Principled Self-Correction in Discrete Diffusion: A UCB-Guided Framework for Text Generation

Masaki Asada[1]
[1]National Institute of Advanced Industrial Science and Technology
masaki.asada@aist.go.jp

Makoto Miwa[2,1]
[2]Toyota Technological Institute
makoto-miwa@toyota-ti.ac.jp

Abstract

Inspired by their success in image synthesis, diffusion models offer a flexible, iterative alternative to rigid left-to-right text generation. However, a fundamental training-inference discrepancy hinders their performance: models are trained on corrupted ground-truth tokens, but at inference time they must denoise inputs corrupted from their own predictions. To bridge this gap, we propose a unified framework. First, **Deeper Self-Prediction (DSP)** is a multi-step training objective that teaches robust self-correction by forcing the model to denoise its own intermediate outputs. Second, **UCB-guided Decoding** is a principled inference algorithm that frames token re-masking as a multi-armed bandit problem, using the Upper Confidence Bound (UCB) to balance exploration and exploitation. Experiments on text generation tasks demonstrate consistent improvements over existing diffusion baselines. The framework achieves higher faithfulness and coherence according to both automatic metrics and LLM-as-a-Judge evaluations.

1 Introduction

The field of natural language generation (NLG) has been largely dominated by autoregressive (AR) models, which construct text through a strictly sequential, left-to-right process (Lewis et al., 2020; Zhang et al., 2022). While powerful, this paradigm imposes a fixed generation order. Inspired by the transformative success of diffusion models in continuous domains such as image synthesis (Ho et al., 2020), there is a compelling motivation to explore their potential for discrete text generation. This alternative paradigm, based on iterative refinement, is not constrained by a fixed generation order. Instead, it allows for a holistic, non-monotonic process where the entire text sequence is gradually denoised from a random state, offering a fundamentally more flexible approach to generation.

However, adapting this promising paradigm to the discrete nature of text presents significant challenges. We argue that a primary obstacle limiting the quality of current text diffusion models is a fundamental **training-inference discrepancy**. This issue arises from a profound mismatch between what the model learns and what it is asked to do at inference time. During its **training task**, the model learns a simplified objective: recovering a *clean sequence* from an input corrupted with uniform random noise (i.e., randomly masked tokens) (Austin et al., 2021a; Lin et al., 2023). This is a controlled, artificial task where errors are independent and lack semantic structure. In contrast, during its **inference task**, the model must iteratively refine its *own outputs*, denoising self-generated, often high-confidence, structured errors and hallucinations (Ji et al., 2023), despite having been trained only to denoise corrupted clean text.

This misalignment means the model is never explicitly trained to recover from the specific types of structured, correlated errors it is guaranteed to produce during inference. This leads to error accumulation and premature convergence to fluent but factually incorrect local optima (Holtzman et al., 2020), preventing the model from reaching its full generative potential.

To bridge this discrepancy, we propose a unified framework designed to align the training objective with the reality of inference. Our approach is two-pronged. First, we introduce **Deeper Self-Prediction (DSP)** during training, a novel, generalized K-step training objective. Instead of merely denoising random corruption, the model is trained to recursively correct its own intermediate predictions. This procedure explicitly teaches the model robust self-correction, making it more adept at handling the errors it will encounter during inference. Our method formalizes and generalizes the 2-step loss concept from prior work (Asada and Miwa, 2025) into a multi-step framework.

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6678–6692
March 24-29, 2026

Second, to fully leverage the enhanced capabilities of a DSP-trained model, we introduce **UCB-guided Decoding** during inference. A model trained with DSP possesses better-calibrated uncertainty estimates. Our principled inference algorithm capitalizes on this by formulating the selection of tokens to re-mask as a multi-armed bandit problem. Applying the Upper Confidence Bound (UCB) algorithm (Lai and Robbins, 1985), it provides a theoretically grounded mechanism to balance exploiting known weaknesses (high-uncertainty tokens) and exploring seemingly correct parts of the sequence to escape local optima.

The two components complement each other effectively: DSP training yields a model with better-calibrated uncertainty estimates, and UCB-guided Decoding leverages this information for a more accurate and robust search of the output space. Our experiments on text summarization, question generation, and math reasoning tasks show that this combined framework outperforms diffusion baselines, achieving large gains in factual consistency and overall quality, thereby advancing the viability of diffusion models as a paradigm for text generation.[1]

2 Related Work

2.1 Discrete Diffusion Models for Text

Adapting diffusion models from continuous domains like images to discrete text has been a central challenge (Hoogeboom et al., 2022; Austin et al., 2021b). Current approaches can be broadly categorized into two families. The first operates in a continuous latent space, applying diffusion to token embeddings, but this can introduce information loss during the discrete-to-continuous mapping (Li et al., 2022b; Gong et al., 2023). The second, which our work builds upon, operates directly in the discrete space by defining the forward (noising) process as progressive token masking or absorption (Austin et al., 2021a). This paradigm can be viewed as an iterative extension of non-autoregressive (NAR) text generation (Li et al., 2022a; Sohrab et al., 2024). Its iterative nature gives rise to the training–inference discrepancy that our work explicitly addresses.

[1]Our codes are available at https://github.com/aistairc/UCB-DSP-DiffusionLM

2.2 The Training-Inference Discrepancy and Exposure Bias

The gap between training conditions and inference reality is a long-standing problem in sequence generation. In AR models, it is known as "exposure bias," where a model trained on ground-truth prefixes (teacher forcing) struggles when conditioning on its own, potentially flawed, predictions during inference (Bengio et al., 2015). While some argue its impact on strong AR models for short generation is limited (Schmidt, 2019), its effects are amplified in the iterative refinement setting of diffusion models. Here, errors are not just propagated forward but can corrupt the entire sequence representation at each step, leading to cascading failures.

Recent work has begun to tackle this issue in diffusion models directly. Asada and Miwa (2025) proposed a two-step loss to force the model to learn from its own predictions. Similarly, Tang et al. (2023) noted that "injecting the noise generated by the model itself into the training stage" can improve performance. Our UCB-DSP framework formalizes and generalizes this intuition into a multi-step objective, providing a more powerful mechanism for teaching self-recovery.

2.3 Self-Prediction and Scheduled Sampling

Our UCB-DSP training objective is conceptually related to a broader class of techniques that expose a model to its own outputs during training. The most classic example is scheduled sampling (Bengio et al., 2015) for AR models, which stochastically feeds the model either a ground-truth token or its own previously generated token as input during training. UCB-DSP can be viewed as a holistic, sequence-level analogue for diffusion models: instead of making a token-level choice, we generate an entire candidate sequence and then train the model to recover from it. This core idea of mitigating exposure bias by training on model-generated states has also been successfully adapted to diffusion models in other domains, validating the approach's potential (Zhang et al., 2025).

2.4 Adaptive and Principled Decoding Strategies

Standard decoding algorithms for text generation include greedy search, beam search, and stochastic methods like top-k or nucleus sampling (Holtzman et al., 2020). More advanced techniques employ an adaptive, iterative refinement process, often re-

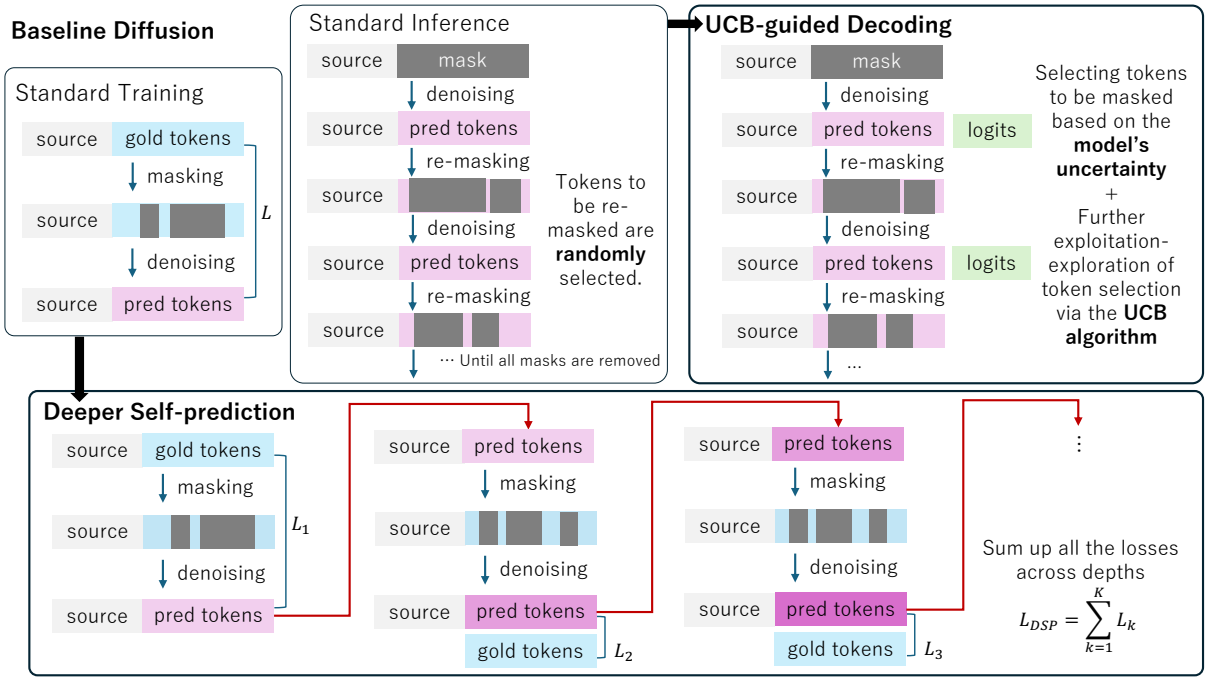

Figure 1: Overview of the proposed UCB-guided Deeper Self-Prediction (DSP) text diffusion framework. The model combines DSP during training and UCB-guided Decoding during inference. DSP teaches the model to recover from its own intermediate outputs, while UCB-guided Decoding adaptively selects tokens to re-mask based on uncertainty and exploration–exploitation trade-off.

masking and regenerating tokens for which the model has low confidence (Ghazvininejad et al., 2019). However, these methods are primarily exploitative, focusing on fixing obvious errors.

Our UCB-guided decoding approach is distinct in its principled handling of the exploration-exploitation trade-off, a concept formalized by the multi-armed bandit (MAB) framework (Lai and Robbins, 1985). By applying the UCB algorithm, which is guided by the principle of "optimism in the face of uncertainty," we provide a theoretically grounded mechanism for deciding when to refine uncertain tokens versus when to revisit high-confidence ones to escape local optima. This moves beyond simple heuristic or entropy-based criteria for adaptive decoding.

3 The UCB-DSP Framework

This section describes our proposed framework. An overview of the entire model architecture and training–inference flow is illustrated in Figure 1.

3.1 Preliminaries: Discrete Diffusion for Text

We first define the standard discrete diffusion process for a target text sequence $Y_0 = (y_1, y_2, \ldots, y_L)$.

Forward Process (q): The forward process gradually corrupts the clean sequence Y_0 into a fully masked sequence Y_T over T timesteps. At each step t, tokens are replaced by a special "<MASK>" token. The state Y_t can be sampled directly from Y_0 using a predefined noise schedule $\bar{\alpha}_t$:

$$q(Y_t|Y_0)$$

where each token y_i is masked with probability $1 - \bar{\alpha}_t$.

Reverse Process (p_θ): The reverse process is a learned denoising model f_θ that aims to recover the original sequence Y_0 from a noisy input Y_t and a source context X. The model predicts the probability distribution for each token in the original sequence:

$$\hat{P}_0 = f_\theta(Y_t, X)$$

The standard training objective is to minimize the negative log-likelihood (cross-entropy) between the

predicted distribution $\hat{P}_0$ and the one-hot representation of the ground-truth sequence Y_0:

$$\mathcal{L}_{\text{standard}} = \mathbb{E}_{t,Y_0,Y_t}\left[-\log p_\theta(Y_0|Y_t, X)\right]$$

The baseline discrete diffusion model is implemented with a decoder-only Transformer architecture. The input sequence is constructed by concatenating a source text and a target text, which can also be regarded as a prompt and a response, respectively. Following prior diffusion-based text generation frameworks (Nie et al., 2025), the forward (noising) and reverse (denoising) processes are applied only to the target segment, while the source remains uncorrupted throughout both training and inference. Unlike standard autoregressive models that employ a causal attention mask restricting each token to attend only to its left-hand context, diffusion models employ a fully visible attention mask, allowing each token in the sequence to attend to both past and future tokens.

3.2 Deeper Self-Prediction during Training

The core of our training-time contribution is the Deeper Self-Prediction (DSP) objective, which simulates the multi-step error correction process required at inference. Instead of a single denoising step, we train the model over a sequence of K self-prediction steps.

Let Y_t be the initial noisy input. The process unfolds as follows:

Step 1: The model predicts the clean sequence: $\hat{Y}_0^{(1)} = \text{argmax} f_\theta(Y_t, X)$. The loss $\mathcal{L}_1$ is computed between the model's logits and the ground-truth Y_0.

Step 2: The model's own prediction $\hat{Y}_0^{(1)}$ is treated as a new ground-truth, re-corrupted to the same noise level t to produce $Y_t^{(1)} \sim q(Y_t|\hat{Y}_0^{(1)})$. The model then denoises this new state, and the loss $\mathcal{L}_2$ is computed against the original ground-truth Y_0.

Step k (up to K): This process is repeated. The input for depth k is generated by corrupting the prediction from depth $k-1$: $Y_t^{(k-1)} \sim q(Y_t|\hat{Y}_0^{(k-1)})$. The model predicts $\hat{Y}_0^{(k)}$ and the loss $\mathcal{L}_k$ is computed against Y_0.

The total DSP loss is a sum of the losses from each self-prediction step:

$$\mathcal{L}_{\text{DSP}} = \sum_{k=1}^{K} \mathcal{L}_k$$

3.3 UCB-guided Decoding during Inference

At inference, we employ UCB-guided Decoding to guide the iterative refinement process. This approach reframes the decision of which tokens to re-mask at each step as a multi-armed bandit (MAB) problem, providing a principled way to balance exploration and exploitation.

The Multi-Armed Bandit Analogy: At each denoising step t, we have a limited budget of tokens we can choose to re-mask. Each token position i can be thought of as a "slot machine" or "arm." "Pulling the arm" corresponds to re-masking that token and allowing the model to regenerate it. The "reward" is the potential improvement in the overall quality of the generated sequence. The goal is to allocate our limited pulls to the arms that will yield the highest total reward. The challenge is that the rewards are uncertain. Some tokens are obviously problematic (high uncertainty), while others seem correct but might be part of a larger, subtle error (a local optimum).

The UCB Algorithm: The UCB algorithm provides a theoretically grounded solution to this exploration-exploitation dilemma. It operates on the principle of "optimism in the face of uncertainty." The score for selecting token i for re-masking at denoising step t (from T down to 1) is calculated as:

$$\text{Score}_i(t) = V_i + C\sqrt{\frac{\log(T-t+1)}{N_i(t)+1}}$$

This score has two components:

1. Value Heuristic (V_i): This is the "exploitation" term, which estimates the immediate value of re-masking token i. As value heuristics, the following can be used:

- **Least Confidence-based:** $V_i = 1 - P_i^{(\text{top-1})}$, where $P_i^{(\text{top-1})}$ denotes the probability of the most likely token. Higher values represent lower model confidence, signifying a greater need for re-masking.
- **Entropy-based:** $V_i = -\sum_j P_{i,j} \log P_{i,j}$, where $P_{i,j}$ is the predicted probability of token j at position i. Higher entropy corresponds to higher prediction uncertainty.
- **Margin-based:** $V_i = 1 - (P_i^{(\text{top-1})} - P_i^{(\text{top-2})})$, which measures ambiguity between

Algorithm 1 UCB-guided Decoding

Input: Source X, model f_θ, total steps T, factor C
Initialize $Y^{(T)}$ with all <MASK> tokens
Initialize counts $N_i = 0$ for all positions $i = 1, \ldots, L$
for $t = T$ down to 1 **do**
  Get logits $L^{(t-1)} = f_\theta(Y^{(t)}, X)$
  Get candidate sequence $Y_{\text{cand}}^{(t-1)} = \text{argmax}(L^{(t-1)})$
  Get probabilities $P^{(t-1)} = \text{softmax}(L^{(t-1)})$
  for each token position $i = 1$ to L **do**
    Compute value V_i from $P_i^{(t-1)}$
    Compute score $\text{Score}_i = V_i + C\sqrt{\frac{\log(T-t+1)}{N_i+1}}$
  end for
  Determine number of tokens to mask, $k_t = \lceil L \cdot t/T \rceil$
  Select tokens to mask $\mathcal{M}_t$ = top-k_t indices by Score_i
  for each $i \in \mathcal{M}_t$ **do**
    $N_i \leftarrow N_i + 1$
  end for
  Construct next input $Y^{(t-1)}$ where:

$$Y_i^{(t-1)} = \begin{cases} \texttt{<MASK>} & \text{if } i \in \mathcal{M}_t \\ Y_{\text{cand},i}^{(t-1)} & \text{otherwise} \end{cases}$$

end for
return $Y_{\text{cand}}^{(0)}$

the top two candidate probabilities. A smaller gap implies higher uncertainty and a greater chance of beneficial refinement.

2. UCB Bonus: This is the "exploration" term. $N_i(t)$ is the number of times token i has been re-masked up to step t, and C is an exploration hyperparameter. This term gives a higher score to tokens that have been revisited less frequently. It encourages the model to re-evaluate confident-but-potentially-wrong predictions, providing a mechanism to escape the local optima that purely exploitative (e.g., confidence-based) re-masking strategies can fall into.

The complete algorithm is detailed in Algorithm 1. At each step t, we calculate the score for all tokens, select the top-k_t tokens with the highest scores to mask, and proceed to the next step.

Handling of Padding Tokens In confidence-based re-masking, positions predicted as padding tokens (used to fill the sequence length) often exhibit substantially higher confidence than positions corresponding to normal tokens, which can distort the behavior of the proposed confidence-based UCB-guided decoding. To address this issue, we partition token positions into two groups based on whether the argmax of the logits corresponds to a padding token or not, and apply UCB-guided decoding independently within each group.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate our framework on four representative conditional text generation benchmarks: two for text summarization (XSum (Narayan et al., 2018) and MSNews (Liu et al., 2021)), and two for question generation (SQuAD (Rajpurkar et al., 2016) and MSQG (Liu et al., 2021)). XSum involves highly abstractive and factually demanding summaries, while MSNews covers longer and more diverse news-style documents. SQuAD and MSQG evaluate reasoning-oriented generation quality, including answer relevance and contextual consistency. Dataset statistics are provided in Appendix A.

Model Backbone. Our baseline discrete diffusion model is implemented with a decoder-only Transformer architecture based on Llama-3.2-1B[2]. The model takes as input the concatenation of a *source* text X and a *target* text Y_0 (equivalently, a *prompt* and a *response*). As described in subsection 3.1, the forward and reverse diffusion processes are applied only to the target segment Y_0, while X remains visible throughout. A *fully visible attention mask* is employed to enable bidirectional context access, in contrast to the causal masking used in standard autoregressive models.

Training Configuration. All models are trained under identical conditions for fair comparison. We use the AdamW optimizer with linear warm-up and cosine decay scheduling. For DSP training, the number of self-prediction steps K is determined based on validation performance, specifically by selecting the configuration that achieves the highest Faithfulness/Consistency score on the validation set. Uniform step-wise loss weights ($\lambda_i = 1/k$) are applied during training. For UCB-guided Decoding, the exploration coefficient C is fixed to 1.5 unless otherwise stated, following the theoretical constant in the original UCB formulation and used unchanged across all datasets. Detailed hyperparameters are listed in Appendix B.

Evaluation Protocol. We report both automatic metrics and LLM-as-a-Judge evaluations. For summarization, ROUGE-1/2/L (Lin, 2004) is used to measure content overlap. For question generation, we report ROUGE-L, BLEU-4 and METEOR

[2] https://huggingface.co/meta-llama/Llama-3.2-1B

Dataset	Model Configuration		R-1	R-2	R-L	Faith.	Cove.	Cohe.
XSum	AR	Transformer (Vaswani et al., 2017)	30.6	10.8	24.4	-	-	-
		BART (Lewis et al., 2020)	38.7	16.1	30.6	-	-	-
		ProphetNet (Qi et al., 2020)	39.8	17.1	32.0	-	-	-
	Diffu.	Diffusion-NAT (Zhou et al., 2024)	38.8	15.3	30.8	-	-	-
		Two-step Diffusion (Asada and Miwa, 2025)	38.5	14.8	30.9	-	-	-
		Standard Training and Decoding (Baseline)	39.31	14.67	30.92	27.20	10.91	29.06
		UCB-DSP (Ours)	**43.25**	**19.25**	**35.26**	**47.51**	**15.04**	**43.18**
		w/o UCB-guided Decoding	40.71	17.05	33.79	25.69	8.41	21.39
		w/o Deeper Self-Prediction	41.69	17.83	33.93	43.60	12.54	37.80
MSNews	AR	Transformer (Vaswani et al., 2017)	33.0	15.4	30.0	-	-	-
		BART (Lewis et al., 2020)	41.8	23.1	38.3	-	-	-
	Diffu.	Diffusion-NAT (Zhou et al., 2024)	46.8	31.6	44.2	-	-	-
		Two-step Diffusion (Asada and Miwa, 2025)	50.5	35.1	48.0	-	-	-
		Standard Training and Decoding (Baseline)	49.93	34.54	46.92	62.62	30.26	44.37
		UCB-DSP (Ours)	**51.69**	**36.47**	**49.02**	**69.73**	**29.67**	**48.46**
		w/o UCB-guided Decoding	50.09	34.93	47.42	61.20	28.75	42.58
		w/o Deeper Self-Prediction	50.89	35.97	48.33	69.53	29.49	47.80

Table 1: Main ablation results on the **text summarization** tasks (XSum and MSNews). R-1, R-2, and R-L denote ROUGE-1, ROUGE-2, and ROUGE-L scores, respectively. Faith., Cove., and Cohe. represent Faithfulness, Coverage, and Coherence scores evaluated by the LLM-as-a-Judge. Higher is better for all metrics. **Bold** numbers indicate the best performance in each column.

Dataset	Model Configuration		R-L	B-4	MT	Cons.	AnsCons.	Cohe.
SQuAD	AR	Transformer (Vaswani et al., 2017)	29.4	4.6	9.8	-	-	-
		BART (Lewis et al., 2020)	42.5	17.0	23.1	-	-	-
		ProphetNet (Qi et al., 2020)	**48.0**	**19.5**	**23.9**	-	-	-
	Diffu.	Diffusion-NAT (Zhou et al., 2024)	46.6	16.1	21.9	-	-	-
		Two-step Diffusion (Asada and Miwa, 2025)	43.5	15.4	23.0	-	-	-
		Standard Training and Decoding (Baseline)	40.87	13.39	20.86	63.60	60.25	53.56
		UCB-DSP (Ours)	44.07	16.06	21.81	**76.92**	**71.89**	**73.31**
		w/o UCB-guided Decoding	41.37	13.24	19.56	54.86	51.69	44.44
		w/o Deeper Self-Prediction	43.64	15.40	21.40	74.98	69.57	71.23
MSQG	AR	Transformer (Vaswani et al., 2017)	29.3	5.1	16.6	-	-	-
		BART (Lewis et al., 2020)	38.1	10.2	**22.1**	-	-	-
	Diffu.	Diffusion-NAT (Zhou et al., 2024)	33.3	6.6	19.3	-	-	-
		Two-step Diffusion (Asada and Miwa, 2025)	39.0	8.0	20.5	-	-	-
		Standard Training and Decoding (Baseline)	39.20	8.04	20.89	75.45	71.57	57.2
		UCB-DSP (Ours)	40.65	**9.05**	20.91	**80.47**	**74.51**	**62.84**
		w/o UCB-guided Decoding	38.91	8.17	20.05	72.74	67.85	55.93
		w/o Deeper Self-Prediction	**40.75**	8.98	20.82	79.97	73.58	61.81

Table 2: Main ablation results on the **question generation** tasks (SQuAD and MSQG). R-L, B-4, and MT denote ROUGE-L, BLEU-4, and METEOR scores, respectively. Cons., AnsCons., and Cohe. correspond to Consistency, Answer Consistency, and Coherence scores evaluated by the LLM-as-a-Judge. Higher values indicate better performance. **Bold** numbers denote the best results in each column.

scores. To further assess generation quality beyond lexical overlap, we employ an **LLM-as-a-Judge** (GPT-4.1-mini), which evaluates each generated output on a 1–5 scale across multiple dimensions: *Faithfulness*, *Coverage*, and *Coherence* for summarization, and *Consistency*, *Answer Consistency*, and *Coherence* for question generation. Scores are first averaged across all samples and then scaled to a percentage range (0–100) for reporting. The evaluation prompt and scoring guidelines are detailed in Appendix C.

4.2 Main Results: Complementary Effects of DSP and UCB-guided Decoding

Table 1 and Table 2 summarize our main results on abstractive summarization and question generation, while Table 3 reports performance on the

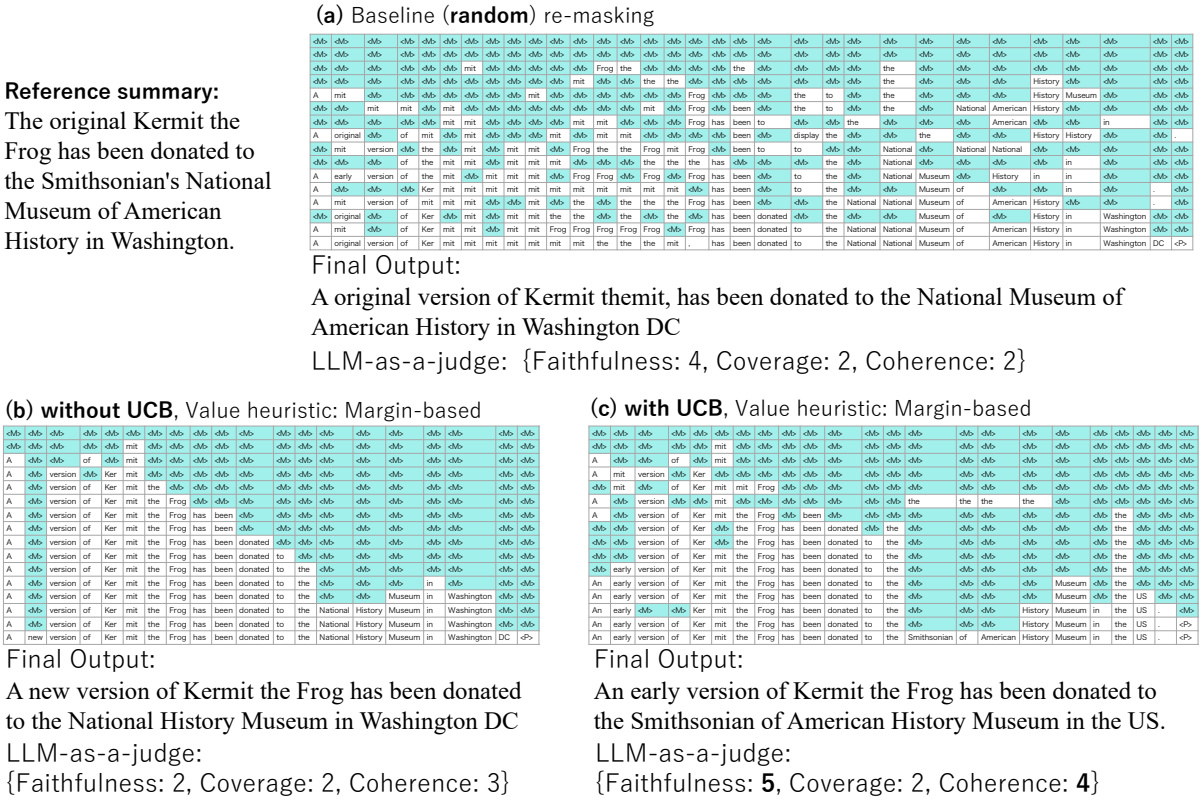

Figure 2: Visualization of the iterative denoising process under different re-masking strategies. Each row represents token states across diffusion steps, where <M> with teal colored cell denotes masked tokens and <P> denotes padding tokens. (a) **Baseline (Random)**: tokens to be re-masked are selected uniformly at random at each step. (b) **Confidence-guided (Uncertainty)**: tokens with higher prediction uncertainty are preferentially re-masked. (c) **UCB-guided (Ours)**: extends (b) by incorporating a UCB exploration term.

Model Configuration	Accuracy (%)
Baseline	46.92
Ours	**57.31**
w/o UCB-guided Decoding	52.84
w/o DSP	53.60

Table 3: Performance comparison on the GSM8K task

GSM8K arithmetic reasoning benchmark. Across all benchmarks, our proposed integrated framework (UCB-DSP) consistently and substantially outperforms the standard diffusion baseline, as well as the single-component variants, with the only exception being ROUGE-L on the MSQG dataset.

Effect of Each Component. Introducing either component alone yields clear improvements over the baseline. Applying UCB-guided Decoding to a standard-trained model (w/o DSP) improves Faithfulness and Consistency by encouraging a more balanced exploration–exploitation strategy during inference, allowing the model to escape local optima caused by overconfident predictions. Training with DSP alone (w/o UCB) produces even larger and more stable gains across metrics, indicating that explicit exposure to the model's own intermediate predictions substantially enhances its self-correction ability and uncertainty calibration. A similar trend is observed on GSM8K: removing either component degrades performance (52.84% w/o UCB and 53.60% w/o DSP) compared to the full model (57.31%), while both remain clearly above the baseline (46.92%). These findings confirm that both training- and inference-level modifications independently contribute to better factual accuracy and coherence.

Combined Model. When DSP training and UCB-guided decoding are combined, the model achieves the strongest overall performance. On XSum, faithfulness improves from 27.2 (Baseline) to 47.5, with nearly a four-point gain in ROUGE-L. Improvements are also observed on MSNews and both question-generation benchmarks, though the additional gains over DSP-only training are smaller, suggesting task-dependent complementarity. DSP

Size	Model Configuration	R-1	R-2	R-L	Faith.	Cove.	Cohe.
1B	Baseline	39.31	14.67	30.92	27.20	10.91	29.06
	Ours	**43.25**	**19.25**	**35.26**	**47.51**	**15.04**	**43.18**
	w/o UCB-guided Decoding	40.71	17.05	33.79	25.69	8.41	21.39
	w/o DSP	41.69	17.83	33.93	43.60	12.54	37.80
3B	Baseline	41.81	16.48	33.08	41.82	16.08	36.48
	Ours	**45.20**	**21.06**	**37.22**	**63.68**	**19.28**	**52.22**
	w/o UCB-guided Decoding	43.56	18.92	35.61	44.03	14.96	35.12
	w/o DSP	44.59	20.24	36.53	60.84	17.90	48.28
8B	Baseline	39.78	15.55	32.42	50.92	13.54	36.60
	Ours	**45.03**	**20.57**	**36.95**	**66.18**	**18.96**	**51.02**
	w/o UCB-guided Decoding	41.96	17.79	34.59	57.12	14.75	41.37
	w/o DSP	44.14	19.81	36.14	63.02	17.95	46.92

Table 4: Performance across model sizes on the XSum dataset. The 1B, 3B, and 8B diffusion models are initialized from the parameters of AR models: `meta-llama/Llama-3.2-1B`, `meta-llama/Llama-3.2-3B`, and `meta-llama/Llama-3.1-8B`, respectively.

	XSum		MSNews		Squad		MSQG	
Self-Prediction Depth (K)	**R-L**	**Faith.**	**R-L**	**Faith.**	**R-L**	**Cons.**	**R-L**	**Cons.**
1 (Standard Training)	34.04	44.11	47.90	69.70	45.71	75.65	40.89	81.29
2	35.05	47.37	48.63	**70.23**	**46.16**	76.17	**41.02**	**81.54**
3	**35.41**	**48.04**	48.65	69.50	45.99	**76.73**	40.74	81.00
4	33.92	41.59	**48.77**	68.73	**46.16**	75.43	40.95	80.71
5	33.93	39.34	47.44	65.39	45.31	70.76	39.63	76.76

Table 5: Effect of self-prediction depth (K) on the performance on the validation set of XSum, MSNews, Squad, and MSQG datasets.

UCB	Value	R-L	Faith.	Cove.	Cohe.
✗	Least conf.	34.96	45.13	13.79	38.05
	Entropy	34.72	42.53	13.15	35.78
	Margin	35.12	45.81	14.37	40.04
✓	Least conf.	35.11	**48.53**	**15.07**	**44.36**
	Entropy	34.75	43.32	13.49	36.74
	Margin	**35.26**	47.51	15.04	43.18

Table 6: Ablation results for different settings of the UCB-guided decoding algorithm. Each configuration varies along two dimensions: whether the UCB exploration term is used, and which value heuristic (V_i) is applied for token selection.

and UCB are complementary components of a unified framework. DSP alone yields limited gains due to poorly calibrated uncertainty and cannot influence inference-time search (e.g., 27.2 $\rightarrow$ 25.7 on XSum), whereas UCB leverages calibrated uncertainty to balance exploration and exploitation, enabling the largest improvements when combined.

4.3 Qualitative Analysis of UCB-guided Decoding

Figure 2 illustrates the token-level re-masking patterns across diffusion steps under three different strategies. In the baseline (Random) setting, the positions of masked tokens (`<M>`) vary irregularly between steps, indicating that re-masking occurs without regard to token confidence. In the confidence-guided (Uncertainty) setting, the re-masked regions gradually become concentrated on a limited subset of tokens, while highly confident tokens remain unchanged across steps. In the UCB-guided setting, the re-masking pattern alternates between uncertain and previously confident regions, showing that some tokens are revisited even after being confidently predicted once.

This strategic exploration enables the model to fix subtle but crucial errors—e.g., refining "A new version" to the more accurate "An early version" to match the reference. The benefit of this improved search appears in the LLM-as-a-Judge scores: our UCB-guided approach attains a perfect Faithfulness score of 5 and a high Coherence score of 4, outperforming both the random baseline and the confidence-only method. This example illustrates how balancing exploration and exploitation yields semantically and factually superior text.

4.4 Analysis Across Model Sizes

We observe consistent gains across all model sizes (1B, 3B, and 8B): the full model consistently out-

performs the diffusion baseline across all metrics, confirming the effectiveness of DSP and UCB-guided decoding regardless of model capacity. Faithfulness improves from 41.82 to 63.68 for the 3B model and from 41.48 to 65.92 for the 8B model. Ablation results for both 3B and 8B show that removing either component leads to a consistent performance drop, highlighting the complementary effects of DSP and UCB-guided decoding.

4.5 Analysis: How Deep Should Self-Prediction Go?

Table 5 analyzes how the self-prediction depth K in DSP training influences validation performance across XSum, MSNews, SQuAD, and MSQG. When $K = 1$, the model reduces to standard diffusion training without self-prediction. As K increases from 1, both ROUGE-L and Faithfulness/-Consistency scores generally improve, confirming that exposing the model to its own intermediate predictions strengthens its ability to recover from realistic inference-time errors. However, the optimal depth varies by dataset: for example, MSNews and SQuAD reach their peak performance at $K = 2$, whereas XSum and MSQG continue to improve up to $K = 3$. Beyond that point, performance begins to decline, suggesting that overly deep recursion may propagate noisy intermediate states and destabilize optimization. These results indicate that the ideal self-prediction depth depends on the dataset, and that dynamically adjusting the prediction depth during training could further improve generalization in more complex generation settings.

4.6 Analysis of Value Heuristics for UCB-guided Decoding

Table 6 presents an ablation study on different configurations of the UCB-guided decoding algorithm. We vary two factors: whether the UCB exploration term is activated, and which *value heuristic* (V_i) is used for token selection. Specifically, we compare three definitions of V_i: **Least confidence** ($1 - P_i^{(\text{top-1})}$), **Entropy** ($-\sum_j P_{i,j} \log P_{i,j}$), and **Margin** ($1 - (P_i^{(\text{top-1})} - P_i^{(\text{top-2})})$).

When the UCB exploration term is disabled, the decoding process reduces to a purely exploitative re-masking strategy. Among the three value heuristics, the margin-based variant already outperforms the others, achieving higher Faithfulness and Coherence scores on the XSum validation set. This indicates that the gap between the top two probabilities provides a more discriminative measure of token-level ambiguity than absolute confidence or entropy alone.

Enabling the UCB exploration term consistently improves performance across all heuristics. This improvement is most pronounced for the margin-based variant, which achieves the highest Faithfulness (47.51) and Coherence (43.18). These results show that the UCB algorithm effectively balances refining uncertain tokens and revisiting overly confident yet erroneous predictions, thereby avoiding local optima that purely exploitative strategies often fall into.

4.7 Computational Overhead and Efficiency

The Deeper Self-Prediction (DSP) objective increases training time due to additional recursive forward passes, resulting in an overhead of $1.46\times$ for $K = 2$ and $2.24\times$ for $K = 3$ compared to the baseline. Notably, this overhead is confined to the training phase. At inference time, the integration of the UCB term incurs minimal computational cost, as its calculation is significantly less demanding than a standard Transformer forward pass. Overall, while DSP requires a moderate increase in training resources, its inference-time impact remains marginal. Given the substantial gains in faithfulness, we consider this trade-off to be well-justified.

5 Conclusion

We identified the training–inference discrepancy as a fundamental weakness in discrete diffusion models for text and proposed a unified framework to address it. Our approach combines Deeper Self-Prediction (DSP) during training, which teaches the model robust self-correction, with UCB-guided Decoding at inference, which performs a principled and adaptive search of the output space. The two components complement each other effectively: DSP produces better-calibrated uncertainty estimates, while UCB-guided Decoding leverages them for more accurate and stable generation. Comprehensive experiments on summarization and question generation confirm that this combination improves factual consistency and overall coherence, paving the way for more reliable and controllable non-autoregressive text generation. As future work, we plan to extend this diffusion framework toward large-scale pre-training, aiming to develop foundation models that integrate self-correction and exploration as core training dynamics.

Limitations

While our proposed framework demonstrates consistent improvements, it is important to acknowledge its limitations. Our main experiments are conducted on a 1B-parameter backbone model, with additional experiments on 3B and 8B models limited to a subset of datasets. While this scale is sufficient to validate the core principles and effectiveness of the UCB-DSP framework against baselines, it remains an open question how these gains extend to substantially larger models (e.g., 15B, 30B, and 70B), which are increasingly prevalent in practice. Future work should therefore evaluate the proposed approach on larger-scale architectures to assess its scalability and generalizability.

Ethical Considerations

We acknowledge several ethical considerations. The improved fluency and faithfulness of our model could be misused to generate more convincing misinformation. The model inherits societal biases from its Llama-3.2-1B backbone, which this work does not mitigate. We acknowledge the use of Google's Gemini AI assistant for the sole purpose of language polishing during manuscript preparation. The authors confirm that all intellectual contributions, including the research design, experimentation, and analysis, are their own.

Acknowledgement

This paper is based on results obtained from a project, JPNP25006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We used ABCI 3.0 provided by AIST and AIST Solutions with support from "ABCI 3.0 Development Acceleration Use".

References

Masaki Asada and Makoto Miwa. 2025. Addressing the training-inference discrepancy in discrete diffusion for text generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7156–7164, Abu Dhabi, UAE. Association for Computational Linguistics.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. 2021a. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021b. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1171–1179, Cambridge, MA, USA. MIT Press.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in neural information processing systems*, volume 33, pages 6840–6851.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

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

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

T.L Lai and Herbert Robbins. 1985. Asymptotically efficient adaptive allocation rules. *Adv. Appl. Math.*, 6(1):4–22.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022a. ELMER: A non-autoregressive pre-trained language model for efficient and effective text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1044–1058, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. 2022b. Diffusion-lm improves controllable text generation. In *Advances in Neural Information Processing Systems*, volume 35, pages 3412–3425.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Zhenghao Lin, Yeyun Gong, Yelong Shen, Tong Wu, Zhihao Fan, Chen Lin, Nan Duan, and Weizhu Chen. 2023. Text generation with diffusion language models: a pre-training approach with continuous paragraph denoise. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Dayiheng Liu, Yu Yan, Yeyun Gong, Weizhen Qi, Hang Zhang, Jian Jiao, Weizhu Chen, Jie Fu, Linjun Shou, Ming Gong, Pengcheng Wang, Jiusheng Chen, Daxin Jiang, Jiancheng Lv, Ruofei Zhang, Winnie Wu, Ming Zhou, and Nan Duan. 2021. GLGE: A new general language generation evaluation benchmark. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 408–420, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

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

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 157–167, Hong Kong. Association for Computational Linguistics.

Mohammad Golam Sohrab, Masaki Asada, Matīss Rikters, and Makoto Miwa. 2024. Bert-nar-bert: A non-autoregressive pre-trained sequence-to-sequence model leveraging bert checkpoints. *IEEE Access*, 12:23–33.

Zecheng Tang, Pinzheng Wang, Keyan Zhou, Juntao Li, Ziqiang Cao, and Min Zhang. 2023. Can diffusion model achieve better performance in text generation ? bridging the gap between training and inference ! In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11359–11386, Toronto, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Xulu Zhang, Xiaoyong Wei, Jinlin Wu, Jiaxin Wu, Zhaoxiang Zhang, Zhen Lei, and Qing Li. 2025. Generating on generated: An approach towards self-evolving diffusion models. *arXiv preprint arXiv:2502.09963*.

Kun Zhou, Yifan Li, Xin Zhao, and Ji-Rong Wen. 2024. Diffusion-NAT: Self-prompting discrete diffusion for non-autoregressive text generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1438–1451, St. Julian's, Malta. Association for Computational Linguistics.

A Dataset Statistics

We describe the data and task settings of the benchmark tasks, including text summarization and question generation tasks.

Text Summarization Text summarization aims to produce a concise version of a document while preserving its salient information. We evaluate our model on the BBC Extreme Summarization (XSum) (Narayan et al., 2018) dataset and the MSNews (Liu et al., 2021) dataset. ROUGE [3] score and LLM-as-a-Judge scores are used for evaluation.

Question Generation Question generation aims to generate questions based on given passages and answers. We use the SQuAD (Rajpurkar et al., 2016) and MSQG (Liu et al., 2021) datasets. ROUGE, BLEU [4], METEOR (Denkowski and Lavie, 2014), and LLM-as-a-Judge scores are used for evaluation.

Math Reasoning We use the GSM8k dataset (Cobbe et al., 2021), which requires multi-step reasoning. Critically, the model is required to output the derivation process and the final answer separated by a special token (e.g., `####`), and only the final answer is used to evaluate accuracy. For example, the expected output format is `<<60/100*5=3>> <<16/2=8>> <<8*3=24>> <<8*5=40>> <<24+40=64>> #### 64`.

The statistics of these datasets are summarized in Table 7. We adopt the same data-splitting settings and evaluation metrics as in previous studies (Li et al., 2022a; Zhou et al., 2024), and we report the baseline scores from Asada and Miwa (2025) for a fair comparison.

Task	Dataset	#Train	#Valid	#Test
Sum.	XSum	204,045	11,332	11,334
	MSNews	136,082	7,496	7,562
QG	SQuAD	75,722	10,570	11,877
	MSQG	198,058	11,008	11,022
Math	GSM8K	384,627	500	1,319

Table 7: Statistics of the four datasets. "Sum." and "QG" stand for text summarization and question generation, respectively.

[3] https://github.com/google-research/google-research/tree/master/rouge

[4] https://huggingface.co/spaces/evaluate-metric/sacrebleu

B Hyperparameter Settings

Key hyperparameters for our experiments are listed in Table 8. We used the AdamW optimizer with a linear warmup and cosine decay schedule for the learning rate. All experiments were conducted using eight NVIDIA H200 GPUs, and we report results from single runs.

Hyperparameter	Value
Backbone Model	Llama-3.2-1B
Peak Learning Rate	1e-4
Total Batch Size	128
Max Epochs	3 / 6 / 20
Warmup Ratio	0.05
Weight Decay	0.001
Diffusion Steps	32
Inference Diffusion Steps	16
Max New Tokens	64
UCB Exploration (C)	1.5

Table 8: Main hyperparameter settings. The maximum number of epochs was set to 3 for the XSum, MSNews, and MSQG datasets, and 6 for the SQuAD dataset, and 20 for the GSM8K dataset.

C LLM-as-a-Judge Evaluation Protocol

To complement automatic metrics such as ROUGE, BLEU, and METEOR, we employed a LLM-as-a-Judge evaluation framework to assess the human-perceived quality of model outputs. This approach leverages GPT-4.1-mini as an expert evaluator that provides fine-grained, aspect-based judgments for both summarization and question generation tasks. Each evaluation prompt explicitly includes the source document, the human reference, and the candidate output produced by the model.

C.1 Summarization Evaluation

For summarization, the evaluator compares the candidate summary against the reference summary while considering the source document as context. The evaluation covers three aspects:

- **Faithfulness:** factual consistency with the source document; penalizes hallucinated or fabricated content.
- **Coverage:** completeness in capturing the main ideas and key information of the source.
- **Coherence:** structural and linguistic quality, including logical flow and readability.

Each aspect is rated on a 1–5 scale (*1 = Poor, 3 = Fair, 5 = Excellent*), followed by a short overall explanation (1–3 sentences). The prompt used for the summarization tasks in our experiments is shown in Listing 1.

C.2 Question Generation Evaluation

For question generation (QG), the evaluator receives the *source passage*, the *given answer*, the *reference question*, and the *candidate question*. It assesses the candidate along three dimensions:

- **Consistency:** factual and contextual consistency with the source passage, ensuring no unsupported content.
- **Answer Consistency:** semantic alignment between the candidate question and the provided answer.
- **Coherence:** grammaticality, clarity, and fluency of the generated question.

The prompt used for the question generation tasks in our experiments is shown in Listing 2.

C.3 Implementation Details

All evaluations were conducted via the OpenAI API using `gpt-4.1-mini`. For each dataset, scores are averaged across all samples and linearly scaled to a 0–100 range for reporting.

```
EVAL_SUM_SYSTEM = """You are an expert summarization evaluator.
Your task is to judge the quality of a candidate summary generated by a model,
by comparing it to the reference summary written by a human, and considering the
    original document as the source.

Please evaluate the candidate summary along the following aspects:

Faithfulness: Is the candidate summary factually consistent with the source document
    ? Are there any hallucinations or fabrications?

Coverage: Does the candidate summary capture the most important information from the
     source?
Does it include the key points that are also reflected in the reference summary?

Coherence: Is the candidate summary well-structured, logically organized, and easy
    to read?
Does it flow smoothly from one idea to the next?

For your output:
1. Provide one short overall explanation (1-3 sentences) summarizing your evaluation
    .
2. Then, give a score from 1 to 5 for each aspect, where:
   1 = Poor
   3 = Fair
   5 = Excellent

Please follow the exact output format below:

Overall Explanation: <your explanation>

Faithfulness: <1-5>
Coverage: <1-5>
Coherence: <1-5>
"""

EVAL_SUM_USER = """Source Document:
((SOURCE))

Reference Summary:
((REF))

Candidate Summary:
((CAND))
"""
```

Listing 1: LLM-as-a-Judge prompt for text summarization tasks

```
EVAL_QG_SYSTEM = """You are an expert evaluator for question generation (QG) tasks.
Your goal is to assess the quality of a candidate question generated by a model,
by comparing it with a human-written reference question and considering both the
    source passage and the given answer.

Please evaluate the candidate question along the following aspects:

Consistency (Cons.): Is the candidate question consistent with the source passage?
Does it avoid introducing contradictions, hallucinated facts, or information not
    supported by the passage?

Answer Consistency (AnsC.): Can the given answer correctly and completely answer the
     candidate question?
Does the question align semantically and contextually with the provided answer?

Coherence: Is the candidate question grammatically fluent, logically sound, and easy
     to understand?
Does it read naturally as a well-formed question?

For your output:
1. Provide one short overall explanation (1-3 sentences) summarizing your evaluation
    .
2. Then, give a score from 1 to 5 for each aspect, where:
   1 = Poor
   3 = Fair
   5 = Excellent

Please follow the exact output format below:

Overall Explanation: <your explanation>

Consistency: <1-5>
Answer Consistency: <1-5>
Coherence: <1-5>
"""

EVAL_QG_USER = """Answer and Source Passage:
((SOURCE))

Reference Question:
((REF))

Candidate Question:
((CAND))
"""
```

Listing 2: LLM-as-a-Judge prompt for question generation tasks