WHEN TO COMMIT? VARIABLE-SIZE SELF-CONTAINED BLOCKS FOR DISCRETE DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete diffusion language models (dLLMs) enable parallel token updates with bidirectional attention, yet typical generation adopts blockwise semi-autoregressive decoding, where fixed-size or heuristic blocks are committed to the prefix. This creates a **future-dependent commitment problem**: tokens may be frozen before enough future context is available, causing early local errors to propagate across later blocks. Motivated by this, we propose **Variable-Size Self-Contained Blocks (VSB)**, a commitment criterion that replaces fixed boundaries with an operational self-containedness test. VSB follows a propose-then-test design, where the bidirectional dLLM first proposes a window hypothesis, then compares the predictive distributions under two views of that hypothesis: a **No-Future** view truncated at a candidate boundary and a **Future-Aware** view over the full window. Low NF-FA divergence indicates that the candidate block remains consistent even when the hypothesised future is hidden, making it a self-contained block to commit. VSB applies to pretrained dLLMs and can be paired with optional No-Future training for stronger alignment. Across reasoning, code, and general knowledge benchmarks, VSB improves over fixed-size and heuristic block baselines. Reference code is at this anonymised repo: https://anonymous.4open.science/r/VSB-code

1 INTRODUCTION

Discrete diffusion language models (dLLMs) generate tokens by iteratively denoising masked token sequences with bidirectional attention, allowing parallel token refinements instead of strictly left-to-right autoregression (Nie et al., 2025; Ye et al., 2025; Li et al., 2025; Liu et al., 2026a; Ni et al., 2025; Shi et al., 2024). In practice, this parallelism is usually realised through **blockwise semi-autoregressive decoding**: the model denoises a block with a fixed window size budget (e.g., 32 tokens), commits them to the generated context, and advances to the next block (Arriola et al., 2025; Chen et al., 2026a; Wu et al., 2026b).

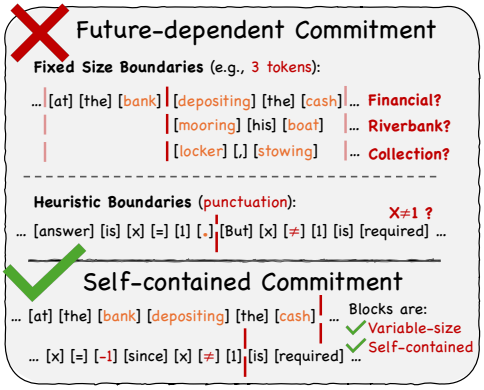


Figure 1: **Future-dependent vs. self-contained commitments in dLLMs.** Fixed or heuristic boundaries can commit tokens whose meaning relies on future context, while self-contained commitment preserves semantic coherence and reduces future dependence.

The commitment step is where blockwise decoding becomes fragile: once a boundary is chosen, the model commit the tokens and use them as conditioning context for all later generation. This is risky because tokens that appear plausible before the boundary may be revised once future context is revealed. However, fixed or heuristic boundaries, such as fixed block sizes or punctuation rules (Arriola et al., 2025; Chen et al., 2026a; Wu et al., 2026b; Lu et al., 2026; Li et al., 2026; Yang et al., 2025; Xiong et al., 2025), do not check this dependency before committing, and can therefore commit tokens before the local content is resolved. Shown in Figure 1, a punctuation-based rule may stop after a comma or closing bracket, while a fixed boundary may cut through an arithmetic expression or semantic unit. In both cases, the

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.

committed tokens can later become inconsistent with the continuation. Since committed tokens serve as fixed conditioning context for subsequent blocks, such premature commitments can cause errors to propagate. We define this as the **future-dependent commitment** problem. This is especially harmful in reasoning tasks, where a symbol, number, or operator may look locally valid but become incorrect once the surrounding expression is completed. While recent re-masking based decoding methods propose to revisit earlier committed tokens (Huang et al., 206; Wang et al., 2026a), many current approaches still rely on fixed block schedules (Chen et al., 2026b; Cheong et al., 2026; Ma et al., 2026a). The central question is therefore **deciding when a candidate block has become stable enough to commit**.

In light of the above question, we propose the **self-containedness** principle. A candidate block is self-contained when its predictions remain stable under two counterfactual views of the same decoding state: a **No-Future** (NF) view that hides tokens beyond a candidate boundary, and a **Future-Aware** (FA) view that reveals the model's current hypothesis for the full window. If these two views induce similar predictive distributions inside the candidate block, then the block is less dependent on the unresolved future and is a better commitment candidate.

To operationalise self-containedness, we introduce **Variable-Size Self-Contained Blocks (VSB)**, a decoding framework that selects variable-size commitment boundaries using NF-FA divergence. Given a candidate boundary, if the tokens before the boundary receive consistent predictions under FA and NF conditions, the block constrained by the boundary is **self-contained and safe to commit**; if they diverge, the **boundary is premature**. This operational self-containedness criterion preserves the strong commitment intuition without fixed lengths or punctuation heuristics. VSB primarily addresses the inference commitment decision, with an optional blockwise No-Future training enhancement that supervises selected blocks under the same truncated conditioning used at commitment time. Our core contributions are:

- **Principle:** We formalise **Self-Containedness** as a criterion for determining block boundaries in blockwise semi-AR decoding. A block is committed only when revealing future tokens does not largely alter its predictive distributions.
- **Method:** We introduce VSB, a decoding paradigm that selects variable-size commitment boundaries by minimising future dependence while preserving blockwise parallelism.

2 PRELIMINARIES

We first define terminology used throughout the paper. Let $x_{1:L}$ denote a length-L token sequence.

1. **Prefix and commitment.** During block decoding, the model maintains a committed prefix $x_{1:p}$, ending at position p. **Committing** means appending the newly generated block of tokens to this prefix. Once committed, these tokens are treated as fixed context and non-revisable.
2. **Block budget and candidate boundary.** We restrict the size of a decoding window to a fixed **block budget** of length W (e.g., 64 tokens). We refer to $(p : q]$ as the budgeted block where $q = \min(p + W, L)$, and $b \in (p : q]$ is a **candidate block boundary** within the budgeted block.

2.1 DISCRETE MASKED DIFFUSION

Discrete masked diffusion training on the full token sequence for text. Let $\mathcal{V}$ be the vocabulary. Given an input token sequence $x_{1:L} \in \mathcal{V}^L$ of length L and diffusion step $t \in \{1, \dots, T\}$, a masked corruption process produces a noisy sequence as: $x^t \sim q_t(\cdot \mid x)$, where some positions are replaced by a special `[MASK]` token $\mathbf{M}$ based on a masking schedule (Nie et al., 2025; Sahoo et al., 2024). A discrete dLLM parameterised by θ outputs per-position logits $\ell_{\theta,i}(x^t, t) \in \mathbb{R}^{|\mathcal{V}|}$ and categorical distributions:

$$p_\theta(x_i \mid x^t, t) = \mathrm{softmax}\big(\ell_{\theta,i}(x^t, t)\big)[x_i]. \tag{1}$$

The standard masked diffusion objective trains on full sequences (Nie et al., 2025; Ye et al., 2025), so the denoiser can attend to all unmasked tokens in $x^t_{1:L}$ when reconstructing masked positions:

$$\mathcal{L}_{\mathrm{D}} = -\mathbb{E}_{(x,t,x^t)} \left[\frac{1}{t} \sum_{i=1}^{L} \mathbf{1}[x_i^t = \mathbf{M}] \log p_\theta(x_i \mid x^t) \right]. \tag{2}$$

Blockwise semi-autoregressive decoding. Standard blockwise decoding partitions generation into fixed-size blocks (Arriola et al., 2025; Wu et al., 2026a; Li et al., 2026; Nie et al., 2025; Liu et al., 2026b; Song et al., 2025). Following the introduced notation, at each step, given a committed prefix ending at position p, the model decodes a fixed *block size*, commits them to the prefix, and then advances. When committing this block, the model conditions only on information up to q, since tokens beyond q have not been generated (e.g., as mask tokens $\mathbf{M}_{q+1:L}$).

2.2 Future-Context Mismatch

Thus, training and inference use different conditional distributions (Sun et al., 2025). During full-sequence training, a token can rely on future suffix context, but during blockwise inference this context may be unavailable, making the token ambiguous (Figure 1). For $i \in (p:q]$, this creates a mismatch:

$$\underbrace{p_\theta(x_i \mid \tilde{x}^t_{1:L}, t)}_{\text{full-sequence masked training}} \neq \underbrace{p_\theta(x_i \mid [\tilde{x}^t_{1:q}; \mathbf{M}_{q+1:L}], t)}_{\text{blockwise decoding}}, \tag{3}$$

Relation to prior blockwise training methods. Existing fixed-block alignment methods (Chen et al., 2026a; Wu et al., 2026a; Arriola et al., 2025; Zhong et al., 2026) restrict future context during training to better match blockwise inference, but usually depend on predefined boundaries such as fixed sizes, punctuation, or line breaks. VSB is complementary: it provides an operational self-containedness criterion for variable boundary selection. See Sec 6 and Appendix D for details.

3 Motivation: Token Commitment via Operational Self-Containedness

Given a budgeted window $(p:q]$, the decoder must decide how many tokens to commit. Committing too few wastes parallelism, while committing too many can freeze predictions that still depend on unresolved continuation. We thus view boundary selection as a commitment-stability problem: *commit only when the candidate block is self-contained under the model's current decoding state*.

Operationally, a block is self-contained if its predictions remain stable when the hypothesised future within the current window is hidden versus revealed. This requires no ground-truth continuation; it tests whether the model's own current continuation makes the block future-sensitive.

3.1 Formalising Self-Containedness

Following the defined notations in Sec 2. Let p denote the end of the committed prefix, and $q = \min(p + W, L)$ be the end of a budgeted decoding window. For a candidate boundary b with $p < b \leq q$, the candidate committed block is $(p:b]$, and the hypothesised future inside the window is $(b:q]$ with tokens beyond q as masked $\mathbf{M}_{q+1:L}$, which we omit in the following for clarity:

$$\mathrm{Block}(p, b) = (p:b], \mathrm{Future}(b, q) = (b:q]. \tag{4}$$

A candidate block ending at b is self-contained if predictions within $\mathrm{Block}(p, b)$ remain consistent after revealing $\mathrm{Future}(b, q)$. We formalise this using two conditioning semantics:

Definition 3.1: NF / FA Views

Let $\hat{x}_{1:q}$ denote the model's current decoding hypothesis over a prefix $1:p$ for a budgeted window $(p:q]$. For a candidate boundary b with $p < b \leq q$, the candidate block is $(p:b]$ and the hypothesised future is $(b:q]$.

For each position $i \in (p:b]$, let $\mathrm{mask}_i(\hat{x}_{1:q})$ denote the sequence of replacing position i with the mask token $\mathbf{M}$ while keeping the remaining hypothesis tokens fixed. The hypothesis $\hat{x}$ is fixed during boundary scoring. We have:

No-Future (NF) predictive distribution:

$$P^{\mathrm{NF}}_{\theta,i}(p, b; \hat{x}, t) = p_\theta\left(X_i = \cdot \mid \mathrm{mask}_i(\hat{x}_{1:b}), t\right). \tag{5}$$

Future-Aware (FA) predictive distribution:

$$P^{\mathrm{FA}}_{\theta,i}(p, q; \hat{x}, t) = p_\theta\left(X_i = \cdot \mid \mathrm{mask}_i(\hat{x}_{1:q}), t\right). \tag{6}$$

Self-containedness Divergence. We quantify the **operational self-containedness** of a candidate block with boundary b as the average token-level KL divergence $\mathcal{D}_{KL}$ between these predictive distributions. We use $S(p, b)$ for brevity:

$$S(p, b; q, \hat{x}, t) = \frac{1}{b-p} \sum_{i=p+1}^{b} \mathcal{D}\big(P_{\theta,i}^{\mathrm{NF}} \,\|\, P_{\theta,i}^{\mathrm{FA}}\big)\,. \tag{7}$$

DEFINITION 3.2: ε-self-contained block

A block $\mathrm{Block}(p, b)$ is ε-self-contained under hypothesis $\hat{x}_{1:q}$ and step t if $S(p, b; \hat{x}, t) \le \varepsilon$.

Intuition: The hypothesis $\hat{x}_{1:q}$ supplies candidate token values; the candidate block is operationally self-contained when the NF and FA conditionals are close for tokens inside $(p : b]$. It measures how strongly, on average, predictions for $x_{p+1:b}$ depend on future context beyond b. Intuitively, **the lower the divergence, the more self-contained the block is**. Fig. 2 shows that self-containedness prefers semantically complete boundaries over fixed-size blocks.

Why Fixed-Size Is Structurally Limiting. A fixed-size strategy commits W tokens each step and is reliable only when $S(p, p+W)$ is small. Since future dependence varies across blocks, fixed W can be too conservative in easy regions or overcommit in sensitive ones, motivating variable boundaries. See formal discussion in Appendix B.

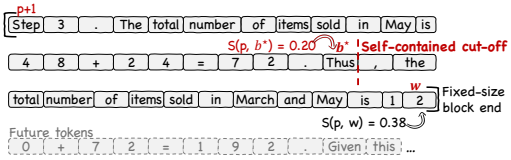


Figure 2: Fixed-size blocks may cut off highly future-dependent spans, while **VSB selects more semantically complete boundaries**.

3.2 What Does Self-Containedness Certify?

Self-containedness can thus be viewed as a commitment stability condition. The key question therefore becomes whether revealing the hypothesised future would largely change predictions inside the candidate block. The following results show that low self-containedness divergence controls this model-internal commitment instability.

Interpretation. Lemma 3.3 implies that when the self-containedness divergence is small, revealing the context after b induces only a small change in the model's predictive distributions within the block, on average. The proof is in Appendix C.1.

LEMMA 3.3: NF-FA divergence controls predictive future-sensitivity

If a candidate block $(p : b]$ is ε-self-contained, then the average total-variation shift between NF and FA predictions is bounded:

$$\frac{1}{b-p} \sum_{i=p+1}^{b} \delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right) \le \sqrt{\frac{\varepsilon}{2}}.$$

Interpretation. Prop. 3.4 connects distribution-level stability to commitment decision. A token can be safely committed when revealing the context after b causes only a small change to its prediction distribution, so that the top-1 prediction remains stable. Proof is in Appendix C.2.

PROPOSITION 3.4: Top-1 token stability

For a position i, let $y_i^{\mathrm{NF}} = \arg\max_y P_{\theta,i}^{\mathrm{NF}}(y)$, and define the NF margin:

$$\gamma_i = P_{\theta,i}^{\mathrm{NF}}(y_i^{\mathrm{NF}}) - \max_{y \neq y_i^{\mathrm{NF}}} P_{\theta,i}^{\mathrm{NF}}(y).$$

If $\delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right) < \frac{\gamma_i}{2}$, then the top-1 prediction remains unchanged after revealing the hypothesised future:

$$\arg\max_y P_{\theta,i}^{\mathrm{NF}}(y) = \arg\max_y P_{\theta,i}^{\mathrm{FA}}(y).$$

Interpretation. Corollary 3.5 lifts the token-level result to block level: under a clear NF top-1 margin, $S(p, b)$ upper-bounds the fraction of predictions that may change after revealing the hypothesised

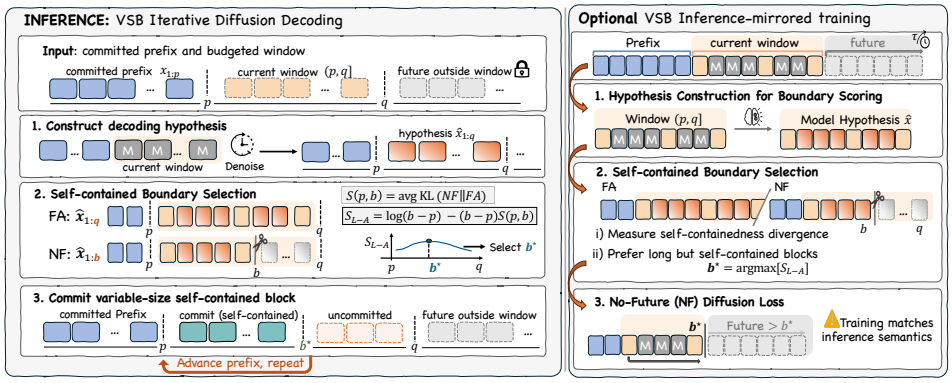


Figure 3: **Overview of VSB.** At inference, diffusion decoding commits the block that best balances length and self-containedness. Optional inference-mirrored training acts as alignment enhancement.

future, giving it an operational interpretation as commitment risk. Proof is in Appendix C.4. Appendix H.2 measures the empirical $\gamma_{\min}$ distribution, where the deferred regions are $1.7\times$ less stable than what VSB commits.

COROLLARY 3.5: Low NF-FA divergence bounds unstable block commitments

Let $n = b - p$. For each position $i \in (p : b]$, define the unstable-commitment indicator:

$$U_i = \mathbf{1}\left[\arg\max_y P^{\mathrm{NF}}_{\theta,i}(y) \neq \arg\max_y P^{\mathrm{FA}}_{\theta,i}(y)\right].$$

Let γ_i be the NF top-1 margin from Proposition 3.4. If every position in the candidate block has margin at least $\gamma_{\min} > 0$, then:

$$\frac{1}{n}\sum_{i=p+1}^{b} U_i \leq \frac{2S(p,b)}{\gamma^2_{\min}}.$$

4 METHOD: VSB

We now present Variable-size Self-contained Blocks (VSB), a blockwise decoding paradigm. At each step, VSB **(i) forms a decoding hypothesis** over a budgeted window (Section 4.1), and **(ii) selects the commitment boundary** with a coarse-to-fine scan of the window's self-containedness (Section 4.2). Section 4.3 assembles the two into the decoding loop. We also introduce an optional blockwise No-Future training enhancement that aligns supervision with the information pattern used at commitment time. Figure 3 provides an overview.

4.1 HYPOTHESIS CONSTRUCTION FOR BOUNDARY SCORING

At inference time the true future tokens are unavailable, so VSB scores candidate boundaries against the model's own continuation. Given a committed prefix ending at p, VSB first denoises the budgeted window $(p : q]$ to form a hypothesis $\hat{x}_{1:q}$, and then computes the NF and FA distributions of Definition 3.1 on this fixed hypothesis. Because only future access differs between the two views, any NF-FA divergence is attributable to the hypothesised future rather than to changes in decoding state. Notably, the hypothesis is not assumed to be correct. Instead, we use it to test how strongly the current block depends on the continuation, and the divergence thus indicates whether the block is ready to be committed. Section 4.2 defines how boundaries are scored and selected on this hypothesis.

Inference hypothesis. The prefix consists of previously committed generated tokens. We run the diffusion denoising process in the budgeted window and take an intermediate partially decoded snapshot $\tilde{x}^{\,t_{\mathrm{hyp}}}$ as the boundary-scoring hypothesis (e.g., step $\frac{T}{2}$). Fig. 3 illustrates this construction.

$$\hat{x}_{p+1:q} = \tilde{x}^{\,t_{\mathrm{hyp}}}_{p+1:q}. \tag{8}$$

4.2 LENGTH-AWARE SELF-CONTAINED BOUNDARY SELECTION.

Given a hypothesis $\hat{x}_{1:q}$, VSB evaluates candidate boundaries $b \in \mathcal{B}(p,q) = \{p+1, \dots, q\}$ with block budget $W = q - p$ (Section 2) using the self-containedness divergence in Eq. 7. A hard rule could commit the longest block whose divergence falls below a threshold (e.g., $\max_b \; b-p \;$ s.t. $\; b \in \mathcal{B}(p,q), \;\; S(p,b) \le \varepsilon$). In practice however, threshold selection can be brittle across tasks, model calibrations, and decoding states. Thus, we use a smooth length-aware objective:

$$S_{\text{L-A}} = \log(b-p) - (b-p)\, S(p,b), \tag{9}$$

and select: $$b^\star = \arg\max_{b \in \mathcal{B}(p,q)} S_{\text{L-A}}. \tag{10}$$

Intuition. Since $S(p,b)$ is an *average* per-token future sensitivity inside $(p,b]$, multiplying by $(b-p)$ yields the *total future-dependence mass* of the block. This matches the commitment risk: once committed, all tokens in the block are frozen, so we penalise how much future-dependent content we are locking in, not merely whether the average token is stable. **Intuitively**, the first term favors longer commits (diminishing returns via $\log$), while the second discourages committing blocks whose total future dependence is large. The relationship between b, S and $S_{\text{L-A}}$ is provided in Figure 4.

Efficient coarse-to-fine boundary selection. Evaluating all W candidates is expensive, as it would require one NF forward for each. VSB therefore selects the boundary in two stages (Algorithm 1):

1. **Coarse scan:** evaluate every s_c-th candidate to find the most promising region around $\tilde{b}$;
2. **Fine scan:** search this region with smaller stride s_f to select the final boundary $b^\star$.

Let $\mathcal{B}_c = \{p + s_c,\, p + 2s_c, \dots, q\}$ denote the coarse candidates, and let $\mathcal{B}_f = \{\tilde{b} - s_c, \dots, \tilde{b} + s_c\} \cap \mathcal{B}(p,q)$, sampled at stride s_f, be the fine-search region (e.g., $s_c = 16, s_f = 4$). The selected boundaries are:

$$\tilde{b} = \arg\max_{b \in \mathcal{B}_c} S_{\text{L-A}}(b), \qquad b^\star = \arg\max_{b \in \mathcal{B}_f} S_{\text{L-A}}(b). \tag{11}$$

The coarse-to-fine search therefore **avoids evaluating every possible boundary**, and evaluates approximately $\frac{W}{s_c} + \frac{2s_c}{s_f}$ candidates. This **approximation works well** because $S_{\text{L-A}}$ changes smoothly across nearby boundaries (Figure 4). Moreover, setting $s_c = 1$ recovers a dense scan over all W candidates, and we show that this dense scanning marginally improves accuracy by at most 0.4% while requiring $5.4\times$ more scoring forwards (Table 2). Finally, since NF and FA differ only in whether future tokens are visible, the theoretical guarantees of Section 3.1 remain applicable.

4.3 FINAL VSB INFERENCE PROCEDURE

Consequently, at inference time, VSB alternates between diffusion denoising a budgeted window and selecting a self-contained commitment boundary, as shown in Algorithm 2. Given a committed prefix $x_{1:p}$, the model denoises the budgeted window $(p : q]$ and constructs an intermediate hypothesis $\hat{x}_{1:q}$ following Section 4.1. VSB then selects $b^\star$ by Eq. 11 and repeats untill generation completes:

$$x_{1:p} \leftarrow \hat{x}_{1:b^\star}, \qquad p \leftarrow b^\star.$$

4.4 OPTIONAL ALIGNMENT VIA BLOCKWISE NO-FUTURE TRAINING

While VSB is designed as a decoding method, we also introduce an **optional** blockwise No-Future training objective. The goal is not to claim complete training-inference equivalence. Instead, the objective aligns supervision with the information available at blockwise decoding time.

Training hypothesis. The prefix $x_{1:p}$ is taken from the ground-truth sequence, and the active budgeted window is initially masked. We construct a model hypothesis at step t_{hyp}:

$$\hat{x}_{p+1:q} = \arg\max_{y} p_\theta\left(y \mid x_{1:p}, \mathbf{M}_{p+1:q}, t_{\text{hyp}}\right). \tag{12}$$

The training and inference hypotheses are related but not identical: training may use gold prefixes, while inference uses generated prefixes.

Training mirrors inference: For each training example, we sample a prefix end position p, define $q = \min(p + W, L)$, construct the hypothesis $\hat{x}_{1:q}$ using Eq. 12, and select $b^\star$ using Eq. 11. We then supervise only positions inside $(p : b^\star]$ under truncated No-Future conditioning. Let $\mathcal{M} \subseteq (p : b^\star]$

Method	*Math & Science* GSM8K (512, 0)	MATH500 (512, 4)	GPQA-Diamond (64, 0)	*Code* HumanEval (512, 0)	MBPP (256, 3)	*General Knowledge* MMLU_Gen (3, 0)	HellaSwag (10, 0)
dLLM-Var	73.92	34.00	23.73	34.76	**40.20**	61.71	71.48
Blockwise-SFT	75.06	37.80	26.26	37.20	35.60	62.81	73.10
D2F	74.90	36.00	20.20	40.24	34.60	62.71	73.33
AdaBlock	75.66	38.20	**28.79**	**45.12**	37.60	<u>62.98</u>	73.94
DAEDAL	<u>80.74</u>	<u>38.60</u>	<u>28.28</u>	39.02	36.00	62.58	<u>75.60</u>
LLaDA-8B	76.64	35.80	26.76	37.20	35.20	62.49	72.74
VSB w/ LLaDA-8B	**80.97**$_{+4.33\%}$	**39.00**$_{+3.20\%}$	<u>28.28</u>$_{+1.52\%}$	<u>41.46</u>$_{+4.26\%}$	<u>38.00</u>$_{+2.80\%}$	**63.15**$_{+0.66\%}$	**76.57**$_{+3.83\%}$
+ *NF Tr. Align.* (Eq. 13)	81.12$_{+4.48\%}$	39.00$_{+3.20\%}$	28.79$_{+2.27\%}$	43.29$_{+6.09\%}$	38.60$_{+3.40\%}$	63.11$_{+0.62\%}$	76.68$_{+3.94\%}$
LLaDA-1.5	79.38	37.00	27.78	36.59	35.60	62.52	72.34
VSB w/ LLaDA-1.5	83.09$_{+3.71\%}$	40.40$_{+3.40\%}$	26.26$_{-1.52\%}$	43.29$_{+6.7\%}$	40.00$_{+4.40\%}$	63.31$_{+0.79\%}$	76.40$_{+4.06\%}$
+ *NF Tr. Align.* (Eq. 13)	83.40$_{+4.02\%}$	39.60$_{+2.60\%}$	28.28$_{+0.50\%}$	46.95$_{+10.36\%}$	39.80$_{+4.20\%}$	62.94$_{+0.42\%}$	76.93$_{+4.59\%}$
LLaDA-2.0-mini	85.06	49.00	31.82	70.12	53.80	68.97	82.49
VSB w/ LLaDA-2.0-mini	88.85$_{+3.79\%}$	55.20$_{+6.20\%}$	32.83$_{+1.01\%}$	75.61$_{+5.49\%}$	55.00$_{+1.20\%}$	70.66$_{+1.69\%}$	82.81$_{+0.32\%}$

Table 1: Overall performance (**Accuracy/Pass@1 (%)**) of VSB against baselines. Numbers in parentheses denote **token budget** and **number of shots**. For fair comparison, best among the **LLaDA-8B based methods** are **bolded** and Runnerups are <u>underlined</u>, respectively. The **relative improvements** of **VSB against the backbone** are labelled.

denote the masked positions inside the selected block at diffusion step t, and $\alpha(p, b^\star) = \frac{b^\star - p}{q - p}$ be a length-dependent weighting factor that discourages degenerate short blocks (e.g., 1 token). The training process is in Algorithm 3 Appendix E, and the No-Future diffusion loss is:

$$\mathcal{L}_{\text{NF}} = \alpha(p, b^\star) \cdot \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} -\log p_\theta(x_i \mid \tilde{x}^t_{1:b^\star}, t). \tag{13}$$

5 Experiments

Datasets and Evaluation. We evaluate on standard LLM benchmarks spanning mathematical & scientific reasoning (GSM8K, MATH500, GPQA-Diamond), code generation (HumanEval, MBPP), and general knowledge (MMLU_Generative, HellaSwag). We report accuracy/pass@1 (%), with more details provided in Appendix F.

Baselines. We compare VSB against blockwise-training and popular models with semi-AR decoding : **(1)** LLaDA family (`LLaDA-8B-Instruct`, `LLaDA-1.5`, and `LLaDA-2.0-mini` (Nie et al., 2025; Zhu et al., 2025; Bie et al., 2025); **(2)** `dLLM-Var` (Yang et al., 2025); **(3)** `Blockwise-SFT` (Sun et al., 2025); **(4)** `D2F` (Wang et al., 2026b); and adaptive block methods **(5)** `AdaBlock` (Lu et al., 2026) and **(6)** `DAEDAL` (Li et al., 2026).

Implementations. We use LLaDA-8B-Instruct as the backbone for all training-based and post-hoc baselines. Dataset-specific token budgets, block lengths, and few-shot counts follow standard settings from prior work (Zhou et al., 2026; Nie et al., 2025; Lu et al., 2026) and are listed in Table 1. For fair comparison, we match the inference-time budget W to the fixed block length used by each baseline. We use the training split of `openai/gsm8k` (Cobbe et al., 2021) for supervised fine-tuning. The default coarse and fine strides are $s_c = \min(\lceil \frac{\text{token budget}}{2} \rceil, 16)$, and $s_f = \min(\lceil \frac{\text{token budget}}{5} \rceil, 4)$, respectively. Since LLaDA-2.0 is already trained with block-wise attention, we omit the optional *NF* training (Eq. 13). Full details are in Appendix G.

Evaluation focus. We highlight that VSB is a commitment-quality method, not a free acceleration method, thus we focus on evaluating whether self-contained boundary selection improves generation quality rather than pure speed. We discuss the trade-off in Section 5.3, Limitations, and Appendix H.3.

5.1 Overall Performance

Table 1 shows that **VSB consistently improves over its backbones and is competitive with prior blockwise decoding and training approaches.** Gains are especially visible on tasks where premature commitment can disrupt multi-step dependencies. On general knowledge tasks, improvements are smaller but positive, suggesting that operational self-containedness does not trade-off broad language modelling capability. Adding the optional training alignment further improves performance. The results support our self-contained commitment.

5.2 Boundary Selection by Self-Containedness

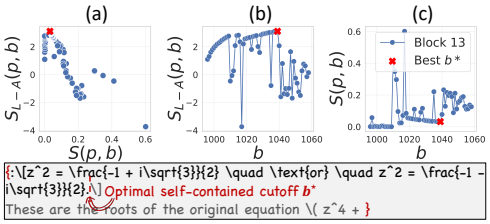


Figure 4: $S(p,b)$ vs. length-aware $S_{\mathrm{L-A}}(p,b)$ across a budgeted block ($W = 64$) starting at prefix $p = 994$.

VSB selects self-contained boundaries by balancing block length against NF-FA divergence, rather than fixed or heuristic cutoffs. Figure 4 shows how self-containedness is used to select a block boundary for a fixed prefix ending at position p (here p=994). Each candidate $b \in (p, p+W]$ is scored within a fixed budget W=64. The selected cutoff $b^\star$ maximises the length-aware-utility objective in Eq. 11, choosing a block that is long enough to preserve parallelism while keeping the total future-dependence mass low. Using raw divergence alone can collapse to very short blocks (Figure 4c), the length term avoids this by trading off stability and commitment length. A full decoded-token case study is provided in Appendix J.

VSB commits lower-divergence blocks by construction. Figure 5 shows that fixed-size decoding yields a broader distribution with higher self-containedness divergence $S(p,b)$, while VSB concentrates committed blocks at lower divergence. This confirms the objective's intended behaviour.

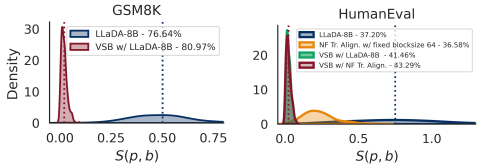


Figure 5: Self-containedness divergence.

5.3 Accuracy-Cost Trade-off Analysis

Metric	MBPP				HellaSwag		GPQA-D		
	$(1,-)$	$(8,2)$	$(16,4)$	$(32,8)$	$(1,-)$	$(5,2)$	$(1,-)$	$(8,2)$	$(16,4)$
Acc.	39.00	38.80	38.60	38.60	76.90	76.68	28.79	28.79	28.79
Speedup	1.0×	1.5×	1.6×	1.6×	1.0×	1.2×	1.0×	1.4×	1.5×

Table 2: Ablation of coarse-to-fine boundary search strides (s_c, s_f) for VSB (w/ NF Tr. Align.). Speedup is measured relative to the full boundary scan per dataset.

Accuracy is not sensitive to the search strides. Table 2 varies the stride pair (s_c, s_f) of the boundary search, down to the dense setting s_c=1 that scores every candidate. Accuracy changes by at most 0.4% across the full range while end-to-end runtime improves by up to 1.6× over the dense setting (MBPP). Appendix H.3 shows VSB matches or exceeds fixed-size decoding at approximately equal forward-pass budgets.

Method	GSM8K		GPQA-D		HellaSwag	
	Acc.	Speedup	Acc.	Speedup	Acc.	Speedup
VSB w/ *NF Tr. Align.*	**81.12**	1.0×	**28.79**	1.0×	**76.68**	1.0×
w/ Cache	80.36	1.2×	28.28	1.2×	75.23	1.3×
w/ Conf. Thresh.	77.48	2.7×	27.27	2.5×	75.97	2.2×

Table 3: Speedups with existing techniques.

VSB is compatible with confidence-threshold decoding and caching. As shown in Table 3, VSB supports confidence-threshold decoding (Wu et al., 2026b; Yu et al., 2026b). Additionally, VSB can cache generated hidden states to reduce redundant computation. However, in the current LLaDA implementation, KV prefix caching can introduce a causal attention bias that affects NF/FA comparisons (Wu et al., 2026b). To preserve correct semantics of self-containedness scoring, we compute NF and FA distributions without caching, and instead reuse intermediate hidden states from the FA forward with our coarse-to-fine search to accelerate boundary scoring. Table 12 in Appendix reports VSB runtime and decoding cost. While incurring slightly higher cost, we believe this is an acceptable trade-off given improved performance. We discuss further in Appendix A Limitations.

5.4 Hyperparameter Sensitivity

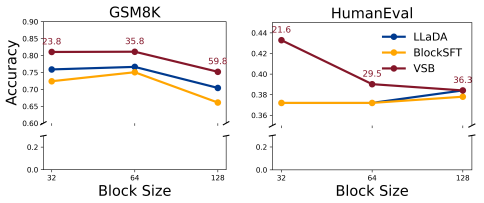


Figure 6: Accuracy vs. block window on GSM8K and HumanEval. The average block size of VSB (w/ NF Tr. Align.) is labelled red.

Shown in Figure 6, **VSB consistently outperforms fixed-size decoding across diverse block budget** W, with the largest gains at small to medium block sizes. Although larger blocks provide more context, committing long blocks can cause unstable predictions. In contrast, VSB commits only self-contained tokens, resulting in shorter and variable average block lengths (as annotated). To further diagnose whether VSB's gains are simply due to shorter effective blocks, Table 13 in the Appendix controls for block length by matching fixed-size LLaDA to VSB's average block size. Under this setting, VSB still outperforms fixed-size decoding at low and medium budgets, showing that its gains

come from selecting better boundaries rather than merely using shorter blocks. This indicates that the improvements stem from boundary selection rather than increased block size, avoiding accumulation of future-dependent tokens.

Table 4 further shows that VSB is not sensitive to the sampling temperature, where the accuracy and average block length remains stable under different temperature values. This suggests that VSB is not dependent on a single deterministic boundary trace.

Temperature	Acc. (%)	Boundary change rate (%)	Avg. block len.
0.0	**85.6**	–	37.47
0.2	85.0	55.3	36.12
0.5	85.2	68.0	35.93

Table 4: Temperature sensitivity of VSB on GSM8K subset. Boundary change is measured relative to 0.0.

5.5 Extended Ablations

Boundary Selection Objective. Table 5 ablates the smooth length-aware objective in Eq. 9. Replacing mean divergence with a maximum term makes decoding overly conservative, while replacing the log-length with a linear reward encourages overcommitment and reduces accuracy. The proposed mean-divergence and log-length utility achieves the best overall accuracy.

Objective form	Acc. (%)	Avg. block len.
Length-aware (Eq. 9)	**81.12**	35.80
Max divergence	80.74	20.48
Linear length	76.19	62.40

Table 5: Ablation of boundary selection objective function (Eq. 9) on GSM8K with VSB w/ NF Tr. Align.

Hypothesis Snapshot and Token-Level Stability. Table 6 ablates which denoising step the hypothesis should be constructed for boundary scoring. The early hypothesis (e.g., Step $\frac{1}{4}T$) moves 74% of boundaries, mostly towards shorter blocks, and lowers accuracy by 4%. This indicates that scoring a hypothesis in which most of the window is still masked therefore yields less stable boundaries and committing more token, while later hypotheses recover final-step accuracy, suggesting that the utilised default middle snapshot ($\frac{T}{2}$) is a practical accuracy-cost trade-off. Appendix Tables 9 and 10 further shows that NF-FA disagreement concentrates on delimiter tokens, arithmetic operators, and code-syntax tokens, where fixed block size can freeze incomplete expressions. These results support our future-dependent commitment motivation.

Hypothesis step	Acc. (%)	Bound. change (%)	Overcommit (%)
Fixed-size baseline	70.8	–	–
Early hypo. $\frac{1}{4}T$	81.2	74.1	29.0
Mid hypo. $\frac{1}{2}T$	**85.6**	25.0	17.0
Final-step hypo.	**85.6**	–	–

Table 6: Hypothesis snapshot ablation on GSM8K subset. Boundary change and overcommit rates are relative to the final-step hypothesis.

6 Related Work

Our work relates to four research areas: (1) **Discrete diffusion language models** denoise masked sequences under bidirectional attention, offering an alternative to AR generation. Representative models include LLaDA (Nie et al., 2025; Zhu et al., 2025) and Dream (Ye et al., 2025). (2) **Blockwise semi-AR decoding** enables practical inference by committing tokens in semi-AR blocks rather than full-context parallel updates (Arriola et al., 2025; Wu et al., 2026a). (3) **Blockwise training alignment** (e.g., Blockwise-SFT) reduces the training-inference mismatch by aligning supervision with fixed-size block commitment (Sun et al., 2025; Chen et al., 2026a). (4) **Heuristic block selection** chooses block boundaries using fixed sizes, punctuation, or other heuristic/learned cues (Li et al., 2026; Lu et al., 2026). VSB differs from prior heuristic boundary schemes by introducing an explicit *self-containedness* criterion (Eq. equation 11) that selects variable-size blocks based on predictive stability with and without future context. A detailed review is provided in Appendix D.

7 Conclusion

We introduced Variable-Size Self-Contained Blocks (VSB), an inference commitment criterion for blockwise decoding in discrete dLLMs. VSB operationalises self-containedness as counterfactual predictive stability: a candidate block is self-contained when its predictions remain stable under No-Future and Future-Aware views of the same model-generated decoding hypothesis. This criterion identifies future-dependent commitments that fixed-size or surface heuristic boundary rules can miss. Across reasoning, code, and general knowledge benchmarks, VSB improves blockwise dLLM decoding while identifying an accuracy-throughput trade-off to inspire future work on efficient self-contained commitment.

AI USE STATEMENT

In this work, we used generative AI tools to assist with validating mathematical proofs, refining technical insights, propose suitable ablation studies, and refactoring the provided reference code. We have not used generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, propose or refine hypotheses, translate content, clean or reformat datasets, support qualitative or thematic data analysis.

Additionally, we used generative AI tools to polish and improve the clarity of the writing in the paper. All AI-assisted mathematical content was manually checked by the authors for correctness and consistency with the stated assumptions and results. AI-assisted code refactoring was reviewed and tested to ensure that it preserved the intended functionality of the original implementation. All AI-assisted writing was reviewed and revised by the authors to ensure technical accuracy and consistency with the work.

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

ETHICS STATEMENT

As a foundational study on discrete diffusion language models, we do not identify any direct negative societal impacts. Our research relies on publicly available datasets and models, all of which are properly acknowledged and pose no risks requiring safeguards. While potential societal implications may exist, none warrant specific emphasis in this study.

REPRODUCIBILITY STATEMENT

We summarise our efforts for reproducible research below:

1. **Metrics.** We report accuracy for reasoning and general knowledge tasks and pass@1 for code-generation tasks. Dataset-specific token budgets, block lengths, few-shot settings, and compute-related measurements are reported in Section 5, Table 1, and Appendix G.
2. **Baselines and Datasets.** We evaluate VSB against fixed-size, training-based, and adaptive blockwise decoding baselines on mathematical reasoning, scientific reasoning, code generation, and general knowledge benchmarks. The baselines and evaluation settings are described in Section 5, while detailed dataset descriptions and few-shot configurations are provided in Appendix F.
3. **Implementation Details.** The software environments, hardware, model backbones, training configuration, and inference hyperparameters are documented in Appendix G. The coarse-to-fine boundary selector, complete VSB inference procedure, and optional No-Future alignment training procedure are given in Algorithms 1, 2, and 3, respectively.
4. **Reference Code.** We provide reference implementation of VSB at given anonymised repository in the abstract.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *ICLR*, 2025.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *NeurIPS*, 2021a.

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. Program synthesis with large language models. *CoRR*, 2021b. URL https://arxiv.org/abs/2108.07732.

Dan Biderman, Jacob P. Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. Lora learns less and forgets less. *TMLR*, 2024.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, Chengxi Li, Chongxuan Li, Jianguo Li, Zehuan Li, Huabin Liu, Ling Liu, Guoshan Lu, Xiaocheng Lu, Yuxin Ma, Jianfeng Tan, Lanning Wei, Ji-Rong Wen, Yipeng Xing, Xiaolu Zhang, Junbo Zhao, Da Zheng, Jun Zhou, Junlin Zhou, Zhanchao Zhou, Liwang Zhu, and Yihong Zhuang. Llada2.0: Scaling up diffusion language models to 100b, 2025. URL https://arxiv.org/abs/2512.15745.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, Zhuocheng Gong, Yanmei Gu, Jian Guan, Kaiyuan Guan, Hongliang He, Zenan Huang, Juyong Jiang, Zhonghui Jiang, Zhenzhong Lan, Chengxi Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Yuan Lu, Yuxin Ma, Xingyu Mou, Zhenxuan Pan, Kaida Qiu, Yuji Ren, Jianfeng Tan, Yiding Tian, Zian Wang, Lanning Wei, Tao Wu, Yipeng Xing, Wentao Ye, Liangyu Zha, Tianze Zhang, Xiaolu Zhang, Junbo Zhao, Da Zheng, Hao Zhong, Wanli Zhong, Jun Zhou, Liwang Zhu, Muzhi Zhu, and Yihong Zhuang. Llada2.1: Speeding up text diffusion via token editing. *CoRR*, 2026. URL https://doi.org/10.48550/arXiv.2602.08676.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, 2021. URL https://arxiv.org/abs/2107.03374.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dparallel: Learnable parallel decoding for dllms. *ICLR*, 2026a.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. Dmax: Aggressive parallel decoding for dllms. *CoRR*, 2026b. URL https://doi.org/10.48550/arXiv.2604.08302.

Shuang Cheng, Yihan Bian, Dawei Liu, Yuhua Jiang, Yihao Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. SDAR: A synergistic diffusion-AutoRegression paradigm for scalable sequence generation. In *ACL*, 2026.

Minsoo Cheong, Donghyun Son, Woosang Lim, and Sungjoo Yoo. Entropycache: Decoded token entropy guided KV caching for diffusion language models. *CoRR*, 2026. URL https://doi.org/10.48550/arXiv.2603.18489.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, 2021. URL https://arxiv.org/abs/2110.14168.

Yonggan Fu, Lexington Whalen, Zhifan Ye, Xin Dong, Shizhe Diao, Jingyu Liu, Chengyue Wu, Hao Zhang, Enze Xie, Song Han, Maksim Khadkevich, Jan Kautz, Yingyan Celine Lin, and Pavlo Molchanov. Efficient-DLM: From autoregressive to diffusion language models, and beyond in speed. In *ICML*, 2026.

Raghavv Goel, Risheek Garrepalli, Sudhanshu Agrawal, Christopher Lott, Fatih Porikli, and Mingu Lee. Skip to the good part: Representation structure & inference-time layer skipping in diffusion vs autoregressive LLM. In *Workshop on Scientific Methods for Understanding Deep Learning*, 2026. URL https://openreview.net/forum?id=rLAhK3dPik.

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. Diffusionbert: Improving generative masked language models with diffusion models. In *ACL*, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. Aligning AI with shared human values. In *ICLR*, 2021a.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *ICLR*, 2021b.

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 *NeurIPS Datasets and Benchmarks*, 2021c.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models. *ICLR*, 206.

Jaeyeon Kim, Cheuk Kit Lee, Carles Domingo-Enrich, Yilun Du, Sham M. Kakade, Timothy Ngotiaoco, Sitan Chen, and Michael S. Albergo. Any-order flexible length masked diffusion. *ICLR*, 2026.

Suayp Talha Kocabay and Talha Rüzgar Akkus. Diffutron: A masked diffusion language model for turkish language. *CoRR*, 2026. URL https://doi.org/10.48550/arXiv.2603.20466.

Jinsong Li, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. Beyond fixed: Training-free variable-length denoising for diffusion large language models. *ICLR*, 2026.

Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. A survey on diffusion language models. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2508.10875.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation. In *NeurIPS*, 2022.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *ICLR*, 2024.

Aiwei Liu, Minghua He, Shaoxun Zeng, Linhao Zhang, Chuhan Wu, Wei Jia, Yuan Liu, Yang Yu, Xiao Zhou, and Jie Zhou. Wedlm: Reconciling diffusion language models with standard causal attention for fast inference. *ICML*, 2026a.

Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyan Wei, Shaobo Wang, Yichen Zhu, and Linfeng Zhang. dLLM-cache: Accelerating diffusion large language models with adaptive caching. In *ICML*, 2026b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Guanxi Lu, Hao Mark Chen, Yuto Karashima, Zhican Wang, Daichi Fujiki, and Hongxiang Fan. Adablock-dllm: Semantic-aware diffusion LLM inference via adaptive block size. *ICLR*, 2026.

Linrui Ma, Yufei Cui, Kai Han, and Yunhe Wang. Mask is what DLLM needs: A masked data training paradigm for diffusion llms. *CoRR*, 2026a. URL https://doi.org/10.48550/arXiv.2603.15803.

Linrui Ma, Yufei Cui, Kai Han, and Yunhe Wang. Diffusion in diffusion: Reclaiming global coherence in semi-autoregressive diffusion. *CoRR*, 2026b. URL https://doi.org/10.48550/arXiv.2601.13599.

Jinjie Ni, Qian Liu, Longxu Dou, Chao Du, Zili Wang, Hang Yan, Tianyu Pang, and Michael Qizhe Shieh. Diffusion language models are super data learners. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2511.03276.

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. In *NeurIPS*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *CoRR*, 2023. URL https://doi.org/10.48550/arXiv.2311.12022.

Subham S. Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *NeurIPS*, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *NeurIPS*, 2024.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, Yuwei Fu, Jing Su, Ge Zhang, Wenhao Huang, Mingxuan Wang, Lin Yan, Xiaoying Jia, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Yonghui Wu, and Hao Zhou. Seed diffusion: A large-scale diffusion language model with high-speed inference. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2508.02193.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *NeurIPS*, 2018.

Bowen Sun, Yujun Cai, Ming-Hsuan Yang, and Yiwei Wang. Blockwise SFT for diffusion language models: Reconciling bidirectional attention and autoregressive decoding. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2508.19529.

Yuchuan Tian, Yuchen Liang, Jiacheng Sun, Shuo Zhang, Guangwen Yang, Yingte Shu, Sibo Fang, Tianyu Guo, Kai Han, Chao Xu, Hanting Chen, Xinghao Chen, and Yunhe Wang. From next-token to next-block: A principled adaptation path for diffusion llms. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2512.06776.

Chunqi Wang, Ji Zhang, and Haiqing Chen. Semi-autoregressive neural machine translation. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *EMNLP*, 2018.

Xinyun Wang, Min Zhang, Sen Cui, Zhikang Chen, Bo Jiang, Kun Kuang, and Mingbao Lin. Reversible diffusion decoding for diffusion language models. *CoRR*, 2026a. URL https://doi.org/10.48550/arXiv.2602.00150.

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, Kai Yu, and Zhijie Deng. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. *ICLR*, 2026b.

Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo O. Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dllm v2: Efficient block-diffusion LLM. *ICLR*, 2026a.

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. *ICLR*, 2026b.

Wei Xiong, Wenting Zhao, Weizhe Yuan, Olga Golovneva, Tong Zhang, Jason Weston, and Sainbayar Sukhbaatar. Stepwiser: Stepwise generative judges for wiser reasoning. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2508.19229.

Jingyi Yang, Yuxian Jiang, and Jing Shao. ρ-eos: Training-free bidirectional variable-length control for masked diffusion llms. *CoRR*, 2026. URL https://doi.org/10.48550/arXiv.2601.22527.

Yicun Yang, Cong Wang, Shaobo Wang, Zichen Wen, Biqing Qi, Hanlin Xu, and Linfeng Zhang. Diffusion LLM with native variable generation lengths: Let [EOS] lead the way. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2510.24605.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2508.15487.

Runpeng Yu, Qi Li, and Xinchao Wang. Discrete diffusion in large language and multimodal models: A survey. *TMLR*, 2026a.

Runpeng Yu, Xinyin Ma, and Xinchao Wang. Dimple: Discrete diffusion multimodal large language model with parallel decoding. *EMNLP*, 2026b.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *ACL*, 2019.

Haoyang Zheng, Xinyang Liu, Xiangrui Kong, Nan Jiang, Zheyuan Hu, Weijian Luo, Wei Deng, and Guang Lin. Ultra-fast language generation via discrete diffusion divergence instruct. *ICLR*, 2026.

Linhao Zhong, Linyu Wu, Bozhen Fang, Tianjian Feng, Chenchen Jing, Wen Wang, Jiaheng Zhang, Hao Chen, and Chunhua Shen. Beyond hard masks: Progressive token evolution for diffusion language models. *ACL*, 2026.

Zhanhui Zhou, Lingjie Chen, Hanghang Tong, and Dawn Song. dllm: Simple diffusion language modeling, 2026. URL https://arxiv.org/abs/2602.22661.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *CoRR*, 2025. URL https://doi.org/10.48550/arXiv.2505.19223.

APPENDIX

A LIMITATIONS

Architecture scope. The current study primarily focuses on discrete diffusion language models with **native** global bidirectional attention. Native bidirectional dLLMs remain an active and meaningful modelling direction rather than a superseded implementation variant (Goel et al., 2026; Yang et al., 2026; Ma et al., 2026b; Kocabay & Akkus, 2026).

Extending VSB to hybrid AR-diffusion pipelines, including Dream (Ye et al., 2025), LLaDA 2.1 (Bie et al., 2026), FastdLLM v2 (Wu et al., 2026a), SDAR (Cheng et al., 2026), NBDiff (Tian et al., 2025), and WeDLM (Liu et al., 2026a)-style methods, is therefore not the primary focus, as their AR proposal and diffusion refinement interfaces may change the interpretation of self-containedness and NF/FA probes. Nevertheless, we include experiments on the popular AR-initialised dLLM LLaDA 2.0 (Bie et al., 2025) to evaluate the generalisability of VSB beyond purely native architectures.

Operational self-containedness. VSB estimates self-containedness under the model's current decoding hypothesis. If that hypothesis is inaccurate, a block may appear stable relative to an incorrect continuation. The guarantee is therefore local but meaningful: VSB tests commitment readiness under the model state that will actually drive the next decoding step. The method should therefore be interpreted as a model-internal commitment criterion, not as a guarantee of correctness under all continuations.

Inference-time overhead. We acknowledge that current implementation of VSB introduces additional inference cost through boundary scoring. Nonetheless, we have proposed a **coarse-to-fine search to reduce this cost**, but it remains an approximate solver for the exact boundary objective. Thus, VSB should be interpreted as a **commitment-quality method** rather than a free speedup method. It improves the accuracy of blockwise dLLM decoding by avoiding future-dependent commitments, while coarse-to-fine search provides a practical accuracy-cost trade-off.

Moreover, efficient caching is less straightforward under globally bidirectional attention. Because VSB compares NF and FA conditionals under different context truncations, standard KV caching can limit cache reuse or alter the effective conditioning by freezing intermediate representations computed with future information. We therefore distinguish exact VSB from approximate acceleration variants. Our future-aware activation caching strategy reuses hidden states from a full-context forward while recomputing a small number of upper layers under masked attention, providing practical speedups while approximately preserving NF/FA semantics. Extending caching mechanisms for future-aware self-containedness testing remains an engineering problem for future work.

B Extended Discussion on Fixed Block Size Limits

Under the self-containedness principle, whether a block can be committed depends on how strongly its predictions rely on future context. As a result, the commit length that satisfies $S(p, b) \leq \varepsilon$ can vary substantially across prefixes: some regions admit long self-contained blocks, while others require shorter commits.

A fixed-size decoding strategy, however, commits a constant block size W regardless of content. Whenever this fixed length exceeds what self-containedness permits at a given prefix, the committed block violates $S(p, b) \leq \varepsilon$ and becomes future-dependent. Formally:

Proposition B.1: Characterisation of fixed-size self-containedness

Fix a block size $W > 0$. A fixed-size commitment strategy with block size W satisfies $S(p, b) \leq \varepsilon$ for all committed blocks if and only if: $S(p, p + W) \leq \varepsilon$ for every prefix position p at which a block of size W is committed.

Interpretation. Proposition B.1 shows why fixed-size commitment is restrictive under self-containedness: since the same W is applied everywhere, self-containedness must hold uniformly across all prefixes. When future dependence varies across the sequence, no single fixed block size can satisfy this condition consistently, motivating adaptive boundary selection within a fixed block budget $(p : q]$. The proof is in Appendix C.3.

C Proof of Technical Insights

C.1 Proof of Lemma 3.3

Proof. For each position $i \in (p, b]$, let P_i^{NF} and P_i^{FA} denote the predictive distributions under No-Future (NF) and Future-Aware (FA) conditioning. We seek to bound their average discrepancy within the block, by Pinsker's inequality:

$$\delta_{\mathrm{TV}}\big(P_i^{\mathrm{NF}}, P_i^{\mathrm{FA}}\big) \leq \sqrt{\tfrac{1}{2}\, D_{\mathrm{KL}}\big(P_i^{\mathrm{NF}} \| P_i^{\mathrm{FA}}\big)}.$$

Given that the square-root $\sqrt{\cdot}$ function is concave, applying the Jensen's inequality and averaging over positions $i = p+1, \ldots, b$, we have:

$$\frac{1}{b-p}\sum_{i=p+1}^{b}\delta_{\mathrm{TV}}\big(P_i^{\mathrm{NF}}, P_i^{\mathrm{FA}}\big) \leq \sqrt{\frac{1}{2(b-p)}\sum_{i=p+1}^{b} D_{\mathrm{KL}}\big(P_i^{\mathrm{NF}} \| P_i^{\mathrm{FA}}\big)}.$$

By definition of ε-self-containedness under token-level KL:

$$\frac{1}{b-p}\sum_{i=p+1}^{b} D_{\mathrm{KL}}\big(P_i^{\mathrm{NF}} \| P_i^{\mathrm{FA}}\big) \leq \varepsilon.$$

Substituting this bound gives:

$$\frac{1}{b-p}\sum_{i=p+1}^{b}\delta_{\mathrm{TV}}\big(P_i^{\mathrm{NF}}, P_i^{\mathrm{FA}}\big) \leq \sqrt{\tfrac{\varepsilon}{2}}.$$

Thus, when a block is ε-self-contained, revealing the hypothesised future changes the model's token-level predictive distributions within that block by at most $\sqrt{\varepsilon/2}$ on average. □

C.2 Proof of Proposition 3.4

Proof. Let $y^\star = y_i^{\mathrm{NF}}$ and let v be any token with $v \neq y^\star$. By the definition of total variation distance, for any token y,

$$\left|P_{\theta,i}^{\mathrm{NF}}(y) - P_{\theta,i}^{\mathrm{FA}}(y)\right| \leq \delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right).$$

Therefore,

$$P_{\theta,i}^{\mathrm{FA}}(y^\star) \geq P_{\theta,i}^{\mathrm{NF}}(y^\star) - \delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right),$$

and for any $v \neq y^\star$,

$$P_{\theta,i}^{\mathrm{FA}}(v) \leq P_{\theta,i}^{\mathrm{NF}}(v) + \delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right).$$

Hence,

$$\begin{aligned} P_{\theta,i}^{\mathrm{FA}}(y^\star) - P_{\theta,i}^{\mathrm{FA}}(v) &\geq P_{\theta,i}^{\mathrm{NF}}(y^\star) - P_{\theta,i}^{\mathrm{NF}}(v) \\ &\quad - 2\delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right). \end{aligned}$$

Since

$$P_{\theta,i}^{\mathrm{NF}}(y^\star) - P_{\theta,i}^{\mathrm{NF}}(v) \geq \gamma_i,$$

we have

$$P_{\theta,i}^{\mathrm{FA}}(y^\star) - P_{\theta,i}^{\mathrm{FA}}(v) \geq \gamma_i - 2\delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right).$$

If $\delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right) < \frac{\gamma_i}{2}$, then this difference is positive for every $v \neq y^\star$. Thus $y^\star$ remains the top-1 token under FA conditioning. □

C.3 Proof of Proposition B.1

Proof. Given a fixed-size block commitment with length $W > 0$. ($\Rightarrow$) If all committed blocks satisfy $S(p, b) \leq \varepsilon$, then in particular each committed block $(p, p+W]$ satisfies $S(p, p+W) \leq \varepsilon$.

($\Leftarrow$) If $S(p, p+W) \leq \varepsilon$ holds for every committed block $(p, p+W]$, then the strategy satisfies the condition for all committed blocks. □

C.4 Proof of Corollary 3.5

Proof. Let $\mathcal{F} = \{i \in (p : b] : U_i = 1\}$ be the unstable positions, with $K_i = D_{\mathrm{KL}}\left(P_{\theta,i}^{\mathrm{NF}} \middle\| P_{\theta,i}^{\mathrm{FA}}\right)$. Assume that for any $i \in \mathcal{F}$, the top-1 prediction changes. Since $\gamma_i \geq \gamma_{\min}$, the contrapositive of Proposition 3.4 would give:

$$\delta_{\mathrm{TV}}\left(P_{\theta,i}^{\mathrm{NF}}, P_{\theta,i}^{\mathrm{FA}}\right) \geq \frac{\gamma_{\min}}{2}.$$

By Pinsker's inequality, this implies $K_i \geq \gamma_{\min}^2/2$. Therefore, we have:

$$|\mathcal{F}| \cdot \frac{\gamma_{\min}^2}{2} \leq \sum_{i \in \mathcal{F}} K_i \leq \sum_{i=p+1}^{b} K_i = nS(p, b).$$

Since $|\mathcal{F}| = \sum_{i=p+1}^{b} U_i$, dividing by n gives:

$$\frac{1}{n} \sum_{i=p+1}^{b} U_i \leq \frac{2S(p, b)}{\gamma_{\min}^2}.$$

□

Method type	Signal used	Tests future dependence?
Fixed block	Position	No
Punctuation / syntax	Surface boundary	No
Confidence / entropy	Single-view uncertainty	No
Remasking	Post-commit revision	Indirectly
VSB	NF/FA counterfactual shift	Yes

Table 7: VSB differs from prior boundary rules by directly testing whether candidate committed tokens depend on hypothesised future context.

D Extended Related Work

Discrete diffusion language models. Diffusion-based language modeling presents a promising alternative to the left-to-right token generation with an iterative denoising process that predicts masked tokens under bidirectional attention (Li et al., 2025; Nie et al., 2025). This paradigm enables parallel token updates, arbitrary-order refinement, and natural support for infilling. Early work such as Diffusion-LM studied continuous or discrete diffusion for controllable text generation (Li et al., 2022; Austin et al., 2021a; He et al., 2023; Sahoo et al., 2024), motivating subsequent scaling of diffusion objectives to large Transformer backbones. Recent discrete dLLMs, notably LLaDA (Nie et al., 2025) and Dream (Ye et al., 2025), demonstrate that masked discrete diffusion training can reach competitive capability at the multi-billion scale compare to auto-regressive models (Liu et al., 2026a; Bie et al., 2025). A key practical issue is efficiency and length control. Standard diffusion decoding typically requires many denoising iterations and often assumes a fixed generation length. Several lines of work address these constraints (Yu et al., 2026a; Song et al., 2025; Liu et al., 2026b; Zhong et al., 2026). DAEDAL proposes a training-free strategy that expands length by inserting mask tokens when the current allocation appears insufficient, reducing wasted computation from overly long allocations while avoiding truncation from short ones (Li et al., 2026). dLLM-Var instead modifies training to predict `[EOS]` so that variable-length generation is native to the model, while still supporting blockwise diffusion-style inference (Yang et al., 2025). On acceleration, D2F converts a pretrained dLLM into an AR-diffusion hybrid through a teacher-student distillation process that distill the teacher full bidirectional attention predictions into a student model with blockwise causal attention. (Wang et al., 2026b), while DiDi-Instruct distills a pretrained masked dLLM into a faster student with substantially fewer denoising steps (Zheng et al., 2026). Our focus is orthogonal to these methods by target the *training-inference mismatch* created by combining bidirectional diffusion training with blockwise semi-AR commitment, and this is achieved by making block boundaries *variable* under our proposed *self-containedness* principle.

Blockwise semi-autoregressive decoding. In practice, many dLLMs decode using a blockwise semi-autoregressive procedure: tokens are denoised in parallel within a window, then a prefix block is committed and the window shifts forward (Arriola et al., 2025; Fu et al., 2026; Wu et al., 2026a). This pattern benefits from KV caching and provides a favorable trade-off between latency and quality, but it introduces an additional design choice: where to place boundaries (Wu et al., 2026b; Lu et al., 2026; Liu et al., 2026b). Historically, blockwise and semi-autoregressive decoding has been studied for autoregressive models and sequence-to-sequence generation like Blockwise parallel decoding and SAT (Stern et al., 2018; Wang et al., 2018). For dLLMs, the boundary choice interacts strongly with bidirectional attention: tokens inside a block can exploit within-block context during denoising, but committed tokens become immutable and must remain consistent with later context. Fixed-size blocks, punctuation heuristics, or other surface rules can therefore commit tokens that would have been revised once future context becomes visible, propagating inconsistencies across future blocks (Chen et al., 2026a; Sun et al., 2025). Our method formalises this risk through an explicit *self-containedness* commitment criterion (Eq. 7), and selects variable-size blocks by trading off commit length against self-containedness. This differs from the fixed-size blocks or heuristic boundaries approach, where our the goal is to ensure that committed content are self-contained under the semantics available at commitment time without needing future available context to be refined.

Blockwise training alignment. A central challenge is that conventional diffusion training and SFT mask tokens across the whole input response, whereas blockwise semi-AR inference commits blocks without seeing the future (Arriola et al., 2025). This discrepancy can yield a mismatch in conditioning semantics: training gradients may encourage token predictions that implicitly depend on future tokens that will not be available when the block is committed (Wu et al., 2026a; Sun et al., 2025; Chen et al., 2026a; Zhong et al., 2026). Blockwise SFT targets this mismatch by aligning fine-tuning with fixed-size blockwise decoding through training on a selected active block while masking all future tokens (Sun et al., 2025). This directly matches the inference-time information pattern for a chosen block size. While VSB shares the objective of aligning training to blockwise inference, it differs in how the blocks were principally chosen. Rather than assuming a fixed block length, we introduce an explicit self-containedness criterion that guides *variable-size* blocks. Concretely, we evaluate candidate boundaries by comparing predictions for the current decoding region under *future-aware* versus *no-future* conditioning. Training then applies supervision under the same no-future truncation semantics implied by the selected boundary, reducing mismatch not just for a single fixed block size but for a *content-dependent* boundary policy. In this sense, our method can be viewed as complementing Blockwise SFT, where we aim for an effective boundary selection driven by *self-containedness*.

Semantic-aware block generation. Recent work has challenged the fixed block size assumption and proposed semantic-aware methods to guide generation (Li et al., 2026; Yang et al., 2025; Kim et al., 2026; Xiong et al., 2025). AdaBlock-dLLM presents a training-free approach that uses confidence dynamics of semantic-associated tokens like punctuations during denoising to adjust block size and better align boundaries with local semantic structure (Lu et al., 2026). Related ideas also appear in multimodal discrete diffusion models, where Dimple provides a "confident decoding" strategy that unmasks tokens when confidence exceeds a threshold, reducing the number of iterations needed in practice (Yu et al., 2026b). While these methods adapt decoding based on confidence heuristics, they typically do not directly enforce that a committed block would remain resolved once future context is revealed.

VSB differs from confidence-, entropy-, and surface-boundary methods in what signal it measures. Confidence and entropy describe how certain the model is under a single conditioning view. In contrast, VSB asks a counterfactual question: would the model's predictions inside the candidate block change if the hypothesised future were hidden? This distinction is important because a token can be locally high-confidence while still depending on later context, such as an operator, delimiter, or partially completed reasoning step. Our method thus aligns with the broader motivation of semantic coherence, but provides a different mechanism (See Table 7). Instead of defining steps by prompts, punctuation, or confidence-only rules, we test self-containedness via predictive consistency with and without future context. This yields a principled boundary signal that can be used both at inference time for variable-size commitment and at training time for alignment under the same no-future semantics. VSB is therefore not a generic uncertainty score, but a future-dependence test for the commitment decision.

E ALGORITHMS

To facilitate understanding of VSB, several algorithms are presented. Algorithm 1 is the efficient coarse-to-fine search boundary selection. Algorithm 2 shows the full decoding loop. Algorithm 3 shows the *No-Future* alignment training process of VSB, which aligns the global bidirectional attention training with block-wise semi-AR inference conditioning.The boundary-selection step (C) is written as a scan over the candidate set for clarity.

Algorithm 1 VSB Coarse-to-Fine Boundary Selection

Input: Prefix end p, window end w, coarse stride s_c, fine stride s_f
1. Coarse search sparsely scans the window to locate a promising boundary region
Construct coarse candidate set $\mathcal{B}_c \leftarrow \{p+1+s_c, p+1+2s_c, \ldots, w\}$
for each $b \in \mathcal{B}_c$ **do**
    Compute self-containedness $S(p,b) \leftarrow$ Eq. 7
end for
Select coarse optimum $\tilde{b}$ from $\mathcal{B}_c$ by Eq. 11

2. Fine search refines the boundary choice around the best coarse candidate
Define fine neighborhood $\mathcal{B}_f \leftarrow [\tilde{b}-s_c, \tilde{b}+s_c] \cap (p, w]$ with stride s_f
for each $b \in \mathcal{B}_f$ **do**
    Compute self-containedness $S(p,b) \leftarrow$ Eq. 7
end for
Select final boundary $b^\star$ by Eq. 11
return $b^\star$

Algorithm 2 VSB Inference

Input: Prompt $x_{1:p}$, dLLM p_θ, denoising steps $t \in \{T, \ldots, 1\}$, block length W, hypothesis step t_{hyp}
Initialise $p \leftarrow \ell_{\text{prompt}}$
while $p < L$ **do**
    Set $q \leftarrow \min(p+W, L)$
    Initialise $(p, q]$ as masked
    Run diffusion from timestep $t = T$ to $t = 1$ to obtain $\hat{x}_{1:q}$ and store hypothesis $\hat{x}_{1:q}^{t_{\text{hyp}}}$
    Compute FA logits on $(p, q]$ using $\hat{x}_{1:q}^{t_{\text{hyp}}}$

    # *Boundary Selection*
    if $s_c \neq 1$ **then**
        Select $b^\star$ by Alg. 1
    else
        # *Dense search* $s_c = 1$
        for each $b \in \mathcal{B}(p, q)$ **do**
            Compute NF logits by truncation to b
            Compute $S(p,b) \leftarrow$ Eq. 7
        end for
        Select $b^\star$ by Eq. 11
    end if
    Commit $\hat{x}_{p+1:b^\star}$
    Update $p \leftarrow b^\star$
end while
return generated sequence

F EXTENDED DATASET DESCRIPTION

GSM8K (Cobbe et al., 2021) GSM8K is a benchmark of grade-school level math word problems created to test multi-step reasoning and arithmetic skills in language models. It contains a total of

Algorithm 3 VSB Training with *No-Future* Alignment

Input: Input sequences x, dLLM p_θ, diffusion steps $t \in \{1, \dots, T\}$, block length W
while not converged **do**
  Sample prefix end $p \sim \mathrm{Unif}\{1, \dots, L-1\}$
  Set window end $q \leftarrow \min(p + W, L)$
  Define candidate boundaries $\mathcal{B} = \{b \mid p < b \leq q\}$
  (A) Construct scoring hypotheses
  Set $\hat{x}_{1:p} \leftarrow x_{1:p}$
  Initialise $\hat{x}_{p+1:q} \leftarrow$ Masked
  Construct window hypotheses: $\hat{x}_{p+1:q} \leftarrow$ Eq. 12
  (B) Compute *Future-Aware* prediction
  $f_i^{\mathrm{FA}} \leftarrow p_\theta(\cdot \mid \mathrm{mask}_i(\hat{x}_{1:q}))$ for $i \in (p, q]$
  (C) Boundary scoring
  for each $b \in \mathcal{B}$ **do**
    $f_i^{\mathrm{NF}}(b) \leftarrow p_\theta(\cdot \mid \mathrm{mask}_i(\hat{x}_{1:b}))$ for $i \in (p, b]$
    $S(p, b) \leftarrow$ Eq. 7
  end for
  Select boundary $b^\star \leftarrow$ Eq. 11
  (D) No-Future diffusion training
  Corrupt tokens in $(p, b^\star]$ to obtain $\tilde{x}^t_{1:b^\star}$
  Compute *No-Future* loss: $\mathcal{L}_{\mathrm{NF}} \leftarrow$ Eq. 13
  Update parameters: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\mathrm{NF}}$
end while

approximately 8,500 problems with human-written solutions requiring elementary calculation steps to reach a numerical answer. The test split consists of 1,319 test examples. We evaluate under the 0-shot scenario.

MATH500 (Lightman et al., 2024) MATH-500 is a curated subset of 500 competition-style math problems drawn from the broader MATH benchmark (Hendrycks et al., 2021c) that was used in OpenAI's work (Lightman et al., 2024). Each problem is intended to challenge models' mathematical reasoning by requiring multi-step solution strategies that go beyond simple arithmetic. We evaluate under the 4-shot scenario.

MMLU GENERATIVE (Hendrycks et al., 2021b;a) MMLU generative is the generative adaptation of the Massive Multitask Language Understanding (MMLU) benchmark, which spans a wide range of academic subjects across STEM, humanities, and social sciences. The underlying benchmark consists of multiple-choice questions across 57 subjects. The generative is a generation variant where the model is asked to generate the answer letter. We evaluate under the 0-shot scenario.

HUMANEVAL (Chen et al., 2021) HumanEval is a code generation benchmark designed to evaluate Python function synthesis from docstrings. It contains 164 programming problems, each with a natural language "docstring" specification and accompanying unit tests. The test set consists of 164 questions, with correctness determined by executing generated code against the groundtruth. We evaluate under the 0-shot scenario.

MBPP (Austin et al., 2021b) MBPP (Mostly Basic Programming Problems) is a coding dataset of Python programming questions aimed at assessing fundamental coding abilities. The testset includes 500 short programming tasks. We use the instruction following variant of this dataset with 3-shot for evaluation.

GPQA-DIAMOND (Rein et al., 2023) GPQA-Diamond is a difficult subset of the Graduate-Level Google-Proof Q&A (GPQA) benchmark that focuses on expert-validated, high-difficulty multiple-choice science questions. The Diamond subset contains 198 questions, representing the hardest portion of the larger GPQA set. We evaluate under the 0-shot scenario.

HELLASWAG (Zellers et al., 2019) HellaSwag is a commonsense reasoning benchmark that asks models to select the most suitable followup given a narrative context. The dataset is structured for multiple-choice evaluation and contains 10,042 test examples. We evaluate under the 0-shot scenario.

G IMPLEMENTATION DETAILS

We use the `lm-evaluation-harness` framework (v0.4.10.dev0) [1] for an unified evaluation of publicly available datasets. Experiments were conducted using Python 3.12.11, PyTorch 2.7.0+gitf717b2a (ROCm compatible) and ROCm v6.4.3 with AMD Instinct MI300X and Python 3.9.19, PyTorch 2.3.1 with Cuda 12.2 on NVIDIA RTX A6000 with 48GB of memory. LLaDA-8B-Instruct (Nie et al., 2025) is used as the primary pretrained backbone for all training or post-hoc baselines. For VSB, we additionally report results on both LLaDA-1.5 (Zhu et al., 2025). Given that dataset-specific token budgets, block sizes, and few-shot settings various in prior work (Zhou et al., 2026; Nie et al., 2025; Lu et al., 2026; Arriola et al., 2025), we follow a consistent evaluation pipeline with reference to simple dLLM (Zhou et al., 2026) and evaluate all methods under the same configuration within our paper (Table 8). We measure a mix of 0/n-shot as it tests the decoding method with/withoutout in-context scaffolding. For the training-based VSB variant, we sweep the block length $W \in \{32, 64, 96\}$ with default $W = 96$. To ensure fair comparison, the block length W at inference time is matched to the block size used by the baselines. We use the medium step to construct the hypothesis (e.g., $t_{\text{hyp}} = \lfloor T/2 \rfloor$), or the last denoising pass when the window completes earlier. For models that require supervised fine-tuning, we use the training split of the `openai/gsm8k` dataset. We apply LoRA fine-tuning for training based methods under standard configuration presented in prior works (Sun et al., 2025; Hu et al., 2022; Biderman et al., 2024). VSB is trained with the AdamW optimiser (Loshchilov & Hutter, 2019) and cosine learning-rate schedule, using bf16 precision. Inconsistencies of results with the original reported statistics may stem from dataset configuration, software environment differences, and unavailable hyperparameters. We have reproduced the results to the best of our ability while maintaining a consistent evaluation configuration across the methods.

Dataset	Shots	Max generation length	Block size
GSM8K	0	512	64
MATH500	4	512	64
GPQA-Diamond	0	64	64
HumanEval	0	512	32
MBPP-Instruct	3	256	64
MMLU-Generative	0	3	3
HellaSwag-Gen	0	10	10

Table 8: Evaluation settings for each dataset.

H EXTENDED EXPERIMENTS

This section reports additional diagnostics that are not included in the main text due to space limit. They test four possible confounds: sensitivity to the hypothesis snapshot, sensitivity to sampling temperature, whether gains are explained by shorter average block lengths, and whether the proposed length-aware objective and search procedure are necessary.

H.1 TOKEN CLASSES ASSOCIATED WITH FUTURE DEPENDENCE

Tables 9 and 10 examine where large NF-FA disagreements arise. High-disagreement events concentrate heavily on delimiter or closure tokens, followed by arithmetic and code-syntax tokens. These are exactly the token classes where committing too early can freeze an unfinished expression, number, or program structure.

[1] https://github.com/EleutherAI/lm-evaluation-harness/tree/main

Token type	**Occurrence** (%)	**Mean probe KL**	**Hold-off tokens**
Delimiter/closure	76.3	0.2418	32.428
Numeral/arithmetic	10.8	**0.2715**	29.223
Identifier/code syntax	12.4	0.2550	**34.335**
Other	0.5	0.0382	5.133

Table 9: Breakdown of the top quartile of NF-FA disagreement events on MATH500 and HumanEval, grouped by token type at the probe position with the largest KL gap. Occurrence is the proportion of high-disagreement events in each class, and hold-off tokens is the average number of additional tokens delayed before commitment.

Token class	**Boundary change rate** (%)	**Correctness change** (%)	**Mean KL**
Delimiter/closure	41.3	**6.5**	**0.3550**
Numeral/arithmetic	35.7	4.8	0.1417
Identifier/code syntax	37.1	1.6	0.0339
Other	**42.2**	2.2	0.0606

Table 10: Token-class analysis of unstable boundaries caused by hypothesis updates. Blocks are grouped by the type of the next delayed token after the boundary chosen from the snapshot hypothesis. Boundary change measures how often rescoring with the final candidate changes the boundary; correctness change measures how often final answer correctness changes; mean KL is averaged over positions where the boundary changes.

H.2 Margin Diagnostics

Per-position margins are generally very large, with median $\gamma_i = 1.00$. This means that most individual predictions fall in the high-margin regime considered by Proposition 3.4. At the block level, however, $\gamma_{\min}$ is smaller because it is determined by the least confident position in the block. Its median is 0.41, and 20.5% of committed blocks have $\gamma_{\min} > 0.9$. Since even one low-margin token, often a delimiter, can reduce $\gamma_{\min}$ for the whole block, this block-level measure is conservative. Consistent with Corollary 3.5, the stability guarantee is strongest when $\gamma_{\min}$ is large and becomes weaker as $\gamma_{\min}$ decreases.

Moreover, we also compare the region committed by VSB with the remainder of the same window, which VSB defers but a fixed-size decoder would still commit (Table 11). The deferred region is clearly less stable: it has a $1.77\times$ higher flip rate, $1.81\times$ more low-margin positions, and $1.70\times$ higher mean TV. This suggests that VSB tends to stop committing before predictions become less stable. In contrast, fixed-size decoding continues to commit into this unstable region, whose predictions would later change when more context becomes available. The remaining flips are concentrated on delimiter and punctuation tokens (9.8% vs. 3.2-3.3% for numerals and other tokens), consistent with Appendix Tables 9 and 10.

	Flip rate (%)	**Low-margin positions** ($\gamma_i < 0.9$, **%**)	**Mean TV**
Committed by VSB ($p \to b^\star$)	5.3	5.2	0.061
Deferred by VSB ($b^\star \to q$)	9.4	9.4	0.104

Table 11: Stability of committed vs. deferred regions on GSM8K. Flip rate: fraction of positions whose top-1 prediction changes once future context arrives.

H.3 Computational Cost Analysis

VSB voundary search variants analysis. Table 12 reports runtime and decoding cost for exact and approximate search. Full VSB is the best performing but adds substantial scoring cost. Coarse-to-fine

and coarse-only variants reduce additional function evaluations and improve throughput, making them useful engineering variants when exact scoring is too expensive.

Method	Coarse	Fine	Runtime (s)	Tokens/s	Extra NFE	Avg. block len.	Acc (%)
Fixed-size block baseline	–	–	18.43	13.07	0.0	64.00	79.4
Dense scan (s_c=1)	1	–	56.11	5.73	342.0	38.51	86.0
Coarse-to-fine	16	4	36.58	6.58	63.0	32.03	85.6
Coarse-to-fine	16	8	30.25	7.91	61.4	34.74	85.4
Coarse-to-fine	32	8	32.17	7.26	64.5	27.99	84.0
Coarse-only	16	0	25.03	8.85	34.1	35.68	82.2

Table 12: Computational cost comparison of VSB variants on GSM8K subset (500 samples). Extra NFE is the average additional number of function evaluations introduced by boundary scoring beyond fixed-size decoding.

Budget-Controlled Ablation Table 13 checks whether VSB's gains are simply caused by shorter average blocks. For each VSB budget, the fixed-size LLaDA baseline is matched to VSB's observed average block length, as shown in Figure 6 (e.g., Given a budget of 32 tokens, VSB commits on average 24 tokens for GSM8K, we therefore compare the performance of baseline with fixed-size blocks of length 24).VSB remains strong over fixed-size decoding at lower and medium budgets, across both GSM8K and HumanEval. This directly supports the claim that the gains are not merely due to using effectively smaller blocks, but to choosing better boundaries.

Dataset	Matched Fixed block size (Original budget)	Fixed-size Baseline acc.	VSB (w/ *NF Tr. Align.*) acc.
GSM8K	24 (32)	75.13	**81.05**
GSM8K	36 (64)	75.89	**81.12**
GSM8K	60 (128)	**75.66**	75.20
HumanEval	22 (32)	35.98	**43.29**
HumanEval	30 (64)	37.20	**39.02**
HumanEval	37 (128)	37.20	**38.41**

Table 13: Budget-controlled comparison between VSB and fixed-size LLaDA. The fixed block size is matched to the average block length obtained by VSB under the corresponding VSB budget.

Compute-Matched Comparison. Exact NFE matching is difficult because NFE is an outcome of decoding (early stopping varies per sample), so we compare the nearest achievable operating points of fixed-size decoding and VSB. Together with the equal-average-block-length control (Table 13), Table 14 shows VSB's gains are not purely from more compute: at matched budgets the advantage persists, and with increased test-time compute VSB extends it.

	Fixed-size: Acc (NFE/sample)	VSB: Acc (NFE/sample)
GSM8K	77.0 (132)	**77.6** (128)
HumanEval	32.3 (138)	**33.5** (86)
MATH500	34.6 (223)	**36.0** (214)

Table 14: Accuracy at approximately matched forward-pass budgets. VSB matches or exceeds fixed-size decoding at equal or lower NFE.

I Case Study 1: VSB vs. Fixed-size block Baseline

We provide a case study in Figure 7 comparing the generation output and quality between our VSB and Fixed-size block decoding. Below is a detailed step by step analysis of the outputs with respect to a given question in GSM8K test set:

Question: According to its nutritional info, a bag of chips has 250 calories per serving. If a 300g bag has 5 servings, how many grams can you eat if your daily calorie target is 2000 and you have already consumed 1800 calories?

Problem structure and sensitivity to segmentation. The question involves a short chain of dependent steps. First, the total calories in the bag are computed from the number of servings and calories per serving. Next, the remaining daily calorie budget is obtained from the target and the calories already consumed. Finally, this remaining budget is converted into a fraction of the bag and then into grams using the bag weight. The full reasoning chain can be seen as:

$$5 \times 250 = 1250, \qquad 2000 - 1800 = 200,$$

$$\frac{200}{1250} \times 300 = 48,$$

where each intermediate result is needed by the next step. If numbers, units, or equations are broken across blocks, the error can easily propagate to the final answer. This makes the example highly sensitive to how decoding blocks are segmented.

VSB decoding behavior. With VSB decoding, block boundaries are chosen based on semantic and logical completeness. In this case, each block ends after a complete clause or a finished calculation. For instance, illustrated in top panel of Figure 7, the remaining calorie computation, $2000-1800 = 200$, appears entirely within a single block (Block 6). Likewise, the fraction: $\frac{200}{1250}$ and its simplified form: $\frac{4}{25}$ are kept intact before converting to grams. Because equations and explanations are not split mid-way, each block remains easy to interpret on its own. This allows the final conversion $\frac{4}{25} \times 300 = 48$ to be carried out correctly.

Numerical and unit consistency under VSB. VSB also preserves correct associations between numbers and their meanings. Calories per serving, total bag calories, remaining calories, and bag weight remain clearly separated. In particular, the 300 grams value is correctly treated as the weight of the whole bag, not a serving. This behavior is consistent with the low average self-containedness divergence observed for VSB,

$$\text{Avg } S(p, b^\star) = 0.055,$$

indicating that blocks can be understood without relying on future context.

Fixed-size decoding and fragmentation effects. In contrast, fixed-size decoding with a 64-token budget places boundaries without considering semantic structure. Shown in the bottom panel of Figure 7, blocks break in the middle of equations and sentences. One boundary interrupts the computation (Block 2-3):

$$5 \times 250 = 1250,$$

while another splits an explanatory sentence into incomplete parts (Block 4-5). These fragmented blocks are harder to interpret and depend on later tokens to recover meaning. This is reflected in a higher self-containedness divergence,

$$\text{Avg } S(p, b) = 0.25.$$

The fragmentation leads to a **clear error** later in the reasoning. The model incorrectly treats 300 grams as the weight of a serving rather than the whole bag. This unit confusion directly affects the grams conversion and produces an incorrect final answer.

J Case Study 2: Self-containedness Guided Block Selection

Figures 8 and 9 provide a full decoded-token trace showing how VSB repeatedly applies the self-contained boundary rule across an entire solution. At each prefix position p, VSB evaluates candidate cutoffs $b \in (p, p + W]$ (here with a fixed window budget $W = 64$), scores each candidate by self-containedness, and then selects the boundary $b^\star$ using a length-aware trade-off rather than the raw minimum of the divergence. Figures 8 provides an overview of case study 2's question and VSB's response annotated in blocks, as well as the overall self-containedness divergence across all blocks.

Overall pattern: boundaries align with semantic completion. Shown in Figure 9, across Blocks 0-20, the selected cutoffs repeatedly coincide with visually complete units in the decoded tokens. When the text forms a complete thought (e.g., finishing a sentence, closing an equation, completing an assignment like "a=1"), the divergence for candidates ending at that point is low, and $S_{\text{L-A}}(p, b)$ peaks there. When the text is mid-construction (e.g., an unfinished math environment or an incomplete formula), $S(p, b)$ rises for longer candidates, and the trade-off selects an earlier cutoff that avoids committing future-dependent content.

A key qualitative observation across the blocks in Figure 9 is that spikes in $S(p, b)$ coincide with tokens that clearly require continuation (unfinished equations, half-open LaTeX structures, or phrases that introduce content not yet stated). VSB responds by placing the boundary just before those unstable regions, so the next block can incorporate the missing future context before committing.

Where VSB looks strongest in this example.

1. **It separates stable prose from unstable math.** Early blocks show VSB committing short-to-medium spans that finish a statement, then opening the next block for the upcoming construction (Blocks 0-2). This matches the intended behavior: prose clauses are often self-contained, while upcoming math (definitions, substitutions, quadratic-formula expansions) creates future dependence until the expression is closed.
2. **It avoids committing incomplete equations.** Throughout Blocks 3-14, the decoded content alternates between explanatory text and symbolic expansions. Candidate boundaries that would cut inside a formula tend to have higher divergence, while candidates that end at a natural closure (end of a displayed equation, after the full numerator/denominator is formed, after "Thus," when the next line provides the result) yield lower divergence and therefore higher $S_{\text{L-A}}(p, b)$. This is precisely the behavior described in the main text: tokens beyond $b^\star$ are unstable because their meaning depends on the continuation from the next block.
3. **It remains consistent across the whole generation.** In Blocks 15-20, the same pattern continues: VSB commits a self-contained definition of "roots of unity", then postpones the claim about "6th roots" until the supporting context is in place. The final block includes the concluding sentence and answer, which is naturally self-contained and corresponds to low divergence near the end.

Failure mode analysis (constructively). The main weakness appeared with VSB in the trace is that **some blocks are still very short** (e.g., 3-9 tokens in several places). Importantly, these short blocks are not arbitrary: they occur at positions where the decoded tokens themselves are genuinely small self-contained units, such as

- local parameter assignments or short enumerations (e.g., "b=1", "c=1");
- LaTeX control-sequence glue where committing a longer span would cross into a future-dependent region (opening a fraction/sqrt that will only be completed by later tokens);
- short connective phrases that immediately precede a construction with high divergence (where longer candidates incur a large $(b - p)S(p, b)$ penalty).

So, even when a block is short, it is typically *still self-contained*: committing it does not freeze an interpretation that depends on unseen continuation.

That said, **short blocks do have a practical cost**: they reduce parallelism and therefore limit speedups in exactly the regions where the model is most "structure-sensitive" (dense math or heavy formatting). In this case study, the shortest blocks appear clustered around the most syntactically constrained parts of the solution (quadratic formula expansion, nested radicals/fractions, and equation delimiters). This suggests a principled explanation: when the generation enters a highly future-dependent micro-region, the length-aware objective prefers to commit only the stable prefix of that region, rather than forcing a longer but unstable commitment. This issue can be potentially resolved by defining a "minimum commit" value, nonetheless, we leave this as future practical tips instead of a fundamental analysis on the benefit of self-containedness.

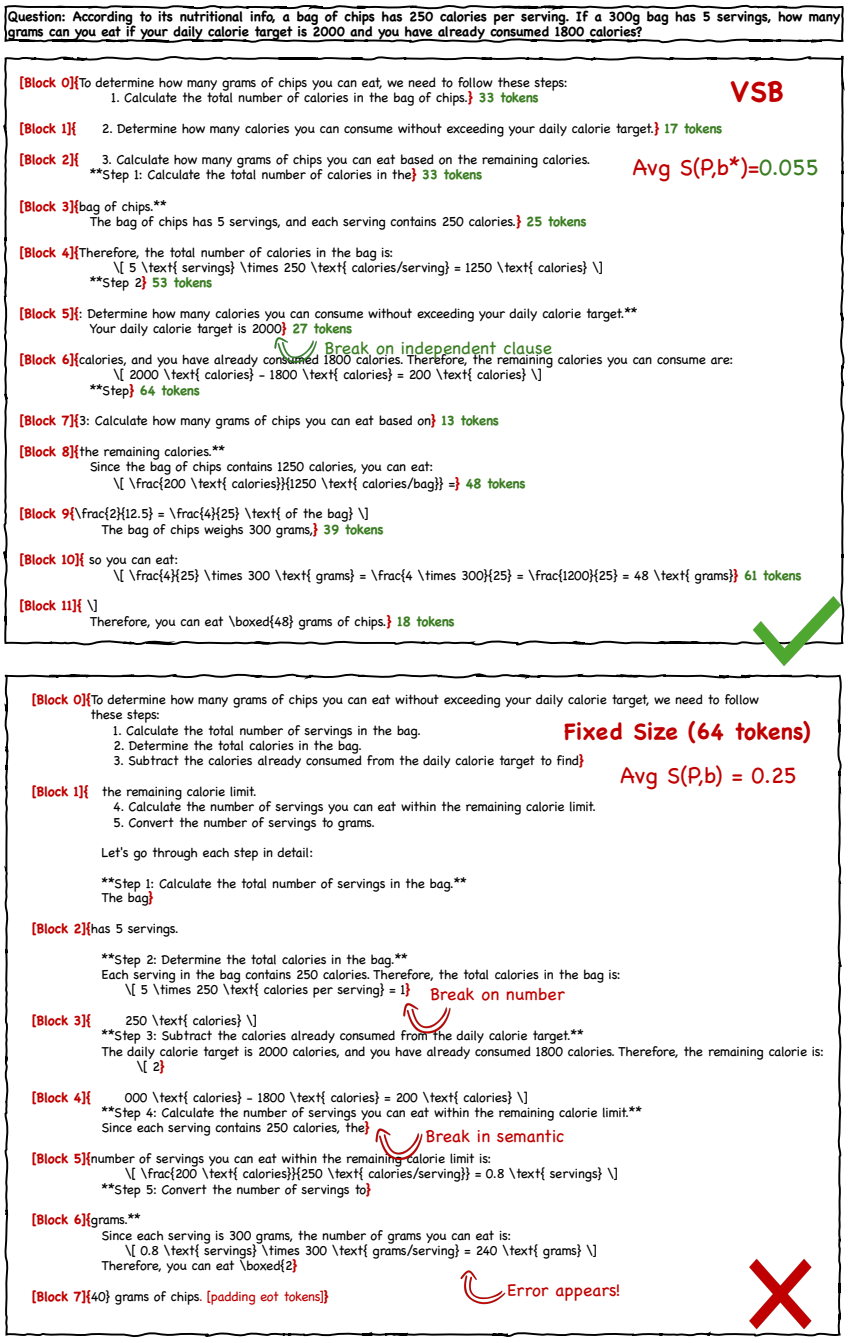
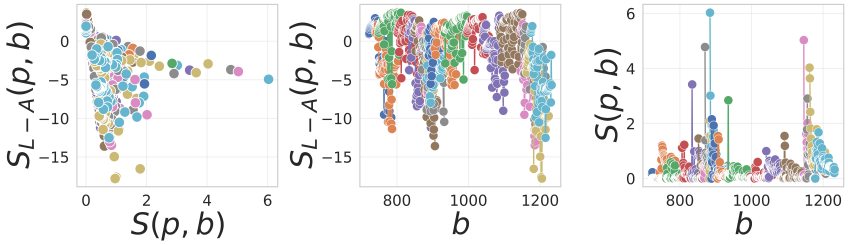


Figure 7: Case study of VSB vs. Fixed-size block baseline (64-tokens). VSB produces lower self-containedness divergence with more semantically coherent blocks at various sizes. Fixed-size blocks results in larger self-containedness divergence and blocks breaks on key number, semantics with errors in the final result.

Problem: What is the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n^{th} roots of unity?

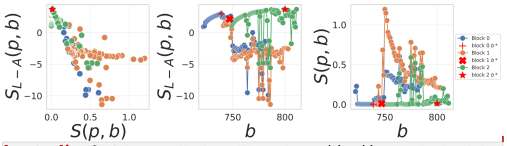
[Block 0]{To find the smallest positive integer \(n \) such that all the roots of \(}
[Block 1]{z^4 + z^2 + 1}
[Block 2]{ = 0 \) are \(n^{\text{th}} \) roots of unity, we start by solving the equation \(z^4 + z^2 + 1 = 0 \). }
First, we rewrite the equation in a more convenient form. }
[Block 3]{Let \(w = z^2 \). Then the equation becomes:
\[w^2 + w + 1 = 0. \] }
[Block 4]{This is a quadratic equation in \(w \). We can solve it using the quadratic formula }
[Block 5]{\(w = \frac{-b \pm}
[Block 6]{\sqrt{b^2 - 4ac}} }
[Block 7]{{2a} \), where \(a = 1 \), \(}
[Block 8]{b = 1}
[Block 9]{\), and \(}
[Block 10]{c = 1 \):
\[w = \frac{-1 \pm}
[Block 11]{\sqrt{1 - 4}}{2} = \frac{-1 \pm i\sqrt{3}}{2}. \]
Thus, }
[Block 12]{ the solutions for \(w \) are:
\[w = \frac{-1 + i\sqrt{3}}{2} \quad \text{and} \quad w = \frac{-1 - i\sqrt{3}}{2}. \]
Since \(w = z^2 \), we have:}
[Block 13]{\[z^2 = \frac{-1 + i\sqrt{3}}{2} \quad \text{or} \quad z^2 = \frac{-1 - i\sqrt{3}}{2}. \] }
[Block 14]{These are the roots of the original equation \(z^4 + z^2 + 1 = 0 \). We need to find the smallest positive integer \(n \) such that these roots are }
[Block 15]{\(n^{\text{th}} \) roots of unity.
The roots of unity are the solutions to the equation \(z^n = 1 \). The roots of unity are given by:
\[z = e^{2\pi i k / n} }
[Block 16]{\quad \text{for} \quad }
[Block 17]{k = 0, 1, 2, }
[Block 18]{\ldots, n-1. \]
We observe that the roots}
[Block 19]{of \(z^4 + }
[Block 20]{z^2 + 1 = 0 \) are exactly the 6th roots of unity. Therefore, the smallest positive integer \(n \) that satisfies the condition is \(n = 6 \).
The answer is: \(\boxed{6}\)}

(a) VSB response - all blocks

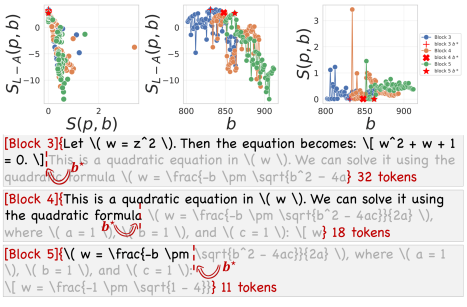


(b) Self-containedness across all blocks

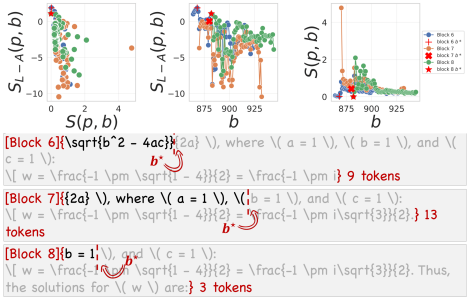
Figure 8: Case study 2 Overview



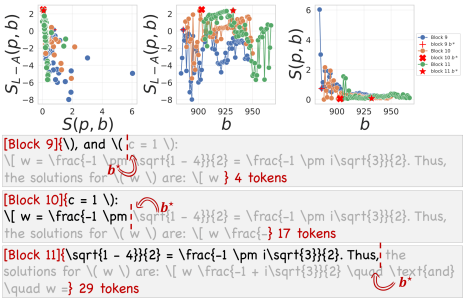
(a) Block 0-2



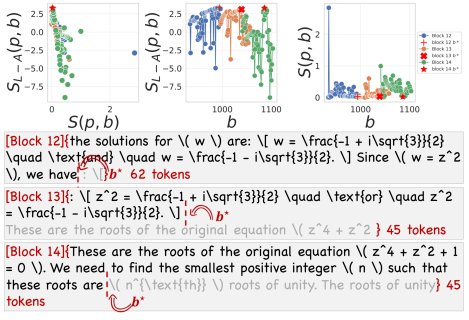
(b) Block 3-5



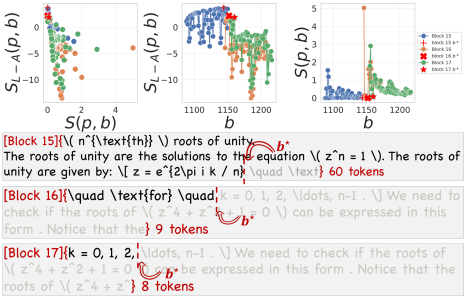
(c) Block 6-8



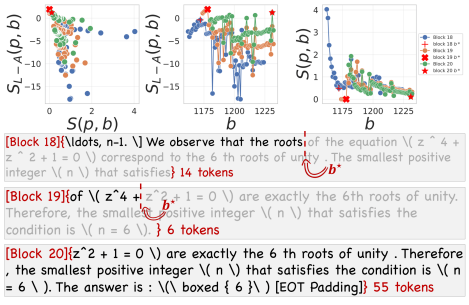
(d) Block 9-11



(e) Block 12-14



(f) Block 15-17



(g) Block 18-20

Figure 9: Self-containedness divergence and corresponding decoded tokens with self-contained boundary highlighted.