. LTR survives a post-hoc local-family correction and remains positive on fresh samples, while candidate-first does neither; this does not adjust for selecting the cell or family, and each $n{=}128$ run reuses 32 exploratory samples (Tables 7 and 8).

The interventions also do not isolate every factor perfectly, since the timing control realizes 2.11–2.70 promotions rather than fixing the count and selector counts differ (Tables 5 and 6). Finally, most positive estimates use one token per step, giving up dLLM parallelism, with gated Nemotron/MATH500 the exception (Table 15). The evidence is diagnostic, and the mechanism remains only partly identified.

6 Related work

How dLLMs choose positions. LLaDA, Dream, Nemotron, and related systems use confidence-based commitment (Nie et al., 2025; Ye et al., 2025b; Fu et al., 2026; Ghazvininejad et al., 2019; Gong et al., 2025; Inception Labs et al., 2025; Google DeepMind, 2025; Song et al., 2025; Ye et al., 2024; 2025a; Zhao et al., 2025; Zhu et al., 2026; Tang et al., 2026). Other work learns or fixes order (Uria et al., 2014; Vinyals et al., 2016; Yang et al., 2019; Hoogeboom et al., 2022; Shih et al., 2022; Gu et al., 2018; Stern et al., 2019; Gu et al., 2019; Welleck et al., 2019; Li et al., 2021; Pannatier et al., 2024), or uses entropy, planning, and learned policies (Ben-Hamu et al., 2025; Peng et al., 2025; Jazbec et al., 2025; Asano et al., 2026; Lee et al., 2025; Chang et al., 2026; Liu et al., 2026). We compare decoder-policy coverage while fixing the model, loop, and width.

Parallelism versus fidelity. Caching and adaptive width pursue lower latency (Wu et al., 2026; Ma et al., 2025; Israel et al., 2025), while remasking spends extra function evaluations to revise committed tokens (Wang et al., 2025a); theory predicts that parallel commitment trades fidelity for speed (Feng et al., 2025). Closest to our width-two control, dilated scheduling fixes a spatial stride so adjacent tokens never commit together (Luxembourg et al., 2026); we apply that idea to a left-to-right schedule as a diagnostic rather than as a speed method. Adaptive orders can also solve planning tasks that fixed orders cannot (Kim et al., 2025).

Coverage, branching, and test-time sampling. Closest to us, Ni et al. (2026) report lower Pass@k under arbitrary-order than forced LTR decoding. In autoregressive reasoning, resampling one token can redirect a chain of thought (Wei et al., 2022; Bigelow et al., 2025); minority tokens and alternative paths also matter (Lin et al., 2025; Wang et al., 2025b; 2023; Muennighoff et al., 2025). These autoregressive methods choose *what* to sample at a fixed position; a dLLM also chooses *where* and *when*. We separate these choices before attributing coverage to LTR order.

7 Conclusion

The main lesson is that generation order is not one experimental variable. A decoder chooses where, when, and how many positions to commit, what context they can use, and how randomly to sample, and each choice can alter Pass@k. The LTR advantage is strongest at one token per step and low temperature, avoiding adjacent co-commits recovers much of its width-two deficit, and coverage can rise without fixed LTR priority. Widening coverage usually costs single-sample accuracy, though on MBPP the sequential gain is free. Claims that LTR order is intrinsically better, especially when motivating training-time constraints, should first separate these factors.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, specifically writing and refactoring our analysis, evaluation, and figure-generation scripts. We have not used generative AI tools for developing theoretical models or conceptual frameworks, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, or interpreting results, and generating synthetic data sets, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translation, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for creating and editing software code, creating scientific figures, and editing the paper to improve readability. We have reviewed all AI-assisted work. The authors reviewed all AI-assisted code and text, and the released scripts compute every quantitative claim directly from the generation corpus. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work evaluates decoding procedures for publicly released language models on public benchmarks (HumanEval, MBPP, GSM8K, MATH500) and involves no human subjects, no personal data and no new data collection. We execute generated code only inside the standard sandboxed harnesses. Our findings concern sampling behaviour rather than model capabilities, and we see no direct dual-use concern beyond those already present in the released models we study. We report a negative result in Section 5.4 and an error in our promotion-rate calibration procedure. We report both in full to make the limits of the evidence explicit.

REPRODUCIBILITY STATEMENT

The family in Eq. (1), stated in Section 3 and enumerated in Section A.3, expresses all position-selection policies. Section A gives the equivalence tests against the official samplers, the prompts and parsers, and the seeding scheme. Decoder comparisons use paired problem-level bootstraps as defined in Section 2. Committed scripts read the raw generation corpus and generate each table and figure. The supplementary material includes the sampler, the analysis and table generators, and the per-condition completeness audit. The generation corpus itself is roughly 9.6 GB across 1,343 files, so it exceeds the supplementary size limit; we will release it in full alongside the camera-ready.

REFERENCES

Hikaru Asano, Tadashi Kozuno, Kuniaki Saito, and Yukino Baba. Where-to-unmask: Ground-truth-guided unmasking order learning for masked diffusion language models. *arXiv preprint arXiv:2602.09501*, 2026. 9

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint*, 2021. arXiv:2108.07732. 5

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.24857. 2, 9

Eric Bigelow, Ari Holtzman, Hidenori Tanaka, and Tomer Ullman. Forking paths in neural text generation. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2412.07961. 9

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024. 2

Shenxu Chang, Junchi Yu, Weixing Wang, Yongqiang Chen, Jialin Yu, Philip Torr, and Jindong Gu. Tracedet: Hallucination detection from the decoding trace of diffusion large language models. In *International Conference on Learning Representations*, 2026. 9

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 2, 3, 5

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 5

Guhao Feng, Yihan Geng, Jian Guan, Wei Wu, Liwei Wang, and Di He. Theoretical benefit and limitation of diffusion language model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2502.09622. 9

Yonggan Fu, Lexington Whalen, Abhinav Garg, Chengyue Wu, Maksim Khadkevich, Nicolai Oswald, Enze Xie, Daniel Egert, Sharath Turuvekere Sreenivas, Shizhe Diao, Chenhan Yu, Ye Yu, Weijia Chen, Sajad Norouzi, Jingyu Liu, Shiyi Lan, Ligeng Zhu, Jin Wang, Jindong Jiang, Morteza Mardani, Mehran Maghoumi, Song Han, Ante Jukic, Nima Tajbakhsh, Jan Kautz, and Pavlo Molchanov. Nemotron-labs-diffusion: A tri-mode language model unifying autoregressive, diffusion, and self-speculation decoding. Technical report, NVIDIA, 2026. Models at `https://github.com/NVlabs/Nemotron-Labs-Diffusion`. 9

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019. arXiv:1904.09324. 9

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, et al. Scaling diffusion language models via adaptation from autoregressive models. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.17891. 9

Google DeepMind. Gemini diffusion. `https://deepmind.google/models/gemini-diffusion/`, 2025. Accessed 2026-08-10. 9

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *International Conference on Learning Representations (ICLR)*, 2018. arXiv:1711.02281. 9

Jiatao Gu, Changhan Wang, and Jake Zhao. Levenshtein transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1905.11006. 9

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021. arXiv:2103.03874. 5

Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2110.02037. 9

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025. 9

Daniel Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion LLMs via adaptive parallel decoding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2506.00413. 9

Metod Jazbec, Theo X. Olausson, Louis Béthune, Pierre Ablin, Michael Kirchhof, João Monteiro, Victor Turrisi, Jason Ramapuram, and Marco Cuturi. Learning unmasking policies for diffusion language models. *arXiv preprint arXiv:2512.09106*, 2025. 9

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv preprint arXiv:2502.06768*, 2025. 2, 9

Sanghyun Lee, Seungryong Kim, Jongho Park, and Dongmin Park. Lookahead unmasking elicits accurate decoding in diffusion language models. *arXiv preprint arXiv:2511.05563*, 2025. 9

Xuanlin Li, Brandon Trabucco, Dong Huk Park, Michael Luo, Sheng Shen, Trevor Darrell, and Yang Gao. Discovering non-monotonic autoregressive orderings with variational inference. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2110.15797. 9

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, et al. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2305.20050. 5

Zicheng Lin, Tian Liang, Jiahao Xu, Qiuzhi Lin, Xing Wang, Ruilin Luo, et al. Critical tokens matter: Token-level contrastive estimation enhances LLM's reasoning capability. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2411.19943. 9

Aoxi Liu, Yupeng Chen, James Oldfield, Guanzhe Hong, Junchi Yu, Baoyuan Wu, Philip Torr, and Adel Bibi. D^2-Monitor: Dynamic safety monitoring for diffusion llms via hesitation-aware routing. *arXiv preprint arXiv:2605.25893*, 2026. 9

Omer Luxembourg, Haim Permuter, and Eliya Nachmani. Plan for speed: Dilated scheduling for masked diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2506.19037. 9

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dKV-Cache: The cache for diffusion language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.15781. 9

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, et al. s1: Simple test-time scaling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025. arXiv:2501.19393. 9

Zanlin Ni, Shenzhi Wang, Yang Yue, Tianyu Yu, Weilin Zhao, Yeguo Hua, Tianyi Chen, Jun Song, Cheng Yu, Bo Zheng, et al. The flexibility trap: Rethinking the value of arbitrary order in diffusion language models. *arXiv preprint arXiv:2601.15165*, 2026. 2, 9

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025. 9

Arnaud Pannatier, Evann Courdier, and François Fleuret. σ-GPTs: A new approach to autoregressive models. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2024. arXiv:2404.09562. 9

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Avishek Joey Bose, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*, 2025. 9

Andy Shih, Dorsa Sadigh, and Stefano Ermon. Training and inference on any-order autoregressive models the right way. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2205.13554. 9

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2408.03314. 2

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025. 9

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning (ICML)*, 2019. arXiv:1902.03249. 9

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy optimization for reasoning in diffusion language models. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2507.08838. 9

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning (ICML)*, 2014. arXiv:1310.1757. 9

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. In *International Conference on Learning Representations (ICLR)*, 2016. arXiv:1511.06391. 9

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a. arXiv:2503.00307. 9

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b. arXiv:2506.01939. 9

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2203.11171. 9

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2201.11903. 9

Sean Welleck, Kianté Brantley, Hal Daumé III, and Kyunghyun Cho. Non-monotonic sequential text generation. In *International Conference on Machine Learning (ICML)*, 2019. arXiv:1902.02192. 9

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2505.22618. 2, 9

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1906.08237. 9

Jiacheng Ye, Shansan Gong, Liheng Chen, Lin Zheng, Jiahui Gao, Han Shi, et al. Diffusion of thought: Chain-of-thought reasoning in diffusion language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2402.07754. 9

Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Beyond autoregression: Discrete diffusion for complex reasoning and planning. In *International Conference on Learning Representations (ICLR)*, 2025a. arXiv:2410.14157. 2, 9

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025b. 9

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *arXiv preprint arXiv:2504.12216*, 2025. 5, 9, 14

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, et al. LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. arXiv:2505.19223. 9

Appendix guide. Section A records implementation and reproducibility details; Section B tests how far the main interpretations hold; Section C collects extended quantitative results; and Section D contains three visual summaries.

A Reproducibility and Implementation

A.1 Does the fixed loop reproduce the official samplers?

At $T=0$, our loop is output-equivalent to the official implementations in the evaluated settings; the accompanying tests check this directly.

A.2 How are prompts, seeds, and evaluations fixed?

We take prompts and parsers verbatim from Zhao et al. (2025). We derive each seed by hashing the problem identifier and the first sample index in its batch. The released corpus contains approximately 450k JSONL records together with a per-condition completeness audit. Paired problem-level bootstrap code and the official-sampler equivalence tests accompany the submission. The artifact README records the two software environments, package versions, scoring command, and table/figure regeneration commands; official implementations and verifiers also ship with the artifact.

A.3 What changes under each control policy?

Equation (1) defines the common selection family. Each control changes the listed component while retaining the same weights, prompt, block loop, masking rule, temperature unless stated, and evaluation.

Exact candidate-first rule. For each tokenizer, we compile $\mathcal{D} \subset \mathcal{V}$, the token ids that begin a connective word at a word boundary, from a fixed list of 26 stems (*Therefore, Thus, So, Since, However, Then, Because*, ...). Write $\phi(s) = \min \mathcal{M}_B(s)$ for the decoded frontier. A masked position is a candidate when its top prediction begins one of these stems and it lies within a window W of the frontier:

$$\mathcal{F}(s) = \big\{ i \in \mathcal{M}_B(s) \mid \arg\max_v p_i(v \mid s) \in \mathcal{D},\ i < \phi(s) + W \big\}. \tag{4}$$

We use $W{=}32$ throughout, so the window condition is vacuous when $B{=}32$. With lexicographic comparison of pairs, candidate-first scores positions by

$$\sigma_i^{\mathrm{ff}}(s) = \Big(\mathbf{1}[i \in \mathcal{F}(s)],\ \mathbf{1}[i \in \mathcal{F}(s)] H_i(s) + \mathbf{1}[i \notin \mathcal{F}(s)] c_i(s) \Big). \tag{5}$$

When $\mathcal{F}(s) = \emptyset$, this rule produces the same commit set as confidence order and never imposes a fixed positional schedule.

control	What changes	What stays fixed	Question tested
non-adjacent LTR	within-step spacing	width, model, and loop	whether adjacent co-commits contribute to the deficit
entropy-first	selection priority	width, model, and loop	whether uncertainty alone helps
random promotion	chosen positions	calibration procedure, model, and loop	whether targeting particular positions matters
learned selector	candidate score	calibration procedure, model, and loop	whether predicted influence improves targeting
raise temperature at selected positions	local sampling temperature	confidence order	whether local stochasticity reproduces the effect

`fork` is an operational name for detector-selected candidate positions, not a validated semantic definition of reasoning branches. We therefore use candidate terminology below.

B Robustness and Mechanism Diagnostics

B.1 How much does non-adjacent LTR alter positional order?

Non-adjacent LTR walks the block left to right and takes a masked position only once it lies at least `stride` positions past the last one taken, with `stride`$=2$ throughout. When the separation constraint runs out of eligible positions near the end of a block, the remainder falls back to the leftmost unused masked positions so the commit count stays exactly w. At $w{=}2$ on a fresh block this commits $\{0, 2\}$, then $\{1, 3\}$, then $\{4, 6\}$, and so on, and at $w{=}1$ it is identical to LTR order by construction.

Because the LTR schedules depend only on the mask, we replay them exactly, and we replay confidence order through the model (128 LLaDA/HumanEval samples). The audit records distance from the decoded frontier and two neighbor statistics.

policy	mean distance from frontier	committed before left neighbor	left neighbor unavailable
width 2			
confidence	3.23	19.0%	48.8%
LTR order	0.50	0.0%	51.6%
non-adjacent LTR	1.00	25.8%	25.8%
width 4			
confidence	4.72	17.2%	66.8%
LTR order	1.50	0.0%	77.4%
non-adjacent LTR	3.00	38.7%	38.7%

The two neighbor columns answer different questions, and only the second one speaks to adjacency. The first counts commits made strictly before the left neighbor, so it reads 0.0% for LTR order at every width. That is a property of the column rather than of LTR order, whose deprivation is entirely simultaneous. The second column counts a neighbor committed in the same step as unavailable as well, which is the quantity the width-two argument is about.

Read that way, the control is compound in two directions. It halves the rate at which a position is committed without its left neighbor, from 51.6% to 25.8%, but it also inverts about one pair in four and moves commits farther from the frontier. It additionally supplies context that strict left-to-right generation never has. When non-adjacent LTR commits position i after $i-1$ and $i+1$ are both fixed, the model conditions on both sides, and these models attend bidirectionally. Part of the recovery may therefore come from this infilling context rather than from restoring the left predecessor alone, which the present design does not separate.

Confidence order is the reference that keeps the rate reading honest. It commits without the left neighbor at 48.8% at width two, essentially LTR order's 51.6%, and still scores higher. The rate of deprivation therefore does not by itself predict coverage; which positions get deprived evidently matters, and confidence order deprives only positions it has already scored as near-certain.

Table 2: Avoiding adjacent co-commits recovers part of the width-two LTR deficit. Differences are in Pass@32 percentage points; bold marks paired-bootstrap intervals excluding zero. The LLaDA non-adjacent LTR arm has 32 samples per problem and its LTR and confidence references have 128; all Dream and MBPP arms have 32. MBPP has 257 problems against HumanEval's 164, and there standard LTR does not lose at width two, so the recovery has less to recover. Changing selected pairs also changes some pairwise order, so this is not a pure adjacency intervention. The Dream recovery interval includes zero; its lower bound is exactly 0.0 rather than a rounded negative value, because the paired bootstrap over 164 problems places its 2.5th percentile on zero itself.

model	width	non-adj. LTR − LTR	non-adj. LTR − confidence	LTR − confidence
LLaDA	w=2	**+5.3** $[+1.7, +9.3]$	+2.5 $[-0.8, +6.1]$	**-2.8** $[-5.1, -0.7]$
Dream	w=2	+4.9 $[+0.0, +9.8]$	+0.6 $[-4.3, +5.5]$	**-4.3** $[-8.5, -0.6]$
LLaDA MBPP	w=2	+3.9 $[-0.4, +8.2]$	**+4.3** $[+0.4, +8.6]$	+0.4 $[-3.9, +4.7]$

Table 3: Equal test-time compute across commit widths on LLaDA/HumanEval. A budget of 8192 forward passes buys 32 samples at width 1, 64 samples at width 2, 128 samples at width 4. Confidence order is nearly flat across the three, so parallel decoding costs it little once the saved compute is spent on more samples, whereas LTR order degrades sharply. Bold values have 95% intervals excluding zero.

width	samples	confidence Pass@k	LTR Pass@k	LTR − conf.
$w = 1$	32	57.5	63.7	**+6.3 ±3.1**
$w = 2$	64	60.2	57.6	-2.6 ±3.2
$w = 4$	128	58.5	41.5	**-17.1 ±5.8**

The width-two control supports the local-context interpretation without isolating it, since it changes pairwise order and adds bidirectional context alongside the adjacency it removes.

B.2 How sensitive is the candidate detector?

We fixed the 26-stem list before all experiments and did not tune it against Pass@k. In full it is *Therefore, Thus, So, Since, When, Given, However, Let, First, Then, Next, Finally, Now, Similarly, Calculate, Solving, Notice, Specifically, Follows, Because, But, Or, Consider, Also, Express, Write*. A position is a candidate when its untempered top-1 prediction begins one of these stems, case-insensitively and including a leading-space variant. Five stems (*Or, First, Then, Given, Calculate*) account for 74% of HumanEval candidate commits and primarily mark enumeration or case splitting. The five-stem rule makes 77% as many candidate commits and differs from the full rule by -0.2 ± 3.5 pp in Pass@32. Its estimate relative to confidence is $+2.9 \pm 3.6$ pp, compared with $+3.0 \pm 2.9$ pp for the full rule. A small enumerative subset therefore drives most detector behavior.

Restricting the detector to five dominant enumerative stems does not materially change its estimate; this supports its use as an operational perturbation, not as a semantic fork detector.

B.3 How robust is the diversity conclusion?

B.3.1 Metric sensitivity

The similarity between confidence order at T=1.4 and candidate-first at T=0.6 is stable across lexical diversity metrics. Values are problem means over the same 32 samples.

condition	dist-2g	dist-3g	dist-4g	dist-5g	Jaccard	unique
confidence T=0.6	0.103	0.142	0.173	0.199	0.582	0.836
confidence T=1.4	0.202	0.293	0.366	0.425	0.798	0.999
LTR order T=0.6	0.190	0.278	0.349	0.409	0.811	0.999
candidate-first T=0.6	0.190	0.285	0.361	0.424	0.808	0.994

Confidence at T=1.4 and candidate-first differ by at most 0.012 on these measures.

B.3.2 Raising temperature at selected positions

The quadratic form and 0.70 cutoff are post-hoc descriptive choices. The fit has full-range $R^2 = 0.954$ and leave-one-out $R^2 = 0.936$, but below distinct-4gram 0.70 its leave-one-out R^2 is -0.25 (Dream -0.26). It also mis-predicts conditions that raise temperature only at selected positions and that we excluded from fitting (Table 17). At $T_{\text{hot}} = 2.0$, point estimates fall 10–12 points below the curve. Raising temperature at influence-selected positions yields diversity 0.353, close to LTR order's 0.349, but Pass@32 is 48.8 rather than 63.7.

B.3.3 Degeneration and parseability

Distinct-4gram can score corrupted output highly. LTR order at T=1.4 has distinct-4gram 0.949 but distinct-AST 0.010. Within the competitive range, conditioning on parseability removes the distinct-AST association with coverage (Spearman point estimate $+0.08$). Duplicates reduce effective sample count under confidence order, but do not order the competitive arms.

Table 4: Completion uniqueness, structural diversity, and primary failure modes on HumanEval (LLaDA). Unique is the mean per-problem fraction of distinct completions. Distinct-AST measures distinct normalized parse trees, so it scores degenerate output lower than distinct-4gram. Duplicates reduce the effective sample count under confidence order but do not order the competitive conditions.

arm	unique	distinct-AST	correct	no code	syntax
confidence T=0.6	73.9%	23.7%	38.8%	2.0%	1.1%
LTR T=0.6	99.8%	47.8%	35.2%	3.8%	1.9%
candidate-first T=0.6	98.6%	43.6%	32.6%	3.0%	2.7%
confidence T=1.4	99.9%	45.1%	31.6%	2.5%	10.4%
LTR T=1.4	100.0%	1.0%	0.1%	98.5%	1.0%

Diversity describes broad changes in decoding, but neither lexical nor structural diversity reliably discriminates coverage among competitive decoders in these post-hoc diagnostics.

Table 6: Early commits per sample stay flat while their timing moves. Delaying them lowers Pass@32 and raises Pass@1, both toward confidence order. LLaDA/HumanEval at n=128, with counts measured on 320 samples per arm. Bold values have 95% intervals excluding zero.

promotion floor	early commits per sample	mean step	distinct 4gram	Pass@1	Pass@32	ΔPass@32 vs confidence
confidence order	0.00	—	0.173	38.8	57.5	—
$S = 0$	2.70	3.6	0.317	34.9	61.7	**+4.2** ±2.6
$S = 8$	2.49	9.0	0.303	36.1	62.4	**+4.9** ±2.6
$S = 16$	2.12	16.3	0.254	37.3	60.2	**+2.7** ±2.4
$S = 24$	2.11	23.9	0.215	37.5	59.1	+1.6 ±1.8

B.4 Why does the learned selector not outperform simpler controls?

We train the influence model on 35,840 GSM8K events and evaluate it on 30,720 held-out MATH500 events. Within-block influence in Eq. (3) is continuous, so we binarize it for classification. A commit event counts as high-influence when its mean KL exceeds the 75th percentile of events sharing its value of $|R|$, stratified because $|R|$ fixes the within-block step at w=1 and would otherwise dominate the label. This gives a positive rate of 23.1%. The six features are the log and raw vocabulary entropy at the position, the top-1 probability and its logit, the relative decoding step, and the relative number of remaining masked positions. Gradient boosting reaches ROC AUC 0.730, only modestly above entropy alone at 0.712 and logistic regression on the same features at 0.717. Transfer to final-outcome influence is weak (AUC 0.570). Table 18 reports descriptive answer-change rates; because branch pairs are nested within problems, we do not use them for an inferential claim.

Calibration on confidence-order mathematics states does not transfer directly to learned-order code states. In-loop calibration makes the per-masked-position flag rates close (0.108% versus 0.112%), but actual early commits still differ, with 5.09 per sample for the dictionary, 4.34 learned, 1.97 entropy, and 5.41 random. Similar flag rates therefore do not identify position effects at fixed promotion count. Direct paired differences from random are $+0.4 \pm 3.0$ pp for the dictionary, $+0.7 \pm 2.6$ pp for learned selection, and $+0.1 \pm 2.4$ pp for entropy; every interval includes zero.

Table 5: Actual promotions and their mean within-block step on LLaDA/HumanEval. A promotion commits a candidate that confidence order would not choose at that step. Delay d is the required number of consecutive eligible steps; floor S is the earliest step when the roughly count-matched rule may promote. This rule realizes 2.47–3.22 promotions without a trend in S, whereas calibrated selectors have different counts despite similar flag rates. Mean step averages promotion events; Table 9 averages all logged candidate commits.

arm	early commits/sample	mean step
Baseline		
confidence order	0.00	—
Candidate committers		
candidate-first, dictionary	5.09	6.8
learned, calibrated	4.34	—
entropy, calibrated	1.97	4.0
random, calibrated	5.41	9.4
Early commitment held back d steps		
$d = 0$ (candidate-first)	5.09	6.8
$d = 8$	3.28	11.9
$d = 16$	2.09	17.9
$d = 24$	1.28	24.4
$d = 32$	0.06	30.0
Count-matched dial, promotion floor S		
$S = 0$	2.70	3.6
$S = 8$	2.49	9.0
$S = 16$	2.12	16.3
$S = 24$	2.11	23.9

Influence prediction transfers poorly to outcomes, and promotion-count mismatch prevents a clean claim that targeting particular positions beats random early commitment.

B.5 How robust are the headline contrasts?

As a post-hoc local-family sensitivity analysis, we apply paired sign-flip randomization tests to 16 selected LLaDA/HumanEval contrasts spanning priority, temperature, delay, and selector analyses. Eleven percentile intervals exclude zero, nine comparisons survive Benjamini–Hochberg at $q = 0.05$, and six survive Holm at $\alpha = 0.05$. The LTR T=0.6 contrast survives both; candidate-first at T=0.6 does not. This analysis does not adjust for selecting the LLaDA/HumanEval cell or the comparison family.

Table 7: Post-hoc local-family sensitivity analysis for 16 selected LLaDA/HumanEval ΔPass@32 comparisons spanning priority, temperature, delay, and selector analyses. Delay rows use the 128-sample follow-up runs; Table 9 reports the original samples 0–31. Confidence-temperature rows use T=0.6 as baseline; other rows use confidence at the same temperature. Intervals are paired problem-bootstrap 95% percentile intervals, and p-values use two-sided paired sign flips. The analysis does not adjust for selecting this cell or comparison family. A dot marks a comparison that does not survive. Uncorrected, 11 intervals exclude zero; Benjamini–Hochberg at q=0.05 retains 9 and Holm at α=0.05 retains 6.

arm	T	ΔPass@32 [95% CI]	p	uncorrected	BH	Holm
LTR order	1.4	-59.5 $[-65.9, -52.9]$	< 0.0001	✓	✓	✓
candidate-first	1.4	-45.4 $[-51.6, -39.4]$	< 0.0001	✓	✓	✓
learned, q=0.90 (over-promoted)	0.6	-16.6 $[-22.5, -11.0]$	< 0.0001	✓	✓	✓
confidence order	1.4	+5.7 $[+3.2, +8.4]$	< 0.0001	✓	✓	✓
LTR order	0.6	+6.3 $[+3.3, +9.5]$	< 0.0001	✓	✓	✓
learned, calibrated	0.6	+3.4 $[+1.2, +5.8]$	0.0037	✓	✓	✓
entropy, calibrated	0.6	+2.7 $[+0.8, +4.8]$	0.0062	✓	✓	.
candidate-first	1.0	-5.2 $[-9.2, -1.5]$	0.0081	✓	✓	.
random, calibrated	0.6	+2.6 $[+0.6, +4.7]$	0.0117	✓	✓	.
confidence order	1.0	+1.9 $[+0.2, +3.7]$	0.0356	✓	.	.
candidate-first	0.6	+3.0 $[+0.2, +6.0]$	0.0428	✓	.	.
$d = 24$	0.6	-1.4 $[-3.6, +0.8]$	0.2200	.	.	.
$d = 32$	0.6	-1.4 $[-4.4, +1.8]$	0.3897	.	.	.
$d = 8$	0.6	+1.7 $[-2.0, +5.6]$	0.3935	.	.	.
LTR order	1.0	+1.2 $[-1.5, +3.9]$	0.3998	.	.	.
$d = 16$	0.6	-0.2 $[-2.4, +2.3]$	0.8963	.	.	.

Table 8: Headline LLaDA/HumanEval Pass@32 contrasts on all 128 samples and on the 96 samples with indices 32–127 that were not present in the exploratory runs. Intervals are paired problem-bootstrap percentile intervals.

contrast	sample pool	n	ΔPass@32 [95% CI]
LTR order	all	128	+6.3 $[+3.3, +9.5]$
LTR order	fresh only	96	+6.1 $[+3.1, +9.4]$
candidate-first	all	128	+3.0 $[+0.2, +6.0]$
candidate-first	fresh only	96	+2.8 $[-0.1, +5.8]$

The n=128 runs reuse samples 0–31 from exploratory runs and add 96 fresh samples. Table 8 reports the all-sample and fresh-only intervals. We selected follow-up conditions after observing the exploratory runs, so we did not pre-specify which conditions would receive larger samples.

The sequential contrast is robust within this local sensitivity analysis and on fresh samples; neither analysis removes the post-selection limitation. The candidate-first estimate remains less secure.

C Extended Quantitative Results

For compactness, tables use LTR for forced left-to-right priority.

C.1 When do width and temperature change the result?

These tables collect the delay, Pass@1 trade-off, width, temperature, and one-token-per-step model–benchmark comparisons. The width-two non-adjacent LTR audit remains in Section B; we treat wider results only as exploratory schedule controls.

Table 9: Why simple delay does not isolate timing (LLaDA-8B, HumanEval, one token per step). Here d is the required number of consecutive eligible steps before promotion. Increasing it changes timing, promotion count, diversity, and coverage together because candidates can disappear before eligibility. At d=0 the rule is candidate-first; confidence order is the no-promotion limit. Lateness is mean within-block commitment step. Rows use samples 0–31. The final column is the paired-bootstrap Pass@32 difference from confidence order; bold 95% intervals exclude zero.

delay d	n	candidate lateness	distinct-4g	Pass@1	ΔPass@32
confidence	32	16.1	0.173	39.0	—
0 (candidate-first)	32	7.4	0.361	32.7	**+5.5** [+0.6, +10.4]
8	32	12.1	0.290	35.6	+3.0 [−1.2, +7.9]
16	32	17.9	0.216	37.0	+1.2 [−1.8, +4.9]
24	32	24.6	0.185	37.3	+0.0 [−1.8, +1.8]
32	32	30.8	0.178	38.6	+0.0 [−2.4, +2.4]

Table 10: Pass@1 trade-offs for coverage-improving conditions on LLaDA/HumanEval at one token per step.

	Pass@		Δ vs conf.	
decoder	1	32	@1	@32
confidence, T=0.6	38.8	57.5	—	—
confidence, T=1.4	31.6	63.2	-7.2	+5.7
LTR order, T=0.6	35.2	63.7	-3.5	+6.3
candidate-first, T=0.6	32.6	60.5	-6.2	+3.0

Table 11: The LTR coverage difference shrinks and reverses as commit width w increases on LLaDA/HumanEval; n is samples per problem. Bold values have 95% intervals excluding zero.

w	steps	n	conf. Pass@1	LTR − conf. (Pass@32)
1	256	128	38.8	**+6.3** ±3.1
2	128	128	30.6	**-2.8** ±2.2
4	64	128	17.2	**-21.6** ±4.9

Table 12: LTR–confidence Pass@32 differences for all evaluated one-token-per-step model–benchmark pairs. Bold values have 95% intervals excluding zero; LLaDA and Dream HumanEval use n=128. 6 of the 10 cells resolve, and five survive a Benjamini–Hochberg correction across the grid; Dream/MATH500 is the one that does not, at p=0.049. We specified the confirmatory comparison per model before the powered runs.

Model	Benchmark	problems	n	LTR − conf. (Pass@32)
LLaDA-8B	GSM8K	500	32	**+3.0** ±1.6
	HumanEval	164	128	**+6.3** ±3.1
	MATH500	500	32	+2.4 ±3.2
	MBPP	257	32	**+9.3** ±3.9
Dream-7B	GSM8K	500	32	+0.8 ±1.2
	HumanEval	164	128	**+7.9** ±3.8
	MATH500	500	32	**+2.8** ±2.6
Nemotron-3B	GSM8K	500	32	+0.2 ±0.8
	HumanEval	164	32	-0.6 ±2.1
	MATH500	500	32	**+3.2** ±2.4

Table 13: Pass@32 differences from confidence order across sampling temperature T and commit width on LLaDA/HumanEval; n is samples per problem and CF abbreviates candidate-first. Within this sweep, positive detected differences occur only at T=0.6, and confidence-order coverage peaks at T=1.4 before collapsing. Bold values have 95% intervals excluding zero.

regime	decoder	T	n	distinct-4g	Pass@1	Pass@32	ΔPass@32 vs confidence
one token per step	confidence	0.2	32	0.082	40.2	50.6	—
	LTR order	0.2	32	0.164	39.4	55.5	**+4.9 ±4.0**
	confidence	0.4	32	0.132	40.1	56.1	—
	LTR order	0.4	32	0.259	37.7	60.4	+4.3 ±4.3
	confidence	0.6	128	0.173	38.8	57.5	—
	LTR order	0.6	128	0.349	35.2	63.7	**+6.3 ±3.1**
	CF	0.6	128	0.361	32.6	60.5	**+3.0 ±2.9**
	confidence	0.8	32	0.210	37.9	58.5	—
	confidence	1.0	128	0.247	37.0	59.4	—
	LTR order	1.0	128	0.587	24.3	60.6	+1.2 ±2.7
	CF	1.0	128	0.521	14.5	54.2	**-5.2 ±3.8**
	confidence	1.2	32	0.293	35.8	62.2	—
	confidence	1.4	128	0.366	31.6	63.2	—
	LTR order	1.4	128	0.949	0.1	3.7	**-59.5 ±6.5**
	CF	1.4	128	0.849	1.0	17.8	**-45.4 ±6.1**
	confidence	1.6	32	0.509	19.4	57.9	—
	confidence	1.8	32	0.726	6.1	39.0	—
one token per step, Dream	confidence	0.6	128	0.191	43.4	71.5	—
	LTR order	0.6	128	0.435	46.6	79.4	**+7.9 ±3.8**
	CF	0.6	128	0.406	41.6	78.8	**+7.3 ±3.3**
	confidence	1.0	32	0.322	43.1	76.2	—
	LTR order	1.0	32	0.729	27.3	75.0	-1.2 ±4.3
	CF	1.0	32	0.554	20.7	74.4	-1.8 ±4.0
	confidence	1.4	32	0.579	26.2	72.6	—
	LTR order	1.4	32	1.000	0.0	0.0	**-72.6 ±6.7**
	CF	1.4	32	0.885	1.1	21.3	**-51.2 ±7.6**
two tokens per step	confidence	0.6	128	0.290	30.6	56.6	—
	LTR order	0.6	128	0.526	17.6	53.8	**-2.8 ±2.2**
	CF	0.6	32	0.399	27.2	58.5	+1.9 ±3.2
	confidence	1.0	32	0.377	27.3	59.1	—
	LTR order	1.0	32	0.855	3.9	30.5	**-28.7 ±7.0**
	CF	1.0	32	0.555	16.2	52.4	**-6.7 ±4.9**
	confidence	1.4	32	0.532	16.6	52.4	—
	LTR order	1.4	32	0.970	0.0	0.0	**-52.4 ±7.9**
	CF	1.4	32	0.833	3.8	35.4	**-17.1 ±5.8**

C.2 What secondary results bound the interpretation?

The remaining tables collect three forms of supporting evidence. Transfer across models and benchmarks, comparisons among targeted and simple selectors, and observational commit-time diagnostics. We group them here as secondary audits rather than additional headline claims.

Table 14: Parallel-regime Pass@32 differences from confidence order on LLaDA at width two and T=0.6, across three benchmarks. Confidence order itself reaches 96.8, 56.6 and 68.2 on GSM8K, HumanEval and MATH500. Samples per problem are n=32 except LLaDA/HumanEval, which has n=128. Bold values have 95% intervals excluding zero.

Decoder	GSM8K (500)	HumanEval (164)	MATH500 (500)
candidate-first	+0.0 ±1.1	+1.9 ±3.2	+1.2 ±2.6
entropy-first	**-6.4 ±2.5**	**-40.2 ±7.1**	**-10.6 ±4.3**

Table 15: Nemotron Pass@k on GSM8K and MATH500 with n=32 samples per problem, comparing one-token-per-step decoding with the official confidence-gated parallel mode. Paired Pass@32 differences from confidence order in the same regime are, on MATH500, $+3.2$ $[+0.8, +5.6]$ for LTR and $+1.8$ $[-0.6, +4.2]$ for candidate-first at one token per step, and $+2.0$ $[-0.4, +4.4]$ for LTR and $+3.0$ $[+0.6, +5.4]$ for candidate-first when gated. Every GSM8K difference lies within ± 1.6 of zero. Only the two intervals excluding zero support a claim.

		GSM8K (500)			MATH500 (500)		
Regime	Decoder	@1	@8	@32	@1	@8	@32
1 tok/step	confidence	88.9	96.4	97.4	50.6	67.9	73.8
	LTR	89.8	96.7	97.6	49.0	68.9	77.0
	candidate-first	83.7	96.4	98.0	44.6	67.4	75.6
gated par.	confidence	88.2	96.3	97.4	48.9	67.8	73.8
	LTR	88.0	96.9	98.0	46.1	68.1	75.8
	candidate-first	84.7	96.4	97.6	45.4	67.3	76.8

Table 16: Early-commitment selector comparison on LLaDA/HumanEval. The q=0.90 row promotes positions above the selector's 90th-percentile threshold. Calibrated rules target similar flag rates, but their realized promotion counts differ. Differences use confidence order as control; bold marks 95% intervals excluding zero.

arm	n	distinct-4g	Pass@1	Pass@32	ΔPass@32 vs control
commit early, confidence loop, predicted-high-influence positions committed ahead of the queue					
confidence order	128	0.173	38.8	57.5	—
candidate-first (dictionary)	128	0.361	32.6	60.5	**+3.0** ±2.9
learned, q=0.90	32	0.521	8.1	40.9	**-16.6** ±5.8
learned, calibrated	128	0.208	34.5	60.9	**+3.4** ±2.3
entropy-ranked, calibrated	128	0.271	37.0	60.2	**+2.7** ±2.0
uniformly random, calibrated	128	0.273	34.7	60.1	**+2.6** ±2.0

Table 17: Controls that raise sampling temperature only at selected positions on LLaDA/HumanEval. The temperature T_{hot} applies only at those positions. Each influence-based arm changes the same number of positions as its entropy-based control. Pass@32 differences compare adjacent matched rows; n is samples per problem and bold marks 95% intervals excluding zero.

arm	n	distinct-4g	Pass@1	Pass@32	ΔPass@32 vs control
same number of positions, selected by entropy or influence					
entropy-selected, T_{hot}=1.4	32	0.219	35.6	57.9	—
influence-selected, T_{hot}=1.4	32	0.241	34.7	57.9	+0.0 ±3.7
entropy-selected, T_{hot}=2.0	32	0.339	11.9	51.2	—
influence-selected, T_{hot}=2.0	32	0.353	10.7	48.8	-2.4 ±4.9

Table 18: Commit-time entropy, within-block influence, and final-answer changes. Here $\tilde{H}$ is median entropy at candidate commitment; the adjacent column counts candidate commits per sample. *Early* leaves $|R| = 25$–31 positions masked and *late* leaves $|R| = 1$–8. Influence and outcome probes use LLaDA. All entries are descriptive; branch pairs are nested within problems.

	LLaDA GSM8K		LLaDA MATH500		Dream GSM8K	
decoder	$\tilde{H}$ (nats)	count	$\tilde{H}$ (nats)	count	$\tilde{H}$ (nats)	count
confidence	0.51	6.1	0.87	4.5	0.67	6.5
LTR order	1.26	5.4	2.00	4.1	1.61	5.7
candidate-first	2.24	12.4	2.77	11.8	2.36	12.1
	early	late	early	late	early	late
candidate-position influence (nats)	1.10	0.06	1.01	0.07	—	
other-position influence (nats)	1.18	0.71	1.17	1.12	—	
answers changed	28%	25%	54%	46%	—	

D SUPPORTING FIGURES

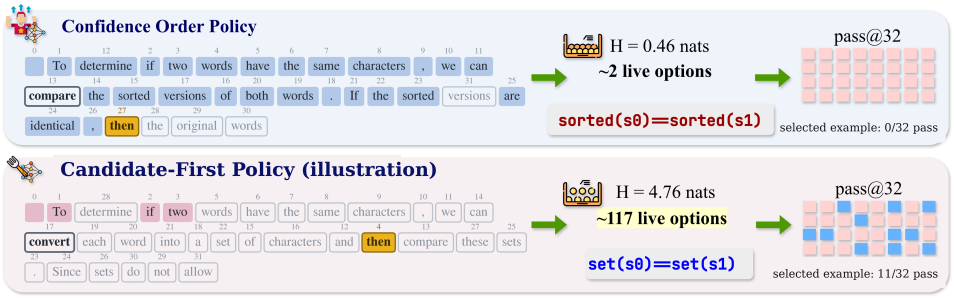

Figure 3: Illustrative candidate-first trajectory. Confidence order commits the candidate *then* position late, whereas candidate-first (called `fork_first` in the artifact) commits it earlier and yields different subsequent continuations. This example illustrates the intervention and carries no aggregate evidential weight; it is not evidence that the detector identifies semantic reasoning forks.

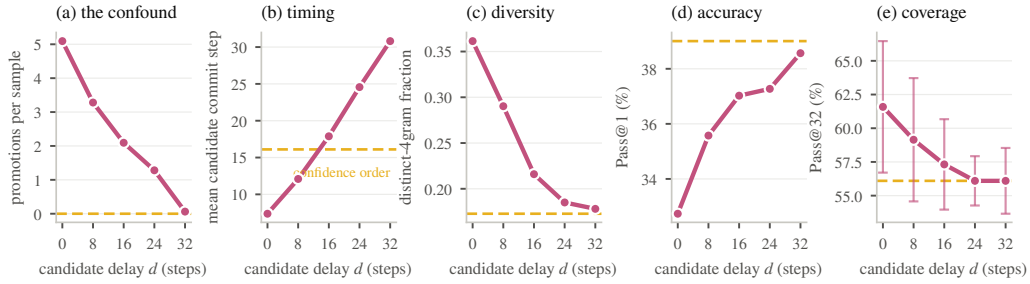

Figure 4: Why a simple delay does not isolate timing. Panel (a) is the confound: promotions per sample collapse from 5.09 to 0.06 across the same sweep that shifts timing in (b), so the movement in diversity (c), Pass@1 (d) and Pass@32 (e) cannot be read as an effect of timing. Dashed lines mark confidence order. These arms use the exploratory n=32 cohort; the count-matched control in Table 6 uses n=128.

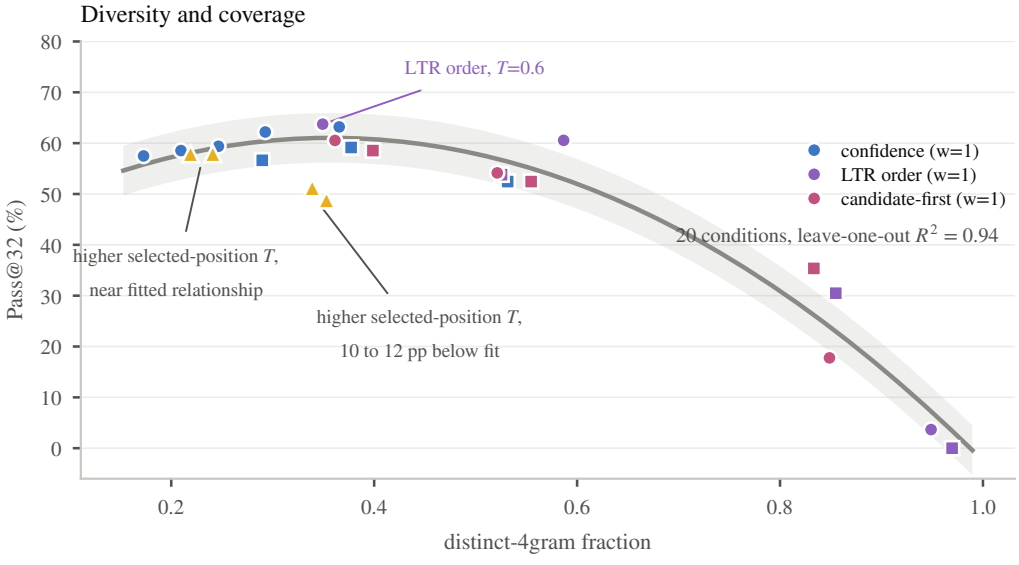

Figure 5: The high overall diversity fit is misleading in the competitive region. This post-hoc fit tracks broad changes and severe failures but does not transfer to the restricted operating range or to conditions that raise temperature only at selected positions and that we excluded from fitting.