FINE-TUNING DIFFUSION LANGUAGE MODELS WITH CONTEXT SELECTION AND TARGET WEIGHTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised fine-tuning of discrete diffusion language models masks some response tokens and trains the model to recover their original values from the visible context. The masking pattern therefore determines both the context available to the model and the tokens it learns to predict. Uniform random masking does not explicitly account for the interaction between these choices. We introduce GoldiMask, which selects tokens to reveal as context by approximately maximizing a submodular objective. This objective uses model signals to balance the benefit of revealing tokens against their value as prediction targets. GoldiMask then weights the remaining targets according to how they benefit from the selected context and their remaining learning potential. Across three backbones and three training datasets, GoldiMask achieves the highest average accuracy in most evaluated settings, demonstrating gains on both reasoning and code generation. Component ablations show that both context selection and target weighting contribute to the gains. GoldiMask also reduces decoding iterations on GSM8K and MATH-500 under confidence-threshold parallel decoding, while maintaining comparable accuracy at higher confidence thresholds.

1 INTRODUCTION

Diffusion Large Language Models (DLLMs) have been recently proposed as an alternative to Autoregressive Large Language Models (ARLMs). ARLMs predict tokens one after the other, while DLLMs can predict multiple tokens in parallel, resulting in two advantages; parallel decoding (Chen et al., 2026a; Wu et al., 2026; Chen et al., 2026d), and revoking earlier decisions, which is especially valuable in domains like robotics (Liang et al., 2026).

As with ARLMs, recent works have explored post-training in DLLMs (Nie et al., 2026; Ye et al., 2025) in order to improve their performance and their reasoning capabilities. During supervised fine-tuning (SFT), some tokens are masked independently, and the model learns to reconstruct the full text. This masking process is generally uniformly at random. The mask therefore defines the learning problem: it determines both the evidence available to the model and the targets it must predict. Choosing useful context and choosing useful supervision are coupled decisions. A random split pairs the two blindly, which makes SFT in DLLMs more challenging than in ARLMs, and a poorly chosen masking scheme can even reduce the base model's accuracy (Parashar et al., 2026; Piskorz et al., 2026).

Recent methods adapt different aspects of diffusion SFT, including the masking rate (Xu et al., 2026), the choice of revealed tokens (Parashar et al., 2026), and the weighting of prediction losses (Ye et al., 2025; Deng et al., 2026). These methods use signals such as entropy, confidence, position, or attention to guide training. However, revealing a token changes both the context available to other targets and which tokens remain supervised, motivating a joint treatment of context selection and target weighting.

To address this coupling, we introduce GoldiMask, an SFT strategy that first selects which tokens to reveal and then determines how strongly to supervise the remaining targets. At a sampled masking ratio, GoldiMask scores a reveal set by the support it provides to the tokens left masked, weighted by their supervision priority. This priority favors intermediate gold-token probabilities (probabilities of correct tokens), balancing local probability responsiveness with remaining prediction headroom. Support is estimated from attention to the revealed positions, with diminishing returns as more

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.

context is supplied. This selection is interaction-aware, since the value of revealing a token depends on the context already selected and the supervision it gives up. We formulate this decision as a non-monotone submodular set-selection problem.

GoldiMask then evaluates the remaining targets with the selected context. It assigns larger loss weights to tokens whose gold-token probability increases and whose updated confidence still indicates useful supervision. Capping and normalizing these weights limits concentration while preserving their total mass. Together, selection and weighting allocate context and supervision within each training example.

We evaluate GoldiMask on reasoning and code-generation tasks across three backbones and three training datasets. GoldiMask achieves the highest average accuracy in most evaluated settings, with ablations supporting the contributions of adaptive context selection and target weighting.

GoldiMask's focus on learning from partial context also motivates evaluating parallel decoding. Predicting several tokens in one iteration requires the model to use the currently available context without waiting for the other predictions to be revealed. We therefore investigate whether coordinating context and supervision during training enables more tokens to be decoded per iteration while maintaining accuracy. On GSM8K and MATH-500, GoldiMask reduces decoding iterations, often maintaining comparable accuracy (Section 4.4).

Our contributions.

- We formulate masking in diffusion SFT as a coupled subset-selection problem and introduce GoldiMask, combining non-monotone submodular reveal selection with response-based, budget-preserving loss weighting.
- We evaluate GoldiMask across three backbones and three training datasets, achieving the highest average accuracy in most evaluated reasoning and code-generation settings. Ablations support the contributions of adaptive selection and target weighting and examine the choice of supervision-priority function.
- We evaluate parallel decoding across three backbones. Under confidence-threshold decoding, GoldiMask reduces decoding iterations on GSM8K and MATH-500, often maintaining comparable accuracy.

2 Related Work

In what follows, we review diffusion language models and recent approaches to their post-training. We refer the reader to Appendix A for additional related work.

Diffusion language models. Motivated by their success in continuous domains such as image generation (Ho et al., 2020; Dhariwal & Nichol, 2021), diffusion language models have appeared as an alternative paradigm to autoregressive language modeling. Diffusion language models operate either in continuous token-embedding spaces (Hu et al., 2026; Chen et al., 2026c) or directly over discrete tokens. Discrete diffusion language models have already shown competitive performance with similarly sized autoregressive models on tasks such as code generation and mathematical reasoning (Nie et al., 2026; Zhu et al., 2026; Ye et al., 2025). In this work, we focus on masked discrete diffusion language models, where the choice of which tokens are revealed is crucial for both inference and fine-tuning.

Post-Training. Vanilla diffusion SFT randomly masks response tokens and trains the model to reconstruct them from the visible context. Recent work adapts this process to make supervision more effective. LIFT (Parashar et al., 2026) selects targets by ground-truth confidence rank using a noise-dependent schedule; GIFT (Xu et al., 2026) assigns token-specific masking rates based on entropy; CART (Ye et al., 2025) weights targets by proximity to visible context; and AGDO (Deng et al., 2026) uses attention to guide denoising and loss weighting. GoldiMask explicitly accounts for the coupling between context and supervision: revealing a token supports other targets but removes that token from supervision. We capture this trade-off in a submodular set objective, then weight the remaining targets by their measured response to the selected context and their supervision priority.

3 METHODOLOGY

In this section, we present GoldiMask, a two-pass procedure for choosing which response tokens to reveal and how strongly to supervise those that remain masked. We first define the training setup and objective, then describe reveal-set selection and target weighting. Figure 1 illustrates the procedure. Algorithm 1 in Appendix B gives the pseudocode for one training step.

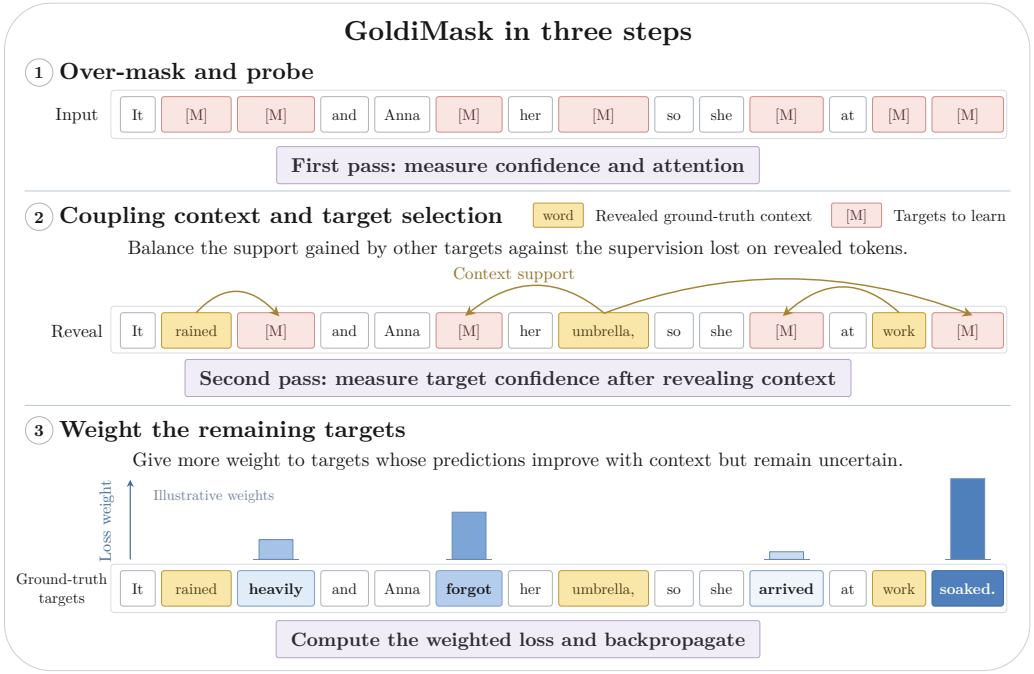

Figure 1: Overview of GoldiMask. A first forward pass on the over-masked response provides ground-truth token probabilities and attention scores for selecting a reveal set. A second forward pass evaluates the remaining targets with the selected context. Their positive context gains and post-reveal priorities determine loss weights through water-filling for the training update. The supervision-priority function λ and support response ϕ remain fixed throughout fine-tuning.

3.1 SETUP AND TRAINING OBJECTIVE

In supervised fine-tuning, each example is a prompt–response pair $x_0 = (q, y)$, where q denotes the prompt and y the ground-truth response. The prompt remains visible, and masking applies only to the response. Let $[L] = \{1, \dots, L\}$ index the L supervisable response positions, excluding padding and special tokens, and let y^i denote the token at position i. The noise level $t \sim \mathcal{U}(0, 1)$ specifies the desired final masking fraction.

We initially mask additional response tokens and then reveal a selected subset (Parashar et al., 2026). We draw $\rho \sim \mathcal{U}(0.1, 1 - t)$ for $t < 0.9$ and $\rho \sim \mathcal{U}(0, 1 - t)$ otherwise, applying the same procedure in both regimes. We independently replace each response token with `[MASK]` with probability $t+\rho$, producing a partially masked response $\widetilde{y}$ and input $\widetilde{x} = (q, \widetilde{y})$. Let $\mathcal{C} \subseteq [L]$ be the set of positions masked in this step. The desired supervised count and corresponding reveal budget are

$$K = \lfloor tL \rfloor, \qquad B = |\mathcal{C}| - K. \tag{1}$$

If $|\mathcal{C}| \leq K$, we reveal no tokens and apply uniformly weighted supervision to all masked candidates. For $1 \leq K < |\mathcal{C}|$, we select a subset $R \subseteq \mathcal{C}$ of size B, called the *reveal set*, and replace each mask at position $i \in R$ with its ground-truth token y^i. Denote the response after these replacements by y_t^R and the corresponding model input by $x_t^R = (q, y_t^R)$. The positions $\mathcal{S} = \mathcal{C} \setminus R$ remain masked, and their original tokens are the *supervised targets*. All visible tokens, including the prompt, the initially unmasked response tokens, and the revealed tokens, form the *context*.

Given the input x_t^R, we train the model to predict the ground-truth tokens at the masked positions $\mathcal{S}$. Let θ denote the model parameters and $p_\theta(y^i \mid x_t^R)$ the probability assigned to the ground-truth token y^i. GoldiMask assigns each supervised position a nonnegative weight w_i that controls its contribution to the per-example loss:

$$\ell_{\mathrm{GM}}(\theta) = -\frac{1}{tL} \sum_{i \in \mathcal{S}} w_i \log p_\theta(y^i \mid x_t^R). \tag{2}$$

Standard masked SFT uses uniform token weights, $w_i = 1$. GoldiMask uses adaptive weights, normalized so that

$$\sum_{i \in \mathcal{S}} w_i = |\mathcal{S}|. \tag{3}$$

Thus, the average weight across supervised positions remains one. The reveal set R and the weights w_i are held fixed during differentiation.

3.2 Selecting useful context

Selecting R balances the context supplied to other tokens against the value of retaining the revealed tokens as prediction targets. The first forward pass on $\widetilde{x}$, without gradient tracking, provides ground-truth token probabilities $p_i^{(1)} = p_\theta(y^i \mid \widetilde{x})$ for $i \in \mathcal{C}$ and attention scores A_{ij} from position i to j, averaged over heads in the final attention block. We set $A_{ii} = 0$ and hold these quantities fixed during selection.

For each remaining target $i \in \mathcal{C} \setminus R$, its total attention to the reveal set serves as a proxy for contextual support:

$$S_i(R) = \sum_{j \in R} A_{ij}. \tag{4}$$

We select R to maximize the priority-weighted support of the remaining targets:

$$F(R) = \sum_{i \in \mathcal{C} \setminus R} \lambda\Big(p_i^{(1)}\Big)\, \phi(S_i(R))\,, \qquad R^\star \in \arg\max_{R \subseteq \mathcal{C}:\, |R| = B} F(R). \tag{5}$$

Here, $\lambda(p_i^{(1)})$ scores the priority of retaining token i as a prediction target, while $\phi(S_i(R))$ estimates the benefit of its contextual support. Revealing a token can increase support for other targets but removes its own contribution from F.

For a ground-truth token probability $p \in [0, 1]$, we use the supervision priority $\lambda(p) = p(1-p)^3$, which peaks at $p = 1/4$. It combines the approximate probability increase $p(1-p)^2$ under an idealized cross-entropy update with the remaining probability gap $1 - p$, favoring targets with room to improve (Appendix H.1). Studies of supervision allocation and context response appear in Appendices H.3 and H.4, respectively.

For a nonnegative attention-support value s, we use $\phi(s) = s/(s + s_0)$, with $s_0 = 0.05$. This increasing, saturating response assigns less value to additional support for targets that already receive substantial context. Appendices H.2 and H.5 give its diminishing-returns properties and empirical support fits. Both functions remain fixed throughout training; their probability and attention inputs are recomputed for each example.

With fixed nonnegative priorities and attention weights, the nondecreasing, concave ϕ makes F submodular but generally non-monotone. The proof appears in Appendix C.2. We evaluate four submodular maximization algorithms, including practical adaptations of established methods. The main results report deterministic greedy (GoldiMask-D) and randomized greedy (GoldiMask-R) (Buchbinder et al., 2014). Appendix C.1 describes all four selectors and their implementation adaptations.

3.3 Weighting the remaining targets

After selecting R, a second forward pass on x_t^R provides updated ground-truth token probabilities $p_i^{(2)} = p_\theta(y^i \mid x_t^R)$ for $i \in \mathcal{S}$. We reuse λ to assess the remaining targets' priority under this context and combine it with their positive log-probability gains:

$$u_i = \big[\log p_i^{(2)} - \log p_i^{(1)}\big]_+ \lambda\Big(p_i^{(2)}\Big)\,, \qquad [a]_+ = \max(a, 0). \tag{6}$$

The first factor identifies targets helped by the reveal. The second favors targets that still merit supervision after receiving that help. Selection evaluates priority before the context has been chosen, whereas weighting evaluates it under the context used for training. Both probabilities are computed with the same model parameters, so the log-probability difference measures the response to the reveal before the training update.

We allocate the total weight $n = |\mathcal{S}|$ in proportion to u_i, subject to a per-token cap $c = 10$. Whenever the positive utilities can absorb the budget, water-filling gives

$$w_i = \min\Big(\frac{u_i}{\nu}, c\Big), \qquad \sum_{i \in \mathcal{S}} \min\Big(\frac{u_i}{\nu}, c\Big) = n. \tag{7}$$

The water-filling level ν is found by bisection in the nondegenerate case. The cap limits weight concentration while water-filling preserves the total loss-weight mass. Appendix G.1 gives the allocation's characterization and boundary cases. The weights are detached and act as fixed coefficients of the cross-entropy loss in the backward pass. Each GoldiMask intervention thus uses two forward passes and one backward pass.

4 Results and Analysis

In this section, we evaluate our approach through the following research questions.

- **RQ1.** Does GoldiMask improve reasoning accuracy over existing supervised fine-tuning methods? (Section 4.2)
- **RQ2.** Do these benefits extend to code generation? (Section 4.3)
- **RQ3.** How does GoldiMask affect the accuracy–decoding-step trade-off? (Section 4.4)
- **RQ4.** How do context selection, target weighting, and their design choices affect performance? (Section 4.5)

4.1 Experimental Setup

Models and training data. We use LLaDA-8B-Instruct (Nie et al., 2026), LLaDA-1.5 (Zhu et al., 2026), and Dream-7B (Ye et al., 2025). We fine-tune all three on s1K (Muennighoff et al., 2025), which contains 1,000 chain-of-thought examples. We additionally fine-tune LLaDA-8B-Instruct on LIFT-SFT-12K (Parashar et al., 2026) as a second reasoning corpus. For code generation, we fine-tune LLaDA-8B-Instruct and Dream-7B on KodCode (Xu et al., 2025) (Section 4.3).

Baselines. We compare against the base models, Vanilla SFT (Sahoo et al., 2024), and recent adaptive SFT methods including CART (Ye et al., 2025), GIFT (Xu et al., 2026), LIFT (Parashar et al., 2026), and AGDO (Deng et al., 2026). These adaptive methods modify masking or loss weighting (Section 2). All reported fine-tuned baselines are retrained in our harness with matched hyperparameters; each results table lists the methods evaluated in that setting.

Training and evaluation. We follow the protocol of Parashar et al. (2026), including their LoRA and optimizer settings. On s1K, training uses a LoRA rank of $r = 128$, a global batch size of 8, and $2{,}480$ steps. We evaluate pass@1 accuracy on GSM8K, MATH-500, Countdown, and Sudoku across three seeds. We use greedy decoding and a common checkpoint and generation-length selection protocol across compared methods (Appendix F). Fixed-length results are reported in Appendix D.1. On LLaDA-8B-Instruct, GoldiMask-D/R require approximately $1.2\times$ Vanilla SFT's training time, comparable to LIFT (Appendix F).

4.2 Reasoning Performance

We first compare GoldiMask's reasoning accuracy with existing supervised fine-tuning methods across the tested backbones and training corpora. On s1K, both GoldiMask-D and GoldiMask-R exceed the strongest SFT baseline in average accuracy across the four tasks on every backbone (Table 1). The strongest variant varies across models, but the average improvement is shared by the two maximizers.

Table 1: **Main results.** Fine-tuning on s1K across three diffusion language models. Values are means with standard deviations in smaller ± entries. Baselines are retrained in our harness with matched hyperparameters. *LIFT is the stronger of its H=2 and H=3 variants per setting, by four-benchmark average. Relative changes are against the strongest baseline in each block. † denotes our code implementation; 'w' denotes a workshop.

Method	Venue	GSM8K	MATH	Countdown	Sudoku	Average
LLaDA-8B-Instruct						
LLaDA-8B	–	78.1	36.1	19.6	11.2	36.3
Vanilla (Sahoo et al., 2024)	NeurIPS'24	79.2 ± 0.2	36.2 ± 0.9	17.2 ± 2.1	14.5 ± 1.3	36.8 ± 0.3
GIFT (Xu et al., 2026)	ICLRw'26	**81.0** ± 0.1	37.3 ± 0.4	21.5 ± 2.8	15.8 ± 0.4	38.9 ± 0.5
CART (Ye et al., 2025)	arXiv'25	80.4 ± 1.0	35.9 ± 1.6	24.0 ± 1.4	14.4 ± 1.6	38.7 ± 0.6
AGDO† (Deng et al., 2026)	ACL'26	78.3 ± 0.8	37.7 ± 0.1	16.9 ± 1.3	12.6 ± 0.7	36.4 ± 0.3
LIFT* (Parashar et al., 2026)	ICML'26	79.1 ± 0.8	36.6 ± 1.0	28.8 ± 2.6	13.5 ± 1.0	39.5 ± 0.4
GoldiMask-R	Ours	80.4 ± 0.2	38.3 ± 1.0	**29.7** ± 0.0	14.4 ± 1.5	40.7 ± 0.6
vs. LIFT		↑1.6%	↑4.6%	↑3.1%	↑6.7%	↑3.0%
GoldiMask-D	Ours	80.4 ± 1.2	**39.5** ± 0.8	27.7 ± 2.7	**16.9** ± 0.5	**41.1** ± 0.7
vs. LIFT		↑1.6%	↑7.9%	↓3.8%	↑25.2%	↑4.1%
LLaDA-1.5						
LLaDA-1.5	–	80.9	37.8	22.6	12.1	38.4
Vanilla	NeurIPS'24	79.2 ± 0.5	32.6 ± 1.6	22.0 ± 1.0	14.4 ± 1.6	37.1 ± 0.9
GIFT	ICLRw'26	80.4 ± 1.3	35.9 ± 0.1	19.7 ± 0.8	19.0 ± 0.4	38.8 ± 0.5
CART	arXiv'25	81.4 ± 0.8	37.9 ± 1.3	22.4 ± 2.2	17.7 ± 0.9	39.8 ± 0.1
AGDO†	ACL'26	80.0 ± 0.5	39.0 ± 0.6	21.5 ± 2.2	14.5 ± 0.6	38.7 ± 0.4
LIFT*	ICML'26	80.4 ± 1.0	37.7 ± 1.6	25.7 ± 1.5	16.1 ± 0.8	40.0 ± 0.4
GoldiMask-R	Ours	81.7 ± 1.0	**40.5** ± 2.7	**29.9** ± 3.0	17.9 ± 0.9	**42.5** ± 1.5
vs. LIFT		↑1.6%	↑7.4%	↑16.3%	↑11.2%	↑6.2%
GoldiMask-D	Ours	**81.9** ± 0.8	37.6 ± 0.2	26.6 ± 1.6	**21.6** ± 1.8	41.9 ± 0.8
vs. LIFT		↑1.9%	↓0.3%	↑3.5%	↑34.2%	↑4.7%
Dream-7B						
Dream-7B	–	76.7	39.8	21.1	8.2	36.5
Vanilla	NeurIPS'24	76.1 ± 0.4	30.6 ± 0.3	25.0 ± 0.9	14.8 ± 2.1	36.6 ± 0.5
GIFT	ICLRw'26	78.7 ± 0.7	40.2 ± 0.8	24.0 ± 1.4	22.7 ± 1.4	41.4 ± 0.0
CART	arXiv'25	77.8 ± 0.2	38.9 ± 0.6	22.3 ± 0.8	17.2 ± 2.7	39.1 ± 0.8
AGDO†	ACL'26	77.6 ± 0.4	40.3 ± 0.8	29.6 ± 1.2	10.8 ± 2.1	39.6 ± 0.9
LIFT*†	ICML'26	78.4 ± 0.1	42.1 ± 0.7	28.7 ± 0.8	23.3 ± 2.5	43.1 ± 0.7
GoldiMask-R	Ours	**79.6** ± 0.5	**44.3** ± 0.1	33.1 ± 3.6	**24.3** ± 0.2	**45.3** ± 1.0
vs. LIFT		↑1.5%	↑5.2%	↑15.3%	↑4.3%	↑5.1%
GoldiMask-D	Ours	78.8 ± 0.6	42.8 ± 0.4	**37.0** ± 4.0	22.8 ± 2.1	**45.3** ± 1.6
vs. LIFT		↑0.5%	↑1.7%	↑28.9%	↓2.1%	↑5.1%

Table 2: **A second corpus.** LLaDA-8B-Instruct fine-tuned on LIFT-SFT-12K at an equal step budget. Conventions and markers follow Table 1.

Name	GSM8K	MATH	Countdown	Sudoku	Average
LLaDA-8B	78.1	36.1	19.6	11.2	36.3
Vanilla	**82.9** ± 1.0	34.6 ± 0.4	19.9 ± 0.3	9.3 ± 0.7	36.7 ± 0.3
GIFT	79.8 ± 0.3	36.1 ± 0.1	23.6 ± 4.7	8.8 ± 1.2	37.1 ± 1.0
CART	80.2 ± 0.9	36.4 ± 2.8	22.4 ± 3.5	15.6 ± 1.2	38.6 ± 1.8
LIFT*	81.3 ± 0.7	36.6 ± 0.5	23.6 ± 1.5	11.3 ± 0.3	38.2 ± 0.4
AGDO†	82.3 ± 0.5	34.8 ± 0.4	19.4 ± 1.7	10.3 ± 0.9	36.7 ± 0.3
GoldiMask-R	82.0 ± 0.6	36.5 ± 0.7	23.1 ± 1.9	14.8 ± 0.1	39.1 ± 0.9
vs. CART	↑2.2%	↑0.3%	↑3.1%	↓5.1%	↑1.3%
GoldiMask-D	82.2 ± 0.4	**40.0** ± 2.0	**29.1** ± 4.7	**16.5** ± 0.2	**41.9** ± 0.5
vs. CART	↑2.5%	↑9.9%	↑29.9%	↑5.8%	↑8.5%

The additional LLaDA-8B-Instruct experiment on LIFT-SFT-12K shows the same pattern: both variants exceed the strongest baseline in average accuracy, with GoldiMask-D achieving the highest

average (Table 2). These results support the approach across the tested backbones and training corpora, while individual benchmark outcomes vary. Additional evaluation results are provided in Appendix D.

4.3 Extension to Code Generation

We next ask whether the benefits observed on reasoning tasks extend to code generation. We report results for LLaDA-8B-Instruct and Dream-7B fine-tuned on KodCode (Xu et al., 2025) and evaluate pass@1 on HumanEval (Chen et al., 2021) and the sanitized MBPP test split (Austin et al., 2021), using a maximum generation length of 256 tokens.

GoldiMask achieves the highest average accuracy across HumanEval and MBPP among the compared fine-tuned methods, with GoldiMask-D leading on LLaDA-8B and GoldiMask-R on Dream-7B (Table 3).

Table 3: **Code generation.** HE denotes HumanEval; MBPP uses the sanitized test split. Relative changes use each backbone's strongest fine-tuned baseline; other conventions follow Table 1.

	LLaDA-8B-Instruct			Dream-7B		
Name	**HE**	**MBPP**	**Avg.**	**HE**	**MBPP**	**Avg.**
Base model	34.1	41.3	37.7	39.6	52.5	46.1
Vanilla SFT	30.8 ± 1.3	41.6 ± 1.7	36.2 ± 0.2	45.4 ± 5.6	42.4 ± 1.7	43.9 ± 3.6
GIFT	35.4 ± 1.7	40.5 ± 2.8	38.0 ± 0.5	44.5 ± 1.7	46.5 ± 0.8	45.5 ± 0.4
CART	33.5 ± 1.7	39.9 ± 1.4	36.7 ± 0.2	47.5 ± 4.7	50.4 ± 0.3	49.0 ± 2.2
AGDO†	32.9 ± 0.9	40.1 ± 2.2	36.5 ± 0.7	41.8 ± 3.0	**53.7** ± 5.5	47.7 ± 4.3
LIFT	33.5 ± 0.0	39.5 ± 0.8	36.5 ± 0.4	42.4 ± 1.3	49.4 ± 1.7	45.9 ± 1.5
GoldiMask-R	34.4 ± 0.4	41.9 ± 4.4	38.2 ± 2.4	**49.1** ± 3.0	51.8 ± 0.6	**50.4** ± 1.2
vs. best baseline	↓ 2.8%	↑ 3.5%	↑ 0.5%	↑ 3.4%	↑ 2.8%	↑ 2.9%
GoldiMask-D	**39.0** ± 1.7	**44.6** ± 3.6	**41.8** ± 2.7	45.4 ± 5.6	49.6 ± 1.4	47.5 ± 2.1
vs. best baseline	↑ 10.2%	↑ 10.1%	↑ 10.0%	↓ 4.4%	↓ 1.6%	↓ 3.1%

4.4 Accuracy and Parallel-Decoding Cost

We compare GoldiMask-D/R/P with the adaptive SFT baselines using dParallel's confidence-threshold decoder (Chen et al., 2026d) across three backbones. Appendix E.2 gives the setup.

Figure 2 shows that GoldiMask consistently requires fewer decoding steps on GSM8K and MATH-500 across the tested backbones and thresholds, with comparable accuracy at higher thresholds. This pattern holds for three reveal-set maximizers, showing that the decoding benefit extends across GoldiMask variants. At lower thresholds, the reduction in steps comes with a larger accuracy trade-off, particularly on Dream-7B. Additional results on Countdown and Sudoku, with a discussion of their constraint-based structure and its implications for parallel decoding, appear in Appendix E.3.

Takeaway 1: Improving parallel decoding also involves how the model is trained. Our results suggest that adapting context and supervision during SFT can reduce decoding iterations at comparable accuracy, even when the decoding rule is unchanged.

4.5 Ablation Study

We examine the contributions of selection and weighting and their design choices on LLaDA-8B-Instruct fine-tuned on s1K.

Contributions of selection and weighting. We compare GoldiMask-D with random reveals, static initial-marginal selection, and uniform target weights. Static selection ranks candidates once by $\Delta(j \mid \varnothing) = F(\{j\}) - F(\varnothing)$ and reveals the top B without updating their scores. Iterative greedy instead recomputes marginal gains after each reveal, accounting for the changing context and remaining targets. All reveal-based variants use the same reveal budget and two-pass procedure; water-filling is retained except in the uniform-weight variant. Vanilla SFT provides a reference.

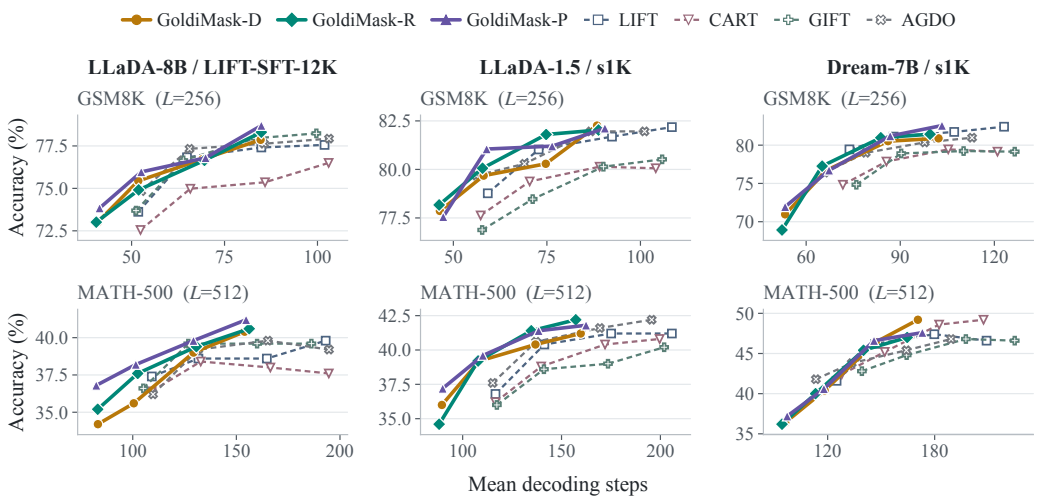

Figure 2: **Accuracy versus parallel-decoding steps.** GoldiMask-D/R/P (solid) and SFT baselines (dashed) on GSM8K and MATH-500. Markers show $\tau = 0.70, 0.80, 0.90, 0.95$ from left to right; lines connect measured settings. Upper-left is better. L denotes generation length.

Table 4: **Contributions of selection and weighting.** Accuracy on LLaDA-8B-Instruct fine-tuned on s1K. All reveal-based configurations use two passes. Bold marks the best result in each column.

Reveal selection	Target weighting	GSM8K	MATH	Countdown	Sudoku	Avg.
None (vanilla SFT)	Uniform	79.2	36.2	17.2	14.5	36.8
Random	Water-filling	80.1	37.5	24.6	14.5	39.2
Static initial-marginal	Water-filling	80.2	37.8	24.6	14.9	39.4
GoldiMask-D	Uniform	79.4	38.1	26.8	15.3	39.9
GoldiMask-D	Water-filling	**80.4**	**39.5**	**27.7**	**16.9**	**41.1**

GoldiMask-D achieves the highest average, 41.1 (Table 4), compared with 36.8 for Vanilla SFT. With water-filling held fixed, random reveals reach 39.2 and static selection reaches 39.4. Recomputing marginal gains therefore improves the average by 1.7 points over static ranking, supporting the benefit of adapting each reveal to earlier choices. With iterative selection held fixed, uniform weighting reaches 39.9, versus 41.1 with water-filling. Together, these comparisons support both adaptive selection and target weighting in this setting.

Choice of supervision priority. We compare $\lambda(p) = p(1-p)^3$ with $1-p$, $p(1-p)$, and $p(1-p)^2$ (Table 5). The three functions with an interior peak reach averages of 40.5–41.1, compared with 38.6 for $1-p$, which gives the highest priority to the least-confident targets. This comparison supports prioritizing targets at intermediate confidence. Our choice has the highest reported average, although the differences among the interior-peaked functions are small. Appendix H provides the theoretical and empirical motivation for the priority and contextual-support functions.

Table 5: **Supervision-priority ablation.** Accuracy on LLaDA-8B-Instruct fine-tuned on s1K, averaged over three seeds. Avg. is the mean across the four benchmarks. Bold marks the highest average.

Priority	GSM8K	MATH	Countdown	Sudoku	Avg.
$\lambda \propto 1-p$ (peak at 0)	79.2	38.6	20.7	15.9	38.6
$\lambda \propto p(1-p)$ (peak at $1/2$)	80.9	38.0	27.0	16.1	40.5
$\lambda \propto p(1-p)^2$ (peak at $1/3$)	81.0	38.4	26.3	16.9	40.7
$\lambda \propto p(1-p)^3$ (ours, peak at $1/4$)	80.4	39.5	27.7	16.9	**41.1**

Choice of loss weighting. Table 6 compares uniform weighting, DRO-tilted hardness, and utility-based weighting with caps. *Uniform weighting* assigns equal weight to every supervised token.

DRO-tilted hardness gives more weight to difficult training examples. Each example's loss is multiplied by $\exp(d/\tau)$, where d is its mean token difficulty and $\tau = 0.25$. A running-average normalizer keeps the multipliers approximately centered at one. This emphasis on difficult examples is inspired by distributionally robust optimization (DRO).

Uncapped utility weighting normalizes the measured token utilities to mean one without imposing a cap. For positive total utility, it assigns $w_i = nu_i / \sum_{j \in \mathcal{S}} u_j$. This preserves the total weight n while allowing a token with high utility to receive a large share of it.

Water-filling distributes a fixed total weight in proportion to measured token utility, subject to a per-token cap. It redistributes capped mass among targets below the cap. We compare the default $c = 10$ with $c = 5$ and $c = 15$.

Water-filling averages 41.1 with $c = 10$, 40.9 with $c = 5$, and 40.6 with $c = 15$. Uncapped utility weighting, DRO-tilted hardness, and uniform weighting reach 39.4, 40.6, and 39.9, respectively. Appendix G explains the allocation properties of water-filling and the role of the cap.

Table 6: **Loss-weighting comparison.** Accuracy on LLaDA-8B-Instruct fine-tuned on s1K, averaged over three seeds. Bold marks the highest average.

Weighting	GSM8K	MATH	Countdown	Sudoku	Avg.
Uniform	79.4	38.1	26.8	15.3	39.9
DRO-tilted hardness	80.6	39.4	24.6	17.9	40.6
Uncapped utility weighting	78.4	37.3	27.1	14.7	39.4
Water-filling ($c = 5$)	80.1	38.9	27.5	17.1	40.9
Water-filling ($c = 15$)	80.1	38.7	27.9	15.8	40.6
Water-filling ($c = 10$, ours)	80.4	39.5	27.7	16.9	**41.1**

Takeaway 2: Favoring targets at intermediate confidence matters more than precisely locating the priority peak. Similarly, limiting token weights matters more than the exact cap, with similar average accuracy for $c = 5, 10, 15$. The priority curve and cap can therefore be chosen flexibly within the tested range.

5 Conclusion

In this work, we introduced GoldiMask, an approach to supervised fine-tuning of discrete diffusion language models that jointly considers which tokens provide context and how strongly the remaining targets are supervised. By combining submodular context selection with response-based loss weighting, GoldiMask improves average benchmark accuracy across the evaluated backbones and training corpora. Its benefits also extend to parallel decoding, often yielding comparable accuracy with fewer decoding iterations. Together, these results highlight context selection and supervision weighting as complementary tools for improving diffusion SFT.

6 Limitations and Future Work

GoldiMask demonstrates the benefits of coordinating context selection and supervision weighting, and provides a starting point for improving both. Our theoretical analysis and empirical studies support the chosen functions, though they do not establish their optimality. The full method trades additional training cost for higher average accuracy.

A promising next step is to develop more direct estimates of how much each token benefits from supervision and how this depends on its context. Such estimates could guide both target selection and loss weighting. The parallel-decoding gains also motivate studying training and decoding together: beyond the confidence-threshold procedure evaluated here, future work could explore how supervision strategies support different ways of generating multiple tokens at once.

AI USE STATEMENT

In this work, we used generative AI tools to improve grammar, clarity, and phrasing in the manuscript and to assist with code development and the preparation of experimental figures. We have not used Generative AI tools to develop the method, construct the theoretical proofs, citation or literature review. The methodology and theoretical proofs were developed by the authors, with limited AI assistance to their presentation and consistency checking. AI-assisted code was verified and tested for correctness by two authors, and AI-assisted writing was reviewed by four authors. The authors take responsibility for the final content of this work, including its text, code, mathematical claims, and reported results.

ETHICS STATEMENT

This work studies supervised fine-tuning of diffusion language models using existing models and datasets. Our evaluation focuses on reasoning, code generation, and decoding efficiency, it does not assess safety, fairness, or potential misuse. Improvements on these benchmarks should therefore not be interpreted as evidence of safer model behavior.

REPRODUCIBILITY STATEMENT

We have documented our method and experimental procedures to support reproducibility. Section 3 presents GoldiMask's objective and weighting mechanism, with pseudocode in Appendix B and selector implementation details in Appendix C.1. Section 4.1 describes the models, training data, and experimental setup. Appendix F provides the checkpoint and generation-length selection protocol and computational resources, while Appendix E.2 details the parallel-decoding evaluation. We will release our code implementation upon acceptance to facilitate reproduction and future research.

REFERENCES

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

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

Giries Abu Ayoub, Mario Barbara, Lluís Pastor-Pérez, Tanja Bien, Aneesh Barthakur, Alaa Maalouf, and Loay Mualem. Supportive token revealing for fast diffusion language model decoding. *arXiv preprint arXiv:2606.04236*, 2026.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas S Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *Advances in Neural Information Processing Systems*, 38:55981–56007, 2026.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1433–1452. SIAM, 2014.

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=Oz335dV48X.

Kecheng Chen, Ziru Liu, Xijia Tao, Hui Liu, Yibing Liu, Xinyu Fu, Shi Wu, Suiyun Zhang, Dandan Tu, Lingpeng Kong, et al. Self-distilled trajectory-aware boltzmann modeling: Bridging the training-inference discrepancy in diffusion language models. *arXiv preprint arXiv:2605.11854*, 2026b.

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

Yixin Chen, Ankur Nath, Chunli Peng, and Alan Kuhnle. Discretely beyond $1/e$: Guided combinatorial algortihms for submodular maximization. *Advances in Neural Information Processing Systems*, 37:108929–108973, 2024.

Yuxin Chen and Andreas Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *International Conference on Machine Learning*, pp. 160–168. PMLR, 2013.

Yuxin Chen, Chumeng Liang, Hangke Sui, Ruihan Guo, Chaoran Cheng, Jiaxuan You, and Ge Liu. Langflow: Continuous diffusion rivals discrete in language modeling. In *ICML 2026 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2026c. URL https://openreview.net/forum?id=fQfnqdOAm9.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dparallel: Learnable parallel decoding for dLLMs. In *The Fourteenth International Conference on Learning Representations*, 2026d. URL https://openreview.net/forum?id=hVOcstAURb.

Jia Deng, Junyi Li, Xin Zhao, Jinpeng Wang, Hongyu Lu, and Ji-Rong Wen. Beyond fully random masking: Attention-guided denoising and optimization for diffusion language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44502–44514, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.2060. URL https://aclanthology.org/2026.acl-long.2060/.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Marwa El Halabi, Suraj Srinivas, and Simon Lacoste-Julien. Data-efficient structured pruning via submodular optimization. *Advances in Neural Information Processing Systems*, 35:36613–36626, 2022.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *International Conference on Machine Learning*, pp. 1311–1320. Pmlr, 2017.

Kai Han, Shuang Cui, Benwei Wu, et al. Deterministic approximation for submodular maximization over a matroid in nearly linear time. *Advances in neural information processing systems*, 33:430–441, 2020.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Keya Hu, Linlu Qiu, Yiyang Lu, Hanhong Zhao, Tianhong Li, Yoon Kim, Jacob Andreas, and Kaiming He. Elf: Embedded language flows. *arXiv preprint arXiv:2605.10938*, 2026.

Celeste Kidd, Steven T Piantadosi, and Richard N Aslin. The goldilocks effect: Human infants allocate attention to visual sequences that are neither too simple nor too complex. *PloS one*, 7(5): e36399, 2012.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. Klass: Kl-guided fast inference in masked diffusion models. *Advances in Neural Information Processing Systems*, 38:92267–92301, 2026.

Fanheng Kong, Jingyuan Zhang, Yahui Liu, Zirui Wu, Yu Tian, Guorui Zhou, et al. Accelerating diffusion llm inference via local determinism propagation. *arXiv preprint arXiv:2510.07081*, 2025.

M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International conference on machine learning*, pp. 19274–19286. PMLR, 2023.

Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Liuao Pei, Tian Nian, Shunbo Zhou, Xiaokang Yang, Jiangmiao Pang, Yao Mu, and Ping Luo. Discrete diffusion VLA: Bringing discrete diffusion to action decoding in vision-language-action policies. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=c3BVcHcSiR.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, yelong shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. Not all tokens are what you need for pretraining. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=0NMzBwqaAJ.

Lizhuo Luo, Zhuoran Shi, Jiajun Luo, Zhi Wang, Shen Ren, Wenya Wang, and Tianwei Zhang. Dawn: Dependency-aware fast inference for diffusion llms. *arXiv preprint arXiv:2602.06953*, 2026.

Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pp. 15630–15649. PMLR, 2022.

Loay Raed Mualem, Ethan R Elenberg, Moran Feldman, and Amin Karbasi. Submodular minimax optimization: Finding effective sets. In *International Conference on Artificial Intelligence and Statistics*, pp. 1081–1089. PMLR, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

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 *Advances in Neural Information Processing Systems*, volume 38, pp. 50608–50646, 2026.

Shubham Parashar, Atharv Chagi, Jacob Helwig, Lakshmi Jotsna Madhavarapu, Sushil Vemuri, James Caverlee, Dileep Kalathil, and Shuiwang Ji. Learnability-informed fine-tuning of diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=IgzMfT6eUx.

Julianna Piskorz, Cristina Pinneri, Alvaro Correia, Motasem Alfarra, Risheek Garrepalli, and Christos Louizos. Masks can be distracting: On context comprehension in diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=fNI8617Hfb.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Murad Tukan, Fares Fares, Yotam Grufinkle, Ido Talmor, Loay Mualem, Vladimir Braverman, and Dan Feldman. Orbslam3-enhanced autonomous toy drones: Pioneering indoor exploration. *arXiv preprint arXiv:2312.13385*, 2023.

Murad Tukan, Loay Mualem, and Moran Feldman. Practical 0.385-approximation for submodular maximization subject to a cardinality constraint. *Advances in Neural Information Processing Systems*, 37:51223–51253, 2024.

Robert C Wilson, Amitai Shenhav, Mark Straccia, and Jonathan D Cohen. The eighty five percent rule for optimal learning. *Nature communications*, 10(1):4646, 2019.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. In *International Conference on Learning Representations*, volume 2026, pp. 57027–57051, 2026.

Guowei Xu, Wenxin Xu, Zhao Jiawang, and Kaisheng Ma. GIFT: Guided importance-aware fine-tuning for diffusion language models. In *ICLR 2026 Workshop on Logical Reasoning of Large Language Models*, 2026. URL https://openreview.net/forum?id=WONIcfOYMP.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 6980–7008, 2025.

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

Lifeng Zhou and Pratap Tokekar. Risk-aware submodular optimization for multirobot coordination. *IEEE Transactions on Robotics*, 38(5):3064–3084, 2022.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11425–11460, 2026.

A Extended related work

See Section 2 for the directly comparable methods.

Trajectory- and order-based approaches. TABOM (Chen et al., 2026b) post-trains on the model's own decoding trajectories with an entropy-ranking loss. It starts, independently, from the same diagnosis as our reveal design: random masking during training does not match the easy-to-hard order of inference. The remedies differ. TABOM changes what the model trains on, while we change how the response is corrupted. Where-to-Unmask (Asano et al., 2026) learns an inference-time unmasking planner over a frozen token model, complementary to training-time corruption design.

Curriculum and learnability. Curriculum learning (Bengio et al., 2009), self-paced learning (Kumar et al., 2010), and learning-progress curricula (Graves et al., 2017) formalize the intuition that training is most effective at intermediate difficulty (Kidd et al., 2012; Wilson et al., 2019). Our function examines how a target's response to additional context varies with its initial confidence, providing empirical motivation for confidence-dependent context allocation. RHO-LOSS (Mindermann et al., 2022) and Rho-1 (Lin et al., 2024) select training points that are learnable and not yet learnt, at example and token granularity via reference models; our estimand is measured by within-token contrasts instead.

Submodular selection. Submodular functions model diminishing returns, where the marginal gain of adding an element decreases as the selected set grows. In our setting, revealing a token provides context to the remaining prediction targets, with diminishing benefit as their contextual support increases. Revealing that token also removes it from supervision. We capture this interaction through a submodular objective that sums concave functions of accumulated attention support over the remaining targets, weighted by their supervision priority. Because each reveal also removes a target's contribution, the objective is generally non-monotone. More broadly, submodular objectives are widely used in machine learning to select representative and non-redundant subsets, including data summarization Chen et al. (2024); Mualem et al. (2024), active learning Chen & Krause (2013), feature selection Tukan et al. (2024), model compression El Halabi et al. (2022), and sensor placement (Zhou & Tokekar, 2022; Tukan et al., 2023). For monotone submodular maximization under a cardinality constraint, greedy optimization admits the classical $(1 - 1/e)$ approximation guarantee (Nemhauser et al., 1978).

Parallel decoding. Autoregressive language models restrict each prediction to the left context during both training and inference, whereas the bidirectional attention of discrete diffusion language models offers a wider range of possibilities. Precisely because many tokens can be decoded at the same time, there is the promise of making inference faster. Methods that enhance faster parallel decoding include certainty-forcing distillation as in dParallel (Chen et al., 2026d), and training-free rules based on confidence (Fast-dLLM and DAWN), entropy (EB-Sampler), temporal stability (KLASS), or locality (LocalLeap) (Wu et al., 2026; Kim et al., 2026; Ben-Hamu et al., 2026; Kong et al., 2025). DAWN (Luo et al., 2026) combines confidence with attention-based dependencies, revealing tokens supported by previously revealed high-confidence anchors while avoiding coupled uncertain tokens. AXON (Ayoub et al., 2026) instead uses confidence to identify reliable anchors and attention to select those most useful to the remaining uncertain positions. Diffusion and autoregressive models can also be combined through speculative decoding (Leviathan et al., 2023): DFlash (Chen et al., 2026a) uses diffusion models to obtain many tokens in parallel before an autoregressive model verifies them.

B Main Algorithm

Algorithm 1 describes the exact-budget GoldiMask-D/R variants. We sample $\rho \sim \mathcal{U}(0.1, 1 - t)$ for $t < 0.9$ and $\rho \sim \mathcal{U}(0, 1 - t)$ otherwise, applying the over-mask-then-reveal procedure in both regimes. If $|\mathcal{C}| \leq K$, we reveal no tokens and apply uniformly weighted supervision to all masked candidates, retaining the loss normalization $1/(tL)$.

Algorithm 1 GoldiMask, one supervised fine-tuning step.

Require: response x of length L; model f_θ; learnability curve $\lambda(\cdot)$; support curve $\phi(\cdot)$; weight cap $c > 1$

$t \sim \mathcal{U}(0, 1)$
if $t < 0.9$ **then**
  $\rho \sim \mathcal{U}(0.1,\ 1 - t)$
else
  $\rho \sim \mathcal{U}(0,\ 1 - t)$
$\mathcal{C} \leftarrow$ positions masked by the forward process at rate $t + \rho$
$K \leftarrow \lfloor tL \rfloor, \quad B \leftarrow |\mathcal{C}| - K$
if $|\mathcal{C}| \le K$ **then**
  $p \leftarrow f_\theta(x \setminus \mathcal{C})$ ▷ no reveal; uniform weights
  return $-\dfrac{1}{tL} \sum_{i \in \mathcal{C}} \log p_i$
$p^{(1)}, A \leftarrow f_\theta(x \setminus \mathcal{C})$
$R \leftarrow \textsc{Select}(F, \mathcal{C}, B)$ ▷ any maximizer of submodular F s.t. $|R| = B$; App. C.1
$\mathcal{S} \leftarrow \mathcal{C} \setminus R$
$p^{(2)} \leftarrow f_\theta(x \setminus \mathcal{S})$
for $i \in \mathcal{S}$ **do**
  $u_i \leftarrow \left[\log p_i^{(2)} - \log p_i^{(1)}\right]_+ \cdot \lambda\big(p_i^{(2)}\big)$
$P \leftarrow \{i \in \mathcal{S} : u_i > 0\}, \quad n_+ \leftarrow |P|$ ▷ water-filling
if $n_+ = 0$ **then**
  $w_i \leftarrow 1$ for all $i \in \mathcal{S}$
else if $cn_+ \le K$ **then**
  $w_i \leftarrow c$ for $i \in P$
  $w_i \leftarrow (K - cn_+)/(K - n_+)$ for $i \in \mathcal{S} \setminus P$
else
  find $\nu > 0$ by bisection so that $\sum_{i \in P} \min(u_i/\nu,\ c) = K$
  $w_i \leftarrow \min(u_i/\nu,\ c)$ for all $i \in \mathcal{S}$
Detach the weights w_i from the computation graph.
return $-\dfrac{1}{tL} \sum_{i \in \mathcal{S}} w_i \log p_i^{(2)}$

where $F(R) = \displaystyle\sum_{i \in \mathcal{C} \setminus R} \lambda\big(p_i^{(1)}\big)\, \phi\big(S_i(R)\big), \qquad S_i(R) = \sum_{j \in R} A_{ij}$

Maximizing the reveal objective. The selection step in Algorithm 1 constructs R greedily on the marginals of F. Because F is submodular but non-monotone, the guarantee usually quoted for greedy selection does not apply here. The selector implementations and their budget conventions are given in Appendix C.1.

C Reveal-Set Selection

This section explains how to optimize the reveal objective and explores how the choice of the selection algorithm affects the corresponding downstream performance. Section C.1 introduces the four evaluated algorithms and their budget conventions. Section C.2 then develops the mathematical basis for selection. It proves that the objective is submodular, explains why it is generally non-monotone, and derives the exact marginal gain used to score each candidate reveal.

C.1 Maximizers for the reveal objective

The reveal objective $F(R) = \sum_{i \in \mathcal{C} \setminus R} \lambda(p_i)\, \phi(S_i(R))$ is submodular but not monotone (Proposition 1): revealing a token adds support to the targets that attend to it and, at the same time, removes that token's own term from the sum, so $F(\varnothing) = F(\mathcal{C}) = 0$ and the value can fall as the reveal set grows. We run GoldiMask with four maximizers with different determinism and budget characteris-

tics. Table 7 summarizes the four methods. Table 8 compares their performance across all reasoning settings against the strongest baseline in each setting.

Selection procedures. The four maximizers construct reveal sets as follows.

- **Greedy** adds the candidate with the largest marginal gain at each step (Equation 8), recomputing gains as the reveal set grows.
- **Randomized greedy** samples uniformly from the B highest-marginal remaining candidates, updating the pool after each selection.
- **TwinGreedy** grows two disjoint candidate sets. At each step, it chooses the feasible token–set pair with the largest marginal gain and adds that token to that set. It returns the set with the higher objective value.
- **Practical** 0.385 (P) retains the original algorithm's three-stage structure. We construct a local-search guide Z, run guided stochastic greedy while excluding Z during the first $\lceil 0.372B \rceil$ iterations, and return whichever solution has the higher objective value. Our implementation uses exact best-swap sweeps to construct the guide and samples uniformly from the B highest-marginal candidates in each greedy iteration, skipping negative-gain additions.

All four use the same marginal-gain oracle, priority and support functions. GoldiMask-D/R use an exact reveal budget, while GoldiMask-T/P may terminate early and retain additional supervised targets. All variants use mean-one water-filling over the remaining targets.

All arms on all settings. Table 8 places the four maximizers side by side on the four settings of Tables 1 and 2, against the strongest baseline of each setting.

C.2 Properties of the reveal objective

Let C be a finite set of candidate tokens. For every pair $i, j \in C$, let $A_{ij} \geq 0$ denote the attention support that revealing token j delivers to token i. We assume that A has zero diagonal and that the attention matrix is fixed throughout the construction of the reveal set. Likewise, let $\lambda_i \geq 0$ be the fixed learnability weight of token i.

For a reveal set $R \subseteq C$, define the total support delivered to token i by

$$S_i(R) \;=\; \sum_{j \in R} A_{ij}.$$

The GoldiMask objective is

$$F(R) \;=\; \sum_{i \in C \setminus R} \lambda_i \, \phi\big(S_i(R)\big), \qquad R \subseteq C.$$

The complement $C \setminus R$ appears because the tokens in R are revealed and therefore are not supervised. Hence, revealing a token can help the remaining supervised tokens while simultaneously removing its own contribution from the objective.

A function $F : 2^C \to \mathbb{R}$ is *submodular* if it satisfies the diminishing-returns property. For $R \subseteq C$ and $x \in C \setminus R$, define the marginal change from adding x to R as

$$\Delta(x \mid R) \;:=\; F(R \cup \{x\}) - F(R).$$

Table 7: **The four maximizers used for reveal-set selection.** The selection procedures below describe our implementations.

Maximizer	Deterministic	Reference
Greedy (D)	yes	Nemhauser et al. (1978), Math. Programming
Randomized greedy (R)	no	Buchbinder et al. (2014), SODA
TwinGreedy (T)	yes	Han et al. (2020), NeurIPS
Practical 0.385 (P)	no	Tukan et al. (2024), NeurIPS

Table 8: **All maximizers on all settings.** Same benchmarks, protocol and conventions as Tables 1 and 2. The baseline row is the strongest baseline of each setting by four-benchmark average; GoldiMask-D, GoldiMask-R and baseline rows are those of the main text. GoldiMask-T and -P are two-seed means using validation-selected checkpoints. The relative change beneath each GoldiMask row is measured against that baseline. Bold marks the best entry per column within a setting.

Name	GSM8K	MATH	Countdown	Sudoku	Average
LLaDA-8B-Instruct / s1K					
LIFT*	79.1	36.6	28.8	13.5	39.5
GoldiMask-D	80.4	**39.5**	27.7	16.9	**41.1**
vs. LIFT	↑1.6%	↑7.9%	↓3.8%	↑25.2%	↑4.1%
GoldiMask-R	80.4	38.3	**29.7**	14.4	40.7
vs. LIFT	↑1.6%	↑4.6%	↑3.1%	↑6.7%	↑3.0%
GoldiMask-T	**81.5**	38.7	26.8	15.1	40.5
vs. LIFT	↑3.0%	↑5.7%	↓6.9%	↑11.9%	↑2.5%
GoldiMask-P	81.0	36.5	28.8	**17.4**	40.9
vs. LIFT	↑2.4%	↓0.3%	0.0%	↑28.9%	↑3.5%
LLaDA-1.5 / s1K					
LIFT*	80.4	37.7	25.7	16.1	40.0
GoldiMask-D	81.9	37.6	26.6	**21.6**	41.9
vs. LIFT	↑1.9%	↓0.3%	↑3.5%	↑34.2%	↑4.7%
GoldiMask-R	81.7	40.5	29.9	17.9	**42.5**
vs. LIFT	↑1.6%	↑7.4%	↑16.3%	↑11.2%	↑6.2%
GoldiMask-T	**82.3**	40.5	26.6	17.6	41.7
vs. LIFT	↑2.4%	↑7.4%	↑3.5%	↑9.3%	↑4.2%
GoldiMask-P	81.7	**40.9**	**30.3**	16.7	42.4
vs. LIFT	↑1.6%	↑8.5%	↑17.9%	↑3.7%	↑6.0%
Dream-7B / s1K					
LIFT*	78.4	42.1	28.7	23.3	43.1
GoldiMask-D	78.8	42.8	37.0	22.8	45.3
vs. LIFT	↑0.5%	↑1.7%	↑28.9%	↓2.1%	↑5.1%
GoldiMask-R	**79.6**	**44.3**	33.1	**24.3**	45.3
vs. LIFT	↑1.5%	↑5.2%	↑15.3%	↑4.3%	↑5.1%
GoldiMask-T	78.7	43.7	**41.8**	24.2	**47.1**
vs. LIFT	↑0.4%	↑3.8%	↑45.6%	↑3.9%	↑9.3%
GoldiMask-P	78.7	44.0	40.8	21.1	46.2
vs. LIFT	↑0.4%	↑4.5%	↑42.2%	↓9.4%	↑7.2%
LLaDA-8B-Instruct / LIFT-SFT-12K					
CART	80.2	36.4	22.4	15.6	38.6
GoldiMask-D	**82.2**	**40.0**	29.1	**16.5**	**41.9**
vs. CART	↑2.5%	↑9.9%	↑29.9%	↑5.8%	↑8.5%
GoldiMask-R	82.0	36.5	23.1	14.8	39.1
vs. CART	↑2.2%	↑0.3%	↑3.1%	↓5.1%	↑1.3%
GoldiMask-T	81.7	39.0	**30.1**	14.0	41.2
vs. CART	↑1.9%	↑7.1%	↑34.4%	↓10.3%	↑6.7%
GoldiMask-P	81.0	37.1	29.1	14.1	40.3
vs. CART	↑1.0%	↑1.9%	↑29.9%	↓9.6%	↑4.4%

Then F is submodular if

$$\Delta(x \mid R) \;\geq\; \Delta(x \mid R') \qquad \text{for every } R \subseteq R' \subseteq C \text{ and } x \notin R'.$$

This means that once more elements have already been selected, adding x cannot become more valuable.

Proposition 1 (Submodularity). *Suppose that $\phi : [0, \infty) \to \mathbb{R}$ is concave and nondecreasing. Then*

$$F(R) = \sum_{i \in C \setminus R} \lambda_i \, \phi\left(\sum_{j \in R} A_{ij} \right)$$

is submodular on 2^C. If, in addition, $\phi(0) = 0$, then $F(\emptyset) = 0$ and $F(R) \geq 0$ for every $R \subseteq C$.

Proof. We prove the diminishing-returns inequality

$$\Delta(x \mid R) \;\geq\; \Delta(x \mid R')$$

for arbitrary sets $R \subseteq R' \subseteq C$ and an arbitrary $x \notin R'$.

Step 1: Diminishing increments of the concave response function.

For any fixed $a \geq 0$, define

$$d_a(s) \;:=\; \phi(s+a) - \phi(s), \qquad s \geq 0.$$

The quantity $d_a(s)$ is the benefit of adding a units of support when the current support level is s.

Because ϕ is concave, its increments decrease as the starting point increases. Therefore,

$$s \leq t \qquad \Longrightarrow \qquad d_a(s) \geq d_a(t).$$

Equivalently,

$$\phi(s+a) - \phi(s) \;\geq\; \phi(t+a) - \phi(t) \qquad \text{whenever } s \leq t.$$

Because ϕ is also nondecreasing and $a \geq 0$, we have

$$d_a(s) = \phi(s+a) - \phi(s) \geq 0.$$

Thus, for every fixed support increment $a \geq 0$, the function d_a is nonnegative and nonincreasing.

Step 2: Derivation of the exact marginal change.

Fix a set $R \subseteq C$ and an element $x \notin R$. For notational convenience, write

$$R_x \;:=\; R \cup \{x\}.$$

For every token i, adding x to the reveal set changes its support according to

$$S_i(R_x) = \sum_{j \in R_x} A_{ij} \quad = \sum_{j \in R} A_{ij} + A_{ix} = S_i(R) + A_{ix}.$$

By definition,

$$F(R_x) = \sum_{i \in C \setminus R_x} \lambda_i \, \phi\big(S_i(R_x)\big).$$

Since $x \notin R$, the objective before revealing x can be decomposed as

$$F(R) = \lambda_x \phi\big(S_x(R)\big) + \sum_{i \in C \setminus R_x} \lambda_i \, \phi\big(S_i(R)\big).$$

The first term is the contribution of x while it is still supervised. The remaining sum contains the tokens that stay supervised both before and after x is revealed.

Therefore,

$$\begin{aligned}
\Delta(x \mid R) &= F(R_x) - F(R) \\
&= \sum_{i \in C \setminus R_x} \lambda_i \, \phi\big(S_i(R_x)\big) - \left[\lambda_x \phi\big(S_x(R)\big) + \sum_{i \in C \setminus R_x} \lambda_i \, \phi\big(S_i(R)\big) \right] \\
&= \sum_{i \in C \setminus R_x} \lambda_i \left[\phi\big(S_i(R_x)\big) - \phi\big(S_i(R)\big) \right] - \lambda_x \phi\big(S_x(R)\big).
\end{aligned}$$

Using

$$S_i(R_x) = S_i(R) + A_{ix},$$

we obtain

$$\Delta(x \mid R) = \sum_{i \in C \setminus (R \cup \{x\})} \lambda_i \left[\phi\big(S_i(R) + A_{ix}\big) - \phi\big(S_i(R)\big)\right] - \lambda_x \phi\big(S_x(R)\big)$$
$$= \sum_{i \in C \setminus (R \cup \{x\})} \lambda_i \, d_{A_{ix}}\big(S_i(R)\big) - \lambda_x \phi\big(S_x(R)\big).$$

Hence the marginal has two components:

$$\boxed{\Delta(x \mid R) = \underbrace{\sum_{i \in C \setminus (R \cup \{x\})} \lambda_i \, d_{A_{ix}}\big(S_i(R)\big)}_{\text{support delivered to the remaining clients}} - \underbrace{\lambda_x \phi\big(S_x(R)\big)}_{\text{value forfeited by revealing } x}}. \tag{8}$$

Step 3: Comparison of the two marginals.

Now fix

$$R \subseteq R' \subseteq C, \qquad x \notin R'.$$

Define

$$I := C \setminus (R' \cup \{x\})$$

to be the tokens that remain supervised under both reveal sets, and define

$$D := R' \setminus R$$

to be the tokens that are supervised under R but have already been revealed under R'.

Because $x \notin R'$, the client set under the smaller reveal set decomposes as the disjoint union

$$C \setminus (R \cup \{x\}) = I \,\dot\cup\, D.$$

Using the marginal formula derived above, we have

$$\Delta(x \mid R) = \sum_{i \in I} \lambda_i d_{A_{ix}}\big(S_i(R)\big) + \sum_{i \in D} \lambda_i d_{A_{ix}}\big(S_i(R)\big) - \lambda_x \phi\big(S_x(R)\big),$$

whereas

$$\Delta(x \mid R') = \sum_{i \in I} \lambda_i d_{A_{ix}}\big(S_i(R')\big) - \lambda_x \phi\big(S_x(R')\big).$$

Subtracting the second expression from the first gives

$$\Delta(x \mid R) - \Delta(x \mid R') = \sum_{i \in I} \lambda_i \left[d_{A_{ix}}\big(S_i(R)\big) - d_{A_{ix}}\big(S_i(R')\big)\right]$$
$$+ \sum_{i \in D} \lambda_i d_{A_{ix}}\big(S_i(R)\big)$$
$$+ \lambda_x \left[\phi\big(S_x(R')\big) - \phi\big(S_x(R)\big)\right].$$

Equivalently, restoring the definitions of I and D,

$$\Delta(x \mid R) - \Delta(x \mid R') = \sum_{i \in C \setminus (R' \cup \{x\})} \lambda_i \left[d_{A_{ix}}\big(S_i(R)\big) - d_{A_{ix}}\big(S_i(R')\big)\right]$$
$$+ \sum_{i \in R' \setminus R} \lambda_i d_{A_{ix}}\big(S_i(R)\big)$$
$$+ \lambda_x \left[\phi\big(S_x(R')\big) - \phi\big(S_x(R)\big)\right].$$

We now show that each of these three terms is nonnegative.

First, since $R \subseteq R'$ and $A_{ij} \geq 0$,

$$S_i(R') = \sum_{j \in R'} A_{ij} \quad = \sum_{j \in R} A_{ij} + \sum_{j \in R' \setminus R} A_{ij} \geq \sum_{j \in R} A_{ij} = S_i(R).$$

Since $d_{A_{ix}}$ is nonincreasing,

$$d_{A_{ix}}\big(S_i(R)\big) \geq d_{A_{ix}}\big(S_i(R')\big).$$

Together with $\lambda_i \geq 0$, this proves that every summand in the first line is nonnegative:

$$\lambda_i \left[d_{A_{ix}}\big(S_i(R)\big) - d_{A_{ix}}\big(S_i(R')\big)\right] \geq 0.$$

Second, monotonicity of ϕ and $A_{ix} \geq 0$ imply

$$d_{A_{ix}}\big(S_i(R)\big) = \phi\big(S_i(R) + A_{ix}\big) - \phi\big(S_i(R)\big) \geq 0.$$

Therefore,

$$\sum_{i \in R' \setminus R} \lambda_i d_{A_{ix}}\big(S_i(R)\big) \geq 0.$$

This term accounts for clients that can benefit from revealing x under the smaller set R but no longer contribute under R' because they have already been revealed.

Third, the same support-ordering argument gives

$$S_x(R') \geq S_x(R).$$

Since ϕ is nondecreasing,

$$\phi\big(S_x(R')\big) - \phi\big(S_x(R)\big) \geq 0.$$

Because $\lambda_x \geq 0$,

$$\lambda_x \left[\phi\big(S_x(R')\big) - \phi\big(S_x(R)\big)\right] \geq 0.$$

This expresses the fact that revealing x forfeits at least as much client value under the larger reveal set, where x has already accumulated more support.

All three terms in

$$\Delta(x \mid R) - \Delta(x \mid R')$$

are therefore nonnegative. Consequently,

$$\Delta(x \mid R) - \Delta(x \mid R') \geq 0 \quad \Longrightarrow \quad \Delta(x \mid R) \geq \Delta(x \mid R').$$

Therefore F is submodular.

Finally, assuming $\phi(0) = 0$, and since ϕ is nondecreasing,

$$\phi(s) \geq \phi(0) = 0 \qquad \text{for every } s \geq 0,$$

and therefore $F(R) \geq 0$ for every $R \subseteq C$. □

For our support function, F is non-monotone whenever there exist $i \neq j$ with $\lambda_i A_{ij} > 0$. Since $\phi(s) > 0$ for $s > 0$,

$$F(\{j\}) \geq \lambda_i \phi(A_{ij}) > 0,$$

whereas $\phi(0) = 0$ gives $F(\emptyset) = 0$, and $F(C) = 0$ because no supervised clients remain. Thus F increases above zero at an intermediate reveal set and must decrease again before the full set is revealed.

Equation 8 gives the gain used by the selector. It also makes the cost of revealing a target explicit.

D Additional Results

D.1 Fixed task-specific generation lengths

Table 9 reports accuracy with generation lengths of 512 tokens for GSM8K and MATH-500 and 128 tokens for Countdown and Sudoku, shared across methods. Checkpoints are selected by validation loss. These comparisons complement the best-over-length results in Tables 1 and 2.

Table 9: **Fixed task-specific generation lengths.** Accuracy (%) at GSM8K/MATH-500/Countdown/Sudoku lengths of 512/512/128/128. Bold marks the highest displayed mean in each column within a panel.

Method	GSM8K	MATH	Countdown	Sudoku	Average
LLaDA-8B / s1K					
GIFT	81.0 ± 0.1	37.3 ± 0.4	21.5 ± 2.8	15.8 ± 0.4	38.9 ± 0.5
CART	80.4 ± 1.0	35.6 ± 2.0	23.8 ± 1.7	14.4 ± 1.6	38.6 ± 0.6
AGDO	78.3 ± 0.8	37.2 ± 0.7	15.5 ± 1.3	12.4 ± 0.6	35.8 ± 0.5
LIFT	79.1 ± 0.8	35.9 ± 1.0	24.9 ± 3.0	13.5 ± 1.0	38.4 ± 0.7
GoldiMask-D	80.4 ± 1.2	**39.5** ± 0.8	21.9 ± 2.0	16.3 ± 1.0	**39.5** ± 0.8
GoldiMask-R	80.4 ± 0.2	38.3 ± 1.0	**25.4** ± 0.0	13.4 ± 1.5	39.4 ± 0.6
GoldiMask-T	**81.5** ± 0.4	38.7 ± 1.3	21.3 ± 3.0	15.1 ± 0.1	39.2 ± 0.3
GoldiMask-P	81.0 ± 1.3	36.5 ± 0.7	22.5 ± 2.5	**17.4** ± 1.3	39.4 ± 0.8
LLaDA-1.5 / s1K					
GIFT	80.4 ± 1.3	35.9 ± 0.1	17.4 ± 3.0	19.0 ± 0.4	38.2 ± 1.0
CART	81.1 ± 1.1	37.6 ± 1.8	20.0 ± 2.3	15.6 ± 0.6	38.6 ± 0.3
AGDO	79.8 ± 0.8	38.8 ± 0.5	20.7 ± 1.2	14.3 ± 0.5	38.4 ± 0.4
LIFT	80.3 ± 1.1	37.5 ± 1.9	21.0 ± 3.7	16.1 ± 0.8	38.7 ± 1.1
GoldiMask-D	81.8 ± 0.8	37.6 ± 0.2	20.0 ± 4.4	**21.6** ± 1.8	40.3 ± 0.8
GoldiMask-R	81.3 ± 1.0	40.5 ± 2.7	22.9 ± 3.0	17.9 ± 0.9	40.7 ± 1.5
GoldiMask-T	**82.3** ± 1.4	40.5 ± 1.0	**23.6** ± 1.9	17.6 ± 0.7	**41.0** ± 0.4
GoldiMask-P	81.7 ± 0.6	**40.9** ± 1.6	22.1 ± 4.7	16.7 ± 0.7	40.4 ± 1.5
Dream-7B / s1K					
GIFT	77.3 ± 0.9	40.2 ± 0.8	23.8 ± 1.1	22.7 ± 1.4	41.0 ± 0.2
CART	77.8 ± 0.2	38.9 ± 0.6	22.3 ± 0.8	17.2 ± 2.7	39.1 ± 0.8
AGDO	77.6 ± 0.4	40.3 ± 0.8	29.1 ± 1.7	10.8 ± 2.1	39.4 ± 1.0
LIFT	78.4 ± 0.1	42.1 ± 0.7	28.7 ± 0.8	23.3 ± 2.5	43.1 ± 0.7
GoldiMask-D	77.9 ± 0.9	42.8 ± 0.4	35.9 ± 2.2	20.3 ± 1.6	44.2 ± 0.9
GoldiMask-R	**78.7** ± 0.5	**44.3** ± 0.1	30.1 ± 3.6	**24.3** ± 0.2	44.4 ± 1.0
GoldiMask-T	78.3 ± 0.5	43.5 ± 0.1	**41.8** ± 1.7	24.2 ± 0.1	**47.0** ± 0.5
GoldiMask-P	**78.7** ± 1.4	44.0 ± 0.8	40.8 ± 6.4	21.1 ± 0.3	46.2 ± 1.7
LLaDA-8B / LIFT-SFT-12K					
GIFT	79.8 ± 0.3	36.1 ± 0.1	23.6 ± 4.7	8.2 ± 2.1	36.9 ± 0.8
CART	80.2 ± 0.9	36.4 ± 2.8	18.2 ± 3.6	15.6 ± 1.2	37.6 ± 1.7
AGDO	**82.3** ± 0.5	34.7 ± 0.3	17.8 ± 0.2	10.3 ± 0.9	36.3 ± 0.1
LIFT	81.3 ± 0.7	36.6 ± 0.5	19.3 ± 3.5	11.3 ± 0.3	37.1 ± 1.0
GoldiMask-D	82.2 ± 0.4	**40.0** ± 2.0	23.0 ± 2.8	**16.5** ± 0.2	40.4 ± 0.2
GoldiMask-R	82.0 ± 0.6	36.5 ± 0.7	17.8 ± 1.9	14.4 ± 0.1	37.7 ± 0.9
GoldiMask-T	81.7 ± 0.3	39.0 ± 2.0	**29.9** ± 3.0	14.0 ± 0.7	**41.2** ± 0.6
GoldiMask-P	81.0 ± 0.4	37.1 ± 0.1	25.4 ± 12.2	14.1 ± 1.4	39.4 ± 3.2

Table 10: **AIME24 pass@16 after fine-tuning on s1K.** Values are mean accuracy (%) ± sample standard deviation across three seeds. Bold marks the highest mean in each column.

Method	**LLaDA-8B**	**LLaDA-1.5**	**Dream-7B**
Vanilla	6.7 ± 0.0	7.8 ± 1.9	6.7 ± 3.3
GIFT	8.9 ± 3.8	8.9 ± 1.9	8.9 ± 5.1
CART	**12.2** ± 3.8	**14.4** ± 5.1	11.1 ± 5.1
LIFT	**12.2** ± 5.1	12.2 ± 1.9	8.9 ± 3.8
GoldiMask-D	10.0 ± 3.3	12.2 ± 1.9	**13.3** ± 6.7

D.2 AIME24 PERFORMANCE ACROSS SEEDS

We additionally evaluate models fine-tuned on s1K on AIME24 using pass@16, reporting means and sample standard deviations across three seeds in Table 10. With only 30 problems, one additional solved problem changes the score by approximately 3.3 percentage points. Several methods show substantial variation across seeds, and the strongest method differs across backbones. GoldiMask-D achieves the highest mean on Dream-7B, matches LIFT on LLaDA-1.5, and slightly trails CART and LIFT on LLaDA-8B.

E PARALLEL-DECODING BEHAVIOR

This section examines the parallel-decoding behavior observed after GoldiMask fine-tuning. We describe the decoding process, evaluation setup, and results. GoldiMask itself operates during training; the analysis here concerns the behavior of the resulting model at inference time.

E.1 DECODING PROCESS

A masked diffusion decoder repeatedly predicts unresolved response positions from a partially completed sequence. Several positions can be filled in the same iteration because their predictions use the same current context. In a confidence-threshold decoder, a position is eligible for commitment when the probability of its most likely token reaches a threshold τ. Committed tokens then provide context for subsequent predictions. The choice of threshold trades the size of the proposed batch against the reliability of its predictions; any fallback or revision rule is also part of the decoder.

Confidence is observable during generation, but correctness is not. In reference-based diagnostics we can additionally count predictions that both match the reference token and satisfy the threshold. We call this quantity *oracle-safe parallelism*. It is a diagnostic of token recovery at a fixed state, distinct from task-level correctness of freely generated text.

E.2 EVALUATION SETUP

Models, tasks, and decoder. We use the same backbones and training corpora as Section 4.2, with the checkpoints used in the original parallel-decoding evaluation: LLaDA-8B-Instruct trained on LIFT-SFT-12K, and LLaDA-1.5 and Dream-7B trained on s1K, for GoldiMask-D, -R and -P and for the retrained baselines LIFT, CART, GIFT and AGDO. Benchmarks are GSM8K, MATH-500, Countdown and Sudoku, with maximum generation length $L = 512$ for MATH-500, $L = 256$ for GSM8K, and $L = 128$ for Countdown and Sudoku. Following dParallel (Chen et al., 2026d), we use greedy confidence-threshold decoding in 32-token blocks processed from left to right. Within each block, we permanently commit masked-token predictions whose confidence reaches τ, falling back to the most confident prediction if none qualifies. All methods decode the full prescribed length without EOS-based early stopping. We sweep $\tau \in \{0.70, 0.80, 0.90, 0.95\}$ and report, per setting and threshold, task accuracy and the mean number of decoding iterations per example.

End-to-end decoding. The reported measurements are end-to-end: matched prompts and decoder settings, with output quality reported together with total decoding iterations. Early commitments change later context, so these trajectories are not comparable state by state. Any wall-clock claim would additionally require matched hardware and timing; a reduction in iteration count is not a measured latency speedup.

Shared-state diagnostics. A complementary protocol holds partially completed reference sequences fixed across models and measures correct eligible predictions, threshold precision, and changes in correct eligibility. This separates prediction behavior from visited states; these diagnostics are not reported below.

E.3 RESULTS

Figures 2 and 3 cover GSM8K/MATH-500 and Countdown/Sudoku, respectively. Comparisons use the most accurate baseline at each threshold.

Reasoning benchmarks. On GSM8K and MATH-500, all three variants on all three backbones reduce steps by 15–21 percent at $\tau = 0.95$, with accuracy differences of -2.2 to $+1.4$ points. Step reductions span 12–32 percent across thresholds and are largest at loose thresholds. On Dream-7B, however, accuracy trails the best baseline by several points at $\tau = 0.70$ and 0.80, so the favorable trade-off holds only at tight thresholds.

Countdown and Sudoku. These puzzles test whether parallel-decoding gains extend to globally constrained outputs. Sudoku requires joint row, column, and subgrid consistency; Countdown requires an expression using the available numbers to reach a target. Token-wise confidence does not ensure joint validity, and permanent commitments prevent later correction. This may explain the more variable accuracy–iteration trade-offs, but our experiments neither isolate this mechanism nor establish that these tasks are inherently unsuitable for parallel decoding.

At $\tau = 0.95$, step reductions range from -8 to $+16$ percent and accuracy differences from -11 to $+10$ points. On Countdown, GoldiMask-D and -P substantially outperform the best baseline on LLaDA-8B, while all variants trail CART on LLaDA-1.5 and LIFT on Dream-7B. Sudoku accuracy differences remain within three points in every setting.

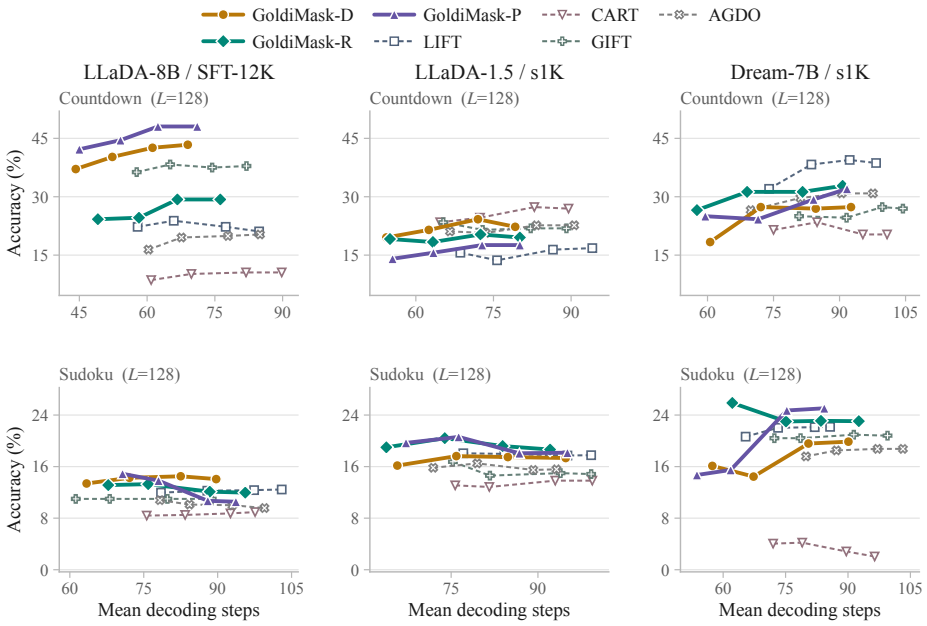

Figure 3: **Accuracy versus decoding cost on Countdown and Sudoku.** Same conventions as Figure 2: three backbones, thresholds $\tau \in \{0.70, 0.80, 0.90, 0.95\}$ from left to right, GoldiMask-D, -R and -P solid against the SFT baselines dashed. Countdown uses $L = 128$ on all backbones.

F Evaluation Protocol and Compute Resources

Checkpoint and generation-length selection. For all experiments, we select each seed's checkpoint solely by the lowest loss on a held-out validation split, following LIFT. For reasoning comparisons other than Table 9, generation length is selected separately for each benchmark from $\{128, 256, 512\}$, with the same procedure across methods.

Training setup and compute resources. We follow the training and evaluation protocol of LIFT (Parashar et al., 2026). Post-training experiments were conducted using four NVIDIA B200 GPUs. We reimplement AGDO (Deng et al., 2026) in our shared harness with matched training budgets and hyperparameters. Each masked candidate is scored by its attention mass on visible context, and the highest-scoring candidates are revealed. The remaining supervised tokens receive weights $1 + \gamma I_i$, where I_i is AGDO's token importance score and $\gamma = 100$. We normalize these weights to

have mean one for comparability. This implementation does not incorporate GoldiMask's selection objective or weighting mechanism.

Training time. Table 11 reports wall-clock training time for LLaDA-8B-Instruct relative to Vanilla SFT at matched optimizer steps and settings on four B200 GPUs. The measurements include all training operations and exclude model loading and evaluation. The reported relative times are averaged across s1K and LIFT-SFT-12K.

Table 11: **Relative training time.** LLaDA-8B-Instruct on four B200 GPUs. Values are averaged across s1K and LIFT-SFT-12K, with Vanilla SFT as the $1.00\times$ reference.

Method	Training time relative to Vanilla SFT
Vanilla SFT	$1.00\times$
CART	$1.01\times$
GIFT	$1.17\times$
AGDO	$1.17\times$
LIFT	$1.24\times$
GoldiMask-D	$1.21\times$
GoldiMask-R	$1.23\times$

G Loss Weighting

After selecting context, GoldiMask converts the measured utilities into loss weights. Section 4.5 compares alternative weighting schemes on LLaDA-8B trained on s1K; here we explain the allocation properties of the chosen rule. Water-filling solves a fixed-utility allocation problem (Proposition 2), and the cap bounds weight concentration (Corollary 1). These properties concern the allocation of loss weight within a training example; they do not imply a fixed gradient norm or optimal downstream learning.

G.1 Properties of the loss weighting

The utility in Equation 6 multiplies the positive log-gain of a target by the priority function λ evaluated at its post-reveal confidence. The product combines evidence that the target responds to the selected context with its remaining priority.

Each training example has a fixed total loss-weight mass $\sum_i w_i = n$ to divide among its supervised tokens. "Proportionally fair" is the standard name for one particular answer: an allocation is proportionally fair when no reallocation can raise the shares of some tokens by more, in utility-weighted proportional terms, than it lowers those of the others. That criterion is exactly what maximizing $\sum_i u_i \log w_i$ encodes, because the logarithm measures each token's satisfaction in *relative* terms, so halving any token's weight costs the same amount whether it held a large share or a small one, and the utilities u_i set how loudly each token argues. The same objective defines the Nash bargaining solution, which is why the two coincide below.

Proposition 2 (Water-filling weights are proportionally fair)**.** *Fix utilities $u_1, \dots, u_n \geq 0$, let $P = \{i : u_i > 0\}$ with $n_+ = |P|$, a cap $c > 1$, and the budget polytope $W = \{w \in \mathbb{R}^n : \sum_i w_i = n,\ 0 \leq w_i \leq c\}$ (nonempty since $c > 1$). Consider the problem $\max_{w \in W} \sum_{i \in P} u_i \log w_i$.*

(a) If $n_+ c > n$, it has the unique solution $w_i^\star = \min\big(u_i/\nu^\star,\ c\big)$ for $i \in P$ and $w_i^\star = 0$ for $i \notin P$, where $\nu^\star > 0$ is the unique root of $\sum_{i \in P} \min(u_i/\nu, c) = n$. Equivalently, $w^\star$ is the asymmetric Nash bargaining solution over the budget set with bargaining powers u_i and disagreement point 0.

(b) If $n_+ c = n > 0$, the unique weights are c on P and zero elsewhere, although the water-filling level is not unique.

(c) If $0 < n_+ c < n$, every solution has $w_i^\star = c$ for all $i \in P$; the remaining budget $n - n_+ c$ may be spread over P^c arbitrarily, and the implementation spreads it uniformly.

(d) If all utilities vanish, the objective is constant on W and the implementation convention is $w_i = 1$ for every i.

Proof. Reduction to P. The objective does not depend on w_i for $i \notin P$, and moving mass from such an i to any uncapped $j \in P$ strictly increases it. Hence at any maximizer either every token in P is capped or $w_i = 0$ on P^c. If $0 < n_+c < n$, capping all of P is forced and leaves $n - n_+c > 0$ to be placed on P^c, where the objective is constant; this is (c). If $n_+c \geq n$, the budget can be met inside P, so $w_i = 0$ on P^c and the problem reduces to the coordinates in P, that is, to the polytope $W_P = \{w \in \mathbb{R}^P : \sum_{i\in P} w_i = n,\ 0 \leq w_i \leq c\}$. The rest of the proof treats this case. If P is empty, the objective is constant, and uniform weighting is feasible; this gives (d).

Existence and uniqueness. On $W_P \cap \{w > 0\}$ the objective $\sum_{i\in P} u_i \log w_i$ is strictly concave, because each term is a positive multiple of the strictly concave log. As any coordinate $w_i \to 0$ the objective tends to $-\infty$, so no maximizer touches the lower boundary, and W_P is compact and convex. The objective is upper semicontinuous on W_P with values in $[-\infty, \infty)$, so a maximum is attained, and by strict concavity at exactly one point.

Shape of the solution. At the maximizer, the Karush–Kuhn–Tucker conditions hold: there are multipliers ν for the budget $\sum_i w_i = n$ and $\mu_i \geq 0$ for the caps $w_i \leq c$ with

$$\frac{u_i}{w_i} \;=\; \nu + \mu_i, \qquad \mu_i\,(w_i - c) = 0.$$

Here $\nu > 0$ in case (a): otherwise every $\mu_i > 0$, every cap binds, and $\sum_{i\in P} w_i = n_+c > n$. At equality, every positive-utility coordinate must instead be capped, which gives the unique weights in (b). Read the two cases off complementary slackness. If the cap is slack ($w_i < c$), then $\mu_i = 0$ and the stationarity condition gives $w_i = u_i/\nu$. If the cap binds, then $w_i = c$, and stationarity requires $\mu_i = u_i/c - \nu \geq 0$, i.e. this case occurs exactly when $u_i/\nu \geq c$. Both cases are summarized by

$$w_i \;=\; \min\bigl(u_i/\nu,\ c\bigr).$$

The level $\nu^\star$ exists and is unique. Define $g(\nu) = \sum_{i\in P} \min(u_i/\nu, c)$, the total weight assigned at level ν. Each summand is continuous and nonincreasing in ν, and strictly decreasing wherever $u_i/\nu < c$; moreover

$$\lim_{\nu\to 0^+} g(\nu) = n_+c, \qquad \lim_{\nu\to\infty} g(\nu) = 0.$$

Since $n < n_+c$ in case (a), the intermediate value theorem gives a root of $g(\nu) = n$, and strict monotonicity below n_+c makes it unique. At $n = n_+c$, every $0 < \nu \leq \min_{i\in P} u_i/c$ gives the same all-capped weights, so the weights are unique but the level is not. In case (a), the root is the $\nu^\star$ found by bisection in the implementation.

Nash bargaining. The asymmetric Nash bargaining solution with bargaining powers u_i and disagreement point 0 maximizes $\prod_{i\in P} w_i^{u_i}$ over the feasible set, and taking logarithms turns this product into $\sum_{i\in P} u_i \log w_i$, the objective above. The two solutions therefore coincide. □

Corollary 1 (A bound on the effective number of supervised tokens)**.** *For any $w \in W$, the effective sample size $n_{\text{eff}} = (\sum_i w_i)^2 / \sum_i w_i^2$ satisfies $n_{\text{eff}} \geq n/c$.*

Proof. $\sum w_i^2 \leq c \sum w_i = cn$, so $n_{\text{eff}} \geq n^2/(cn) = n/c$. □

This is an effective-size bound on the loss weights. It does not count independent gradients or imply a fixed gradient norm. The cap limits weight concentration, while water-filling preserves total loss-weight mass.

Proposition 2 shows that whenever the positive utilities can absorb the budget (i.e., $n_+c \geq n$), the water-filling weights are the *unique* maximizer of $\sum_{i\in P} u_i \log w_i$ for a fixed total loss-weight mass under a per-token cap. Compared to normalize-then-clip, $w_i = \min(n\,u_i/\sum_j u_j,\ c)$, water-filling has the same form but a different level. Normalize-then-clip keeps $\nu' = \sum_j u_j/n$, while water-filling adjusts it to $\nu^\star$ so that the weights still sum to n. Whenever an uncapped normalized weight exceeds the cap, the normalize-then-clip weights sum to less than n: the clipped mass $\sum_i \bigl(n\,u_i/\sum_j u_j - c\bigr)_+$ is lost, not redistributed. It therefore solves the same problem for a smaller mass.

H SUPERVISION PRIORITY AND CONTEXTUAL SUPPORT

Our approach uses the supervision priority $\lambda(p) = p(1-p)^3$ and the contextual support response $\phi(s) = s/(s+0.05)$ to allocate supervision and revealed context. In what follows, we present theoretical and empirical support for these choices. Section H.1 shows that weighting the local probability response by remaining prediction headroom yields our priority function under a simplified training update, giving it an explicit mathematical motivation. Section H.2 establishes that ϕ rewards additional contextual support with diminishing returns, reducing the marginal reward for further supporting an already well-supported target. Section H.3 shows that sampling supervision according to λ improves both held-out denoising loss and average correct-token probability over uniform random selection on both tested backbones. Section H.4 finds that additional context produces the largest absolute probability gains at intermediate initial confidence across the tested models and corpora, supporting confidence-dependent context allocation. Section H.5 finds that a saturating response fits contextual support measurements better than a linear response in high-noise settings, with fitted support scales consistent with our default $s_0 = 0.05$. This section does not establish that these functions are optimal, but explains why we chose them. Additional ablation studies in Section 4.5 provide further empirical support for these choices.

H.1 MOTIVATION FOR THE SUPERVISION-PRIORITY FUNCTION

GoldiMask allocates a limited supervision budget among masked targets. We seek a priority that accounts for both a target's response to additional supervision and its remaining room for improvement. To motivate this choice, we consider how a small cross-entropy update changes the probability assigned to the correct token.

Under a simple local model, this probability increase scales as $p(1-p)^2$. We additionally weight the response by the remaining probability headroom, $1-p$, to place greater emphasis on predictions that still need improvement. This gives $\lambda(p) = p(1-p)^3$. The headroom factor shifts the maximum from $p = 1/3$ for the response alone to $p = 1/4$ for our priority. The following proposition formalizes this construction.

Proposition 3 (Local response weighted by remaining headroom)**.** *Let $p = \sigma(m) \in (0,1)$ be the gold-token probability expressed through a gold-versus-rest margin m. Apply one gradient-descent step of size η to $\ell(m) = -\log \sigma(m)$, treating m as the trainable scalar. If p^+ is the updated probability, then, as $\eta \to 0$,*

$$p^+ - p = \eta p(1-p)^2 + O(\eta^2), \qquad (1-p)(p^+ - p) = \eta p(1-p)^3 + O(\eta^2). \tag{9}$$

The confidence factor in the headroom-weighted response, $\lambda(p) = p(1-p)^3$, has its unique maximum at $p = 1/4$.

Proof. Since $\ell'(m) = -(1-p)$, the updated margin is $m^+ = m + \eta(1-p)$. Using $\sigma'(m) = p(1-p)$, Taylor expansion gives $\sigma(m^+) - \sigma(m) = \eta p(1-p)^2 + O(\eta^2)$. Multiplying by $1-p$ gives the second identity. Finally, $\lambda'(p) = (1-p)^2(1-4p)$ is positive below $1/4$ and negative above it, while $\lambda(0) = \lambda(1) = 0$. □

This construction gives our priority a concrete interpretation as local probability response weighted by remaining headroom. The additional headroom factor expresses our allocation preference. In GoldiMask's reveal objective, this priority is multiplied by contextual support, favoring reveal sets that provide useful evidence to high-priority targets left masked.

The same confidence factor also appears beyond the scalar model. For a direct cross-entropy update of all vocabulary logits, let q be the softmax probability vector, y the correct token, and $q_y = p$. Then

$$(1-p)\Delta p = \eta \left(1 + \sum_{j \neq y} r_j^2\right) p(1-p)^3 + O(\eta^2),$$

where $r_j = q_j/(1-p)$ describes the conditional probability distribution over incorrect tokens. The additional factor lies between $1 + 1/(V-1)$ and 2, where V is the vocabulary size. Updates to shared model parameters also depend on parameter sensitivity and interactions among supervised targets.

These calculations motivate a simple confidence-based proxy; Section H.3 examines its usefulness for allocating supervision during actual fine-tuning.

H.2 Motivation for the saturating support function

After assigning supervision priorities, GoldiMask must assess how much support a reveal set provides to the remaining targets. A linear support score gives every additional unit of attention support the same value, regardless of how much support a target already receives. We instead use a saturating response, so that additional support remains beneficial but contributes less as support accumulates. This encourages the reveal objective to account for targets that still lack contextual support.

For fixed attention, the support received by target i from the revealed set R is $S_i(R) = \sum_{j \in R} A_{ij}$. We map this support to a bounded score using

$$\phi(s) = \frac{s}{s + s_0}, \qquad s_0 > 0.$$

The parameter s_0 controls the support scale at which the response reaches half its maximum. GoldiMask uses $s_0 = 0.05$.

Proposition 4 (Diminishing returns from support)**.** *For $s_0 > 0$, the response $\phi(s) = s/(s + s_0)$ is increasing and concave on $s \geq 0$, with*

$$\phi(0) = 0, \qquad \phi(s_0) = \tfrac{1}{2}, \qquad \lim_{s \to \infty} \phi(s) = 1.$$

Consequently, for any fixed additional support $a \geq 0$, the gain $\phi(s + a) - \phi(s)$ is nonincreasing in s.

Proof. Direct differentiation gives

$$\phi'(s) = \frac{s_0}{(s + s_0)^2} > 0, \qquad \phi''(s) = -\frac{2s_0}{(s + s_0)^3} < 0.$$

The endpoint identities follow by substitution and taking the limit. Moreover,

$$\phi(s + a) - \phi(s) = \frac{a s_0}{(s + s_0)(s + a + s_0)},$$

which is nonincreasing in s. □

Thus, for targets with equal supervision priority, the same additional support earns a larger marginal reward when existing support is smaller. Combined with λ, this lets the objective account for both target priority and diminishing contextual benefit. Concavity also underpins the submodularity result in Proposition 1. Section H.5 examines saturation empirically and provides the measurements informing our default support scale.

H.3 Study 1: confidence-based supervision and subsequent learning

Section H.1 motivates $\lambda(p) = p(1-p)^3$ through local probability response and remaining headroom. We now ask whether using this priority to choose supervised targets improves learning on unseen examples. The experiment holds the number of supervised tokens fixed and changes only how those tokens are selected.

A matched supervision budget. We train LLaDA-8B and LLaDA-1.5 on 600 examples from s1K, indexed $[300, 900)$, for 300 optimizer steps with two seeds per backbone. Within each seed, all rules start from the same model state and use the same examples, order, and corruptions. For each example, exactly 32 masked positions carry the training loss. The remaining masked positions stay masked but do not contribute directly to that example's loss. We compare uniform random selection with sampling proportional to $p(1-p)^3$, $1-p$, or $p(1-p)$, where p is the current model's probability of the correct token before the update. This isolates the supervision-priority component; the experiment does not run GoldiMask's full reveal-selection and capped-weighting procedure.

All runs use LoRA with rank 128, scaling 256, and dropout 0.05 on the query/key/value projections. Optimization uses AdamW with learning rate 5×10^{-6}, weight decay 0.1, gradient clipping at 1.0, and global batch size 8. The four reported rules give 16 training blocks across the two backbones and two seeds.

Measuring learning on unseen examples. At steps 0, 100, and 300, we evaluate 90 held-out examples using the same fixed masks at masking rates $t \in \{0.25, 0.5, 0.75\}$. For a masked target, negative log-likelihood (NLL) is $-\log p$, where p is the probability of its correct token. We report two changes from before to after training. ΔNLL is the change in average held-out denoising loss, so negative values indicate improvement. Δp is the average increase in correct-token probability, evaluated on held-out targets whose initial probability is below 0.9, so positive values indicate improvement. These measurements compare model checkpoints on fixed inputs, rather than responses to newly revealed context. NLL and probability gain summarize different aspects of prediction quality and can rank supervision rules differently.

Table 12: **Learning under a fixed supervision budget.** Changes after 300 training steps, with 32 supervised positions per training example. ΔNLL is the change in held-out negative log-likelihood; Δp is the mean correct-token probability change for held-out targets with initial $p < 0.9$. Arrows indicate the preferred direction. Bold values are best within each column; shading identifies our priority.

	LLaDA-8B		**LLaDA-1.5**	
Supervision rule	ΔNLL $\downarrow$	Δp $\uparrow$	ΔNLL $\downarrow$	Δp $\uparrow$
Uniform random	-0.013	$+0.013$	-0.015	$+0.011$
$1-p$	$\mathbf{-0.017}$	-0.013	$\mathbf{-0.019}$	-0.014
$p(1-p)$	-0.001	$\mathbf{+0.052}$	-0.003	$\mathbf{+0.051}$
Our priority $p(1-p)^3$	-0.016	$+0.032$	-0.018	$+0.030$

Held-out results. Our priority improves both held-out NLL and average correct-token probability more than uniform random selection on both backbones (Table 12). The alternatives favor different outcomes. The $1-p$ rule reduces NLL slightly more but lowers mean correct-token probability; $p(1-p)$ raises mean probability more but achieves much smaller NLL reductions. Our priority provides a balance between the two metrics.

H.4 Study 2: context-response profiles across models and corpora

GoldiMask uses target confidence to guide the allocation of revealed context. We therefore examine how the benefit of additional context varies with a target's initial confidence. Using 100 examples and four paired masking draws per example, we evaluate each frozen model under two nested masks. For targets that remain masked in both inputs, we measure

$$\Delta p = p_2 - p_1, \tag{10}$$

where p_1 and p_2 are the correct-token probabilities before and after revealing additional context. We group these gains by initial confidence p_1.

Across the tested models and corpora, absolute probability gains peak at intermediate confidence. The LLaDA-8B measurements illustrate a broad high-response region. Average gains are 0.46 for initial confidence 0.05–0.10, 0.51 for 0.10–0.20, and 0.49 for 0.20–0.30, falling to 0.01 above 0.90. Thus, although the fitted maximum lies near 0.16, the neighboring bin containing 0.25 retains nearly the same average gain.

Table 13 summarizes fitted peak locations across all four model–corpus settings. The confidence intervals quantify uncertainty in the peak location. Held-out R^2 measures how well a curve fitted on half the examples describes the response profile on the other half, with 1 indicating a perfect fit.

These measurements support confidence-dependent context allocation. The substantial gains across neighboring intermediate-confidence bins are consistent with GoldiMask's smooth priority, which emphasizes responsive targets while retaining a preference for remaining headroom.

Table 13: Study 2: absolute probability gain indexed by pre-reveal confidence. Peak intervals use an example-level bootstrap. The power-family fit is trained on half the examples and evaluated on the other half.

Model / corpus	Fitted peak [95% CI]	Held-out R^2
LLaDA-8B / s1K	$0.159\ [0.148, 0.170]$	0.958
LLaDA-1.5 / s1K	$0.157\ [0.147, 0.168]$	0.952
Dream-7B / s1K	$0.164\ [0.154, 0.177]$	0.968
LLaDA-8B / 12K	$0.163\ [0.144, 0.179]$	0.973

H.5 Study 3: diminishing returns from contextual support

Study 2 examines which targets respond to context. Here we test whether prediction responses support GoldiMask's saturating contextual support score $\phi(s) = s/(s + 0.05)$ rather than a linear response.

Comparing targets across contexts. On LLaDA-8B, we evaluate eight random reveal sets per configuration over 25,217 targets, holding target masks, noise level, and reveal budget fixed. For each target and reveal set, we record attention mass on revealed tokens and the correct token's log-probability. Target-specific intercepts account for fixed differences in difficulty. For held-out evaluation, we center responses and predictors within each target cell, fit one slope per noise bin on even-indexed examples, and evaluate on odd-indexed examples. We swap the halves and average the two scores.

Evidence for saturation. With $s_0 = 0.05$, saturation improves held-out within-target R^2 from 0.033 to 0.041 and from 0.082 to 0.098 in masking intervals 0.50–0.71 and 0.71–0.84, respectively (Table 14). These scores measure explained variation across contexts after removing target-specific means. The advantage is smaller above $t = 0.84$ and absent below $t = 0.25$, supporting diminishing returns particularly in the two reported intervals.

Support scale. The response $\phi(s) = s/(s+s_0)$ reaches half its maximum at $s = s_0$. Fitted scales span 0.05–0.2 across noise bins, placing our default 0.05 at the lower end of the observed range. For equally prioritized targets, ϕ rewards additional support more when existing support is smaller.

Table 14: **Study 3: contextual support and diminishing returns.** Support fits on LLaDA-8B. Arrows compare held-out within-target R^2 for the linear and saturating responses, with $s_0 = 0.05$ for the latter.

Measurement	Result
Held-out R^2, masking interval 0.50–0.71	$0.033 \to 0.041$
Held-out R^2, masking interval 0.71–0.84	$0.082 \to 0.098$
Fitted half-saturation scale s_0 across noise bins	0.05–0.2

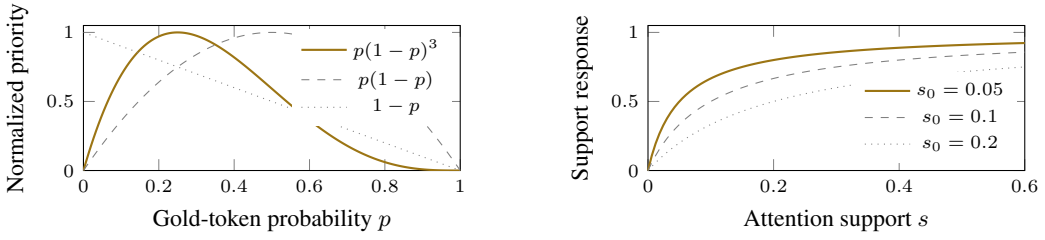

Figure 4: **GoldiMask's functional forms** (analytical, not empirical). Left: priorities normalized to unit maximum. Right: support responses for $s_0 = 0.05$ (default), 0.1, and 0.2. Empirical fits appear in Table 14.