

Dimple: Discrete Diffusion Parallel Generation for Large Multimodal Modal

Runpeng Yu Xinyin Ma Xinchao Wang*

National University of Singapore

{r.yu, maxinyin}@u.nus.edu xinchao@nus.edu.sg

Abstract

This paper introduces Dimple and Dimple+, two discrete diffusion multimodal large language models (dMLLMs). Dimple is derived from a discrete diffusion large language model (dLLM) that initially lacks multimodal understanding and acquires this capability through a hybrid training strategy that begins with autoregressive learning and subsequently transitions to discrete diffusion training. In contrast, Dimple+ is initialized from an autoregressive multimodal large language model and attains parallel decoding ability via purely discrete diffusion training. Both models achieve performance on par with their autoregressive counterparts, while Dimple+ further sets a new state of the art among existing dMLLMs. To improve inference efficiency, we propose Confident Decoding, a method that dynamically adapts the number of tokens generated at each iteration. Experimental results demonstrate decoding speedups of $2\times$ – $6\times$ with only marginal performance loss. Additionally, we show that the Prefilling technique, previously developed for autoregressive models, can be effectively extended to dMLLMs with bidirectional attention, yielding nearly lossless acceleration of $1.7\times$ – $7\times$. Finally, we present the Structure Prior method, which enables fine-grained control over output format and reasoning structure, a capability that is challenging to achieve in autoregressive models.

1 Introduction

Recent months have witnessed a surge of interest in applying diffusion models to natural language processing tasks. These Discrete Diffusion Large Language Models (dLLMs) reframe generation as a denoising process, allowing parallel decoding and improved control over output structure. The first general framework for diffusion over discrete

spaces was introduced in D3PM (Campbell et al., 2024), which set up the mathematical foundation of the diffusion language model. Subsequently, RDM (Zheng et al., 2024), MD4 (Shi et al., 2025a), and MDLM (Sahoo et al., 2024a) reparameterized and simplified the loss for discrete diffusion into a weighted cross-entropy form, which was widely adopted in later researches (Nie et al., 2025; Ye et al., 2025b; Gong et al., 2025; Zhu et al., 2025). Other explorations include extending discrete diffusion models to continuous time (Campbell et al., 2022), discrete flow matching (Gat et al., 2024) and concrete score matching (Meng et al., 2022; Sun et al., 2023). However, these works were typically conducted on models under 2B parameters. Only more recently, LLaDA (Nie et al., 2025), LLaDA 1.5 (Zhu et al., 2025), Diffu-GPT/LLaMA (Gong et al., 2025), and Dream (Ye et al., 2025b) successfully scaled up dLLMs to much larger sizes.

Concurrently, the field of multimodal large language models (MLLMs) has advanced rapidly, with models such as LLaVA (Li et al., 2024a), Qwen-VL (Bai et al., 2025), and InternVL (Mohbat and Zaki, 2024) achieving state-of-the-art results on a range of vision-language benchmarks. However, most of the existing MLLMs rely exclusively on autoregressive generation mechanisms.

In this work, we introduce **Dimple** and **Dimple+**, two *Discrete Diffusion Multimodal Large Language Models* (dMLLMs). Due to the inefficiency of discrete diffusion training in learning multimodal understanding ability, Dimple and Dimple+ utilize a “first-autoregressive-then-diffusion” training strategy and “autoregressive model initialization” strategy, respectively. Dimple begins with Dream (Ye et al., 2025b), a dLLM lacking multimodal understanding ability, and first undergoes autoregressive training followed by the diffusion training. Ablation studies in the supplementary material show that this hybrid approach is significantly more effective for learning multimodal un-

*Corresponding Author: Xinchao Wang (Email: xinchao@nus.edu.sg)

derstanding ability than the pure diffusion training. In contrast, Dimple+ is initialized from Qwen2.5-VL (Bai et al., 2025), an MLLM with strong multimodal understanding ability but without discrete diffusion-based parallel decoding ability. By applying only discrete diffusion training, Dimple+ acquires the parallel decoding capability while inheriting Qwen2.5-VL’s strong multimodal understanding ability, achieving the SOTA performance among dMLLMs.

To further improve inference efficiency and controllability, we propose **Confident Decoding**, which dynamically adjusts the number of tokens decoded per iteration based on a confidence threshold. This technique accelerates decoding by approximately 2× to 6× with only minor performance degradation. We also demonstrate the feasibility of applying the **Prefilling** technique in autoregressive models to dMLLMs with bidirectional attention. Prefilling achieves nearly lossless speedups of 1.7× to 7× on most benchmarks. In addition, we explore the **Structure Prior** technique, which enables fine-grained control over response structure. With Structure Prior, dMLLMs can perform early answer prediction, structured reasoning, and in-place output formatting, which are difficult to achieve with autoregressive models.

Contributions. Our key contributions are as follows:

- We introduce **Dimple** and **Dimple+**, both of which achieve performance comparable to their autoregressive baselines under the same training budget. Moreover, Dimple+ establishes state-of-the-art results among dMLLMs.
- We propose **Confident Decoding**, reimplement **Prefilling**, and explore **Structure Prior** for dMLLM inference that improve the efficiency, flexibility, and controllability.

2 Preliminary

Discrete Diffusion Large Language Models (dLLMs). dLLMs are a class of generative models that conceptualize text generation as a denoising process over discrete time steps. Let $x_0 \sim p_{\text{data}}(x_0)$ denote the original token sequence, and x_t its noisy version at time $t \in [0, T]$. The forward noising process is defined as a Markov chain $q(x_{1:T}|x_0) = \prod_{t=1}^T q(x_t|x_{t-1})$, progressively adding noise to x_0 . For discrete token modeling with an absorbing state, this is formalized as:

$$\begin{aligned} q(x_t|x_{t-1}) &= \text{Cat}(x_t; Q_t^\top x_{t-1}), \\ q(x_t|x_0) &= \alpha_t x_0 + (1 - \alpha_t) \mathbf{m}, \\ \alpha_t &= \prod_{i=1}^t (1 - \beta_i), \end{aligned} \quad (1)$$

where $Q_t = (1 - \beta_t)I + \beta_t \mathbf{1m}^\top$, with $\mathbf{m}$ being the one-hot representation of a special [MASK] token. The reverse (generative) process is modeled as:

$$\begin{aligned} p_\theta(x_{0:T}) &= p_\theta(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t) \\ &= \prod_{t=1}^T q(x_{t-1}|x_0) p_\theta(x_0|x_t), \end{aligned} \quad (2)$$

where each transition $p_\theta(x_{t-1}|x_t)$ is learned to approximate the reverse of the noising process with bidirectional attention. The training loss of Discrete Diffusion, derived from the variational lower bound of $p(x_0)$, is a weighted cross-entropy loss:

$$\begin{aligned} \mathcal{L}_D &= \mathbb{E}_t[\mathcal{L}_t], \\ \mathcal{L}_t &= \mathbb{E}_{q(x_t|x_0)} \left[- \sum_{n=1}^N \frac{1}{t} \delta_{x_t^n, \mathbf{m}}(x_0^n) \log[f_\theta(x_t)^n] \right] \end{aligned} \quad (3)$$

This loss can be implemented as a Masked Language Modeling loss with a random mask ratio. The mask ratio is function of the time index t , which is a random variable. The expectation $\mathbb{E}_t$ is taken over a uniform distribution of time steps $t \in [0, 1]$. In practice, for each training instance, one time index t is samples to determine the proportion of tokens to be masked. The inner expectation $\mathbb{E}_{q(x_t|x_0)}$ is taken over the forward transition probability defined in Eq. 1. N denotes the number of tokens in the current input sample x_0 . Inside the summation $\sum_{n=1}^N$, a weighted cross-entropy is computed over each token. The weights consist of two parts: (1) a time-dependent factor $\frac{1}{t}$, and (2) an indicator function $\delta_{x_t^n, \mathbf{m}}$ which ensures that the loss is only computed for masked tokens. Finally, f_θ denotes the model’s prediction.

Challenges of Discrete Diffusion Training. Due to the use of bidirectional attention and the limited number of sampled timesteps, training with Discrete Diffusion directly incurs higher computational costs compared to autoregressive (AR) training (Prabhudesai et al., 2025; Yu et al., 2025).

1. Compared to next-token prediction loss, masked language modeling has lower utilization of the training corpus. In Auto-Regressive training, every response token receives supervision. In contrast, in discrete diffusion training, only tokens where $\delta_{x_t^n, m} = 1$ (i.e., masked tokens) are used to compute the loss. Take linear masking scheduling as an example, with $t \sim \text{Uniform}(0, 1)$, only about half the tokens are supervised.
2. In diffusion training, the supervision signal does not cover the entire generation trajectory. Discrete diffusion inference proceeds iteratively as $x_T \rightarrow x_{T-1} \rightarrow \dots \rightarrow x_0$, but during training each sample is trained on only a single, or at most a few, randomly sampled steps: $x_t \rightarrow x_{t-1}$. Consequently, it is impossible to cover the complete generation trajectory for each sample. In contrast, in autoregressive training, next-token prediction with causal attention ensures that the supervision signal covers every step of the generation process for each sample.

These two challenges result in discrete diffusion models performing worse than autoregressive models **under the same training budget**. Empirical evidence is presented in Sec. 4.

3 Methodology

3.1 Training

We adopt two strategies to mitigate the limitations of discrete diffusion training. The first applies autoregressive training prior to discrete diffusion training to enhance the model’s learning of multimodal understanding ability. The second initializes the model with the weights of an autoregressive MLLM, followed by discrete diffusion training to enable the acquisition of parallel decoding capabilities. Details of the training configuration, masking strategies, and sample padding are provided in the supplementary material.

Strategy I (Dimple): Starting from a Diffusion Large Language Model We denote the model obtained using Strategy I as Dimple. We initialize the model with Qwen2.5-VL’s vision encoder (Bai et al., 2025), DLM Dream (Ye et al., 2025b), and a randomly initialized two-layer projector (Liu et al., 2023a). Following the alignment and instruction-tuning paradigm of LLaVA (Liu et al., 2023b), we design a three-stage training pipeline: Autoregressive Alignment (AA), Autoregressive Instruc-

tion Tuning (AT), and Diffusion Instruction Tuning (DT).

In AA and AT, we modify Dream’s attention mechanism by replacing the full attention mask with a causal attention mask and applying a next-token prediction loss. We adopt the LLaVA-NEXT (Liu et al., 2024) training recipe and its datasets. During AA, only the projector is trainable; during AT, all components except the vision encoder are trainable. In the experiments reported in Sec. 4, we refer to the model obtained after AA as the aligned model. In the DT phase, we transition back to the diffusion-based generation framework. The full attention mask in Dream is restored, and the loss defined in Equation (3) is applied. The LLaVA-NEXT instruction-following dataset is reused for this phase. All components except the vision encoder are trainable.

Strategy II (Dimple+): Starting from a Autoregressive Multimodal Large Language Model

We denote the model obtained using Strategy II as Dimple+. For initialization, we adopt the full Qwen2.5-VL model (Bai et al., 2025). Training consists of a single Diffusion Instruction Tuning (DT) phase. Specifically, we modify the attention mask in Qwen2.5-VL to full attention and apply the loss defined in Equation (3), updating all components except the vision encoder. The training data consists of the instruction-following dataset from LLaVA-NEXT (Liu et al., 2024) and 1.3M purely text-based instruction-following samples collected from the MAMmoTH-VL (Guo et al., 2025) dataset. Since Qwen2.5-VL (Bai et al., 2025) has already undergone autoregressive alignment and autoregressive instruction tuning, this implementation actually delegates the AA and AT phases to Qwen2.5-VL.

3.2 Inference Techniques

Review: Confidence-Based Decoding. In discrete diffusion models, the generation process proceeds iteratively over a fixed number of steps. At each step, a subset of masked positions is selected and updated based on their confidence scores. Let $\mathbf{x}_t$ denote the input sequence at step t . Let $\mathbf{z}_t \in \mathbb{R}^{L \times V}$ be the corresponding logits, where V is the vocabulary size. Taking MaskGIT’s decoding algorithm (Chang et al., 2022) as an example,

the decoding process is as follows.

Probabilities: $\mathbf{p}_t = \text{softmax}(\mathbf{z}_t/\tau)$;

Confidence: $c_t^{(i)} = \max(p_t^{(i)})$;

Select K positions: $\mathcal{I}_t = \text{TopK}(c_t, K)$;

Sample tokens $x_t^{(i)} \sim \text{Categorical}(p_t^{(i)})$ for $i \in \mathcal{I}_t$.

This approach ensures that tokens with higher prediction confidence are updated earlier.

Confident Decoding. In previous confidence-based decoding (Chang et al., 2022; Ye et al., 2025b; Nie et al., 2025), the number of tokens decoded per step is fixed. However, we argue that decoding should adapt to the semantic structure of the text: some steps may allow many tokens to be confidently predicted, while others may necessitate more caution. We therefore propose **Confident Decoding**, which dynamically adjusts the number of tokens updated per step based on a fixed confidence threshold $\gamma \in (0, 1)$. Formally, at each step t , we define $\mathcal{I}_t = \{i \mid c_t^{(i)} \geq \gamma\}$, where $c_t^{(i)}$ is the confidence score for position i . If $\mathcal{I}_t$ is non-empty, all tokens at positions $\mathcal{I}_t$ are decoded. Otherwise, we decode the token at the position with the highest confidence. This method enables:

- decoding multiple tokens simultaneously when model is highly confident, improving efficiency;
- avoiding low-confidence updates, preserving generation quality.

This adaptive mechanism reduces unnecessary updates of uncertain tokens and enhances both quality and speed of generation.

Prefilling. Let L_{prompt} denote the lengths of the prompt, which includes the question, image, and system prompts. Let L_{answer} denote the length of the response. With vision tokens, the prompt can become unignorable long. The use of a full attention mask in diffusion decoding results in a quadratic complexity $\mathcal{O}((L_{\text{prompt}} + L_{\text{answer}})^2)$ per decoding step. To alleviate this cost, we re-implement the **Prefilling** strategy in the autoregressive models, which saves the key-value pairs of the prompt tokens after the first generation step and reuse them in the following steps, reducing the complexity to $\mathcal{O}(L_{\text{answer}}^2)$. But, due to the use of a full attention mask in dMLLM, the prefilling technique is not strictly lossless.

4 Experiments

All training configurations, evaluation metrics, generation hyperparameters used during evaluation, and other experimental details are provided in the supplementary material. The sensitivity analysis of the generation hyperparameters and additional ablation studies on the “AR-then-Diffusion” training strategy are also included in the supplementary material.

4.1 Benchmark Performance

Comparison with AR Baseline. We compare the performance of Dimple and Dimple+ with their corresponding AR baselines in Tab. 1. Each AR baseline is trained with the same data, parameter initialization, and total number of training iterations as Dimple and Dimple+, ensuring a fair comparison. As shown by the results, both Dimple and Dimple+ outperform their AR baselines. Dimple achieves an average score of 62.4%, 1.8% higher than its baseline, while Dimple+ surpasses the baseline on 9 of 12 benchmarks. These results indicate that, under the same training budget, dM-LLM can match the performance of autoregressive models on instruction-following and visual understanding tasks. We also include the performance of LLaVA-NEXT (Liu et al., 2024) and Qwen2.5-VL (Bai et al., 2025) as references. Compared with Qwen2.5-VL, Dimple+ and its AR baseline improve on some benchmarks but regress on others. On the one hand, this results from insufficient task coverage in the training data, which improves performance on some benchmarks but degrades it on others. On the other hand, it also suggests that learning for parallel decoding is not entirely orthogonal to multimodal understanding; instead, the process of learning parallel decoding affects multimodal understanding ability.

Comparison with Other dMLLMs. We compared the performance of Dimple+, LaViDA (Li et al., 2025a), and LLaDA-V (You et al., 2025) in Tab. 2. Dimple+ significantly outperforms the other two baselines on most benchmarks, achieving state-of-the-art results for dMLLM models. Moreover, training Dimple+ requires substantially lower computational cost. Specifically, Dimple+ is trained on only 2.6M samples, whereas training LLaDA-V involves more than 17M samples. This demonstrates the effectiveness of the “Autoregressive Model Initialization” strategy for dMLLMs.

Model	Dimple-7B (ours)	Dimple-7B (AR Baseline)	Dimple-7B+ (ours)	Dimple-7B+ (AR Baseline)	LLaVA-NEXT-7B	Qwen2.5-VL-7B
#Training Samples/Tokens	1.3M/0.8B	1.3M/0.8B	2.6M/1.6B	2.6M/1.6B	1.3M/-	-/2.6T
Base LLM/MLLM	Dream(Qwen2.5)	Qwen2.5	Qwen2.5 VL	Qwen2.5 VL	Vicuna-1.5	Qwen2.5
GQA	59.24	59.1	63.2	64.1	64.8	60.0
MMBench_en_test	74.6	73.9	81.5	81.2	68.7	83.5
MME_percpetion	1514	1477	1558	1535	1519	1533
MME_cognition	432	457	627	610	332	620
MME_total	1946	1934	2185	2145	1851	2153
POPE	86.2	86.3	87.7	86.9	86.7	87.9
MMMU_val	45.2	45	50.2	49.1	35.8	58.6
SQA_img	77.1	77.2	79.1	83.6	72.8	52.1
AI2D	74.4	73.6	82.4	80	65.4	83.9
ChartQA	63.4	58.5	74.7	74.8	54.9	87.3
TextQA_eval	61.6	53.6	65.4	64.1	64.8	77.8
OCRBench	565	486	699	674	490	783
MathVista_test_mini	42.3	42.6	61.0	59.0	33.0	68.2
MMVet	41.2	37.8	47.5	49.2	47.3	67.1
Average	62.4	60.6	70.7	70.2	58.5	73.8

Table 1: Benchmark Performance Comparison. Dimple outperforms autoregressive baselines trained on similar data scales. In the table, the highlighted values in each row indicate the best-performing results among the following four models: Dimple, Dimple (AR Baseline), Dimple+, and Dimple+ (AR Baseline).

Model	MME_p	MME_c	MME total	MMMU	AI2D	ChartQA	TextQA	MathVista
Dimple+ (ours)	1558	627	2185	58.2	82.4	76.7	65.4	61.0
LaViDA-L	1366	341	1707	43.3	70.0	64.6	56.3	44.8
LaViDA-D	1463	389	1852	42.6	59.0	61.0	57.1	42.1
LLaDA-V	1507	491	1998	48.6	77.8	78.3	65.3	59.7

Table 2: Performance comparison with other Discrete Diffusion MLLMs. Our Dimple+ outperforms other models with a clear Margin on most of the benchmarks. In the table, MME_p, MME_c, and MathVista are abbreviations for MME_perception, MME_cognition, and MathVista_test_mini, respectively.

4.2 Ablation on Prefilling

The following experiments investigate the impact of the prefilling technique on both performance and generation speed in dMLLMs. We conducted experiments with response lengths of 4 and 8, evaluating benchmark performance with and without prefilling. As shown in Tab. 3, prefilling does not lead to significant performance degradation on most datasets, with the average performance drop across all benchmarks being only 0.8%. This indicates that, in the current model, visual perception and the utilization of image tokens remain largely unchanged during text generation. Next, we examined the effect of prefilling on generation speed. By adjusting the batch size to 1 and 32, we simulated scenarios of low and high GPU utilization, respectively. The results show that: (1) under low GPU utilization (batch size = 1), prefilling yields an average speedup of 1.79 $\times$; (2) under high GPU utilization (batch size = 32), the acceleration effect is more pronounced, reaching up to 7 $\times$.

4.3 Ablation on Confident Decoding

We evaluate the Confident Decoding from two perspectives: *efficiency* and *performance*. Confident

Decoding is designed to reduce the number of inference steps, thereby accelerating generation. The last row in Tab. 4 shows the actual number of decoding steps k under a fixed response length of 8, along with the corresponding speedup factors (in parentheses). As shown, Confident Decoding reduces k to 1.25 2.87, resulting in approximately 3 $\times$ to 6 $\times$ speedup.

Next, we compare Confident Decoding with previous decoding strategies (Ye et al., 2025b; Nie et al., 2025): MaskGIT, Entropy, and Random. Unlike these methods, Confident Decoding is the first algorithm capable of dynamically adjusting the number of decoding steps, whereas the others require a predefined fixed value of k . For fairness, we set the number of decoding steps for the other strategies to either the response length (8) or one-quarter of it (2). As shown in Tab. 4, reducing the decoding steps to 2 leads to a substantial performance drop for other methods—up to 10% on CharQA for MaskGIT and Entropy. In contrast, Confident Decoding achieves performance comparable to the 8-step baselines with only about 2 steps. These results demonstrate that Confident Decoding enables faster inference without sacrificing

R.L.	Prefilling	Metric	B.S.	SQA	MMMU	MME	POPE	ChartQA	AI2D
4	×	ACC	1 or 32	84.4	43.4	1946	85.0	51.7	74.5
		TPS	1	34.8	21.3	22.7	22.4	21.5	22.9
			32	45.9	35.5	37.0	36.9	36.3	37.2
	✓	ACC	1 or 32	84.4	43.4	1822	85.1	51.3	74.4
		TPS	1	49.3 ($\times 1.32$)	19.9 ($\times 1.67$)	21.3 ($\times 1.63$)	20.7 ($\times 1.65$)	20.1 ($\times 1.69$)	21.5 ($\times 1.62$)
			32	168.5 ($\times 3.42$)	62.8 ($\times 3.16$)	77.6 ($\times 3.64$)	78.2 ($\times 3.78$)	74.4 ($\times 3.70$)	79.7 ($\times 3.71$)
8	×	ACC	1 or 32	84.1	45.3	1900	84.8	58.4	74.5
		TPS	1	29.2	19.3	19.3	19.5	19.1	19.7
			32	44.9	39.7	40.6	40.4	40.4	40.7
	✓	ACC	1 or 32	84.0	45.2	1905	84.0	58.4	74.2
		TPS	1	49.2 ($\times 1.54$)	18.1 ($\times 2.06$)	21.4 ($\times 2.10$)	20.7 ($\times 2.07$)	20.0 ($\times 2.12$)	21.6 ($\times 2.07$)
			32	288.2 ($\times 5.86$)	116.5 ($\times 6.44$)	145.1 ($\times 6.80$)	147.4 ($\times 7.12$)	138.9 ($\times 6.95$)	147.9 ($\times 6.85$)

Table 3: Ablation Study on the Prefilling Technique. We compared model performance (ACC) and inference speed (Tokens Per Second, TPS) with and without the use of prefilling. Performance-related rows are highlighted in blue. The underlined values indicate the speedup factor achieved with prefilling. “R.L.” and “B.S.” denote Response Length and Batch Size, respectively. The results demonstrate that although, in theory, prefilling is not lossless in dMLLMs, it causes minor drops in performance on most benchmarks while delivering substantial improvements in inference speed.

	Steps	CharQA	SQA	POPE	AI2D
MaskGit	8	63.0	77.2	86.1	72.9
	2	52.2	76.2	85.5	71.1
Entropy	8	63.0	77.2	86.0	73.0
	2	54.9	76.5	85.1	72.9
Random	8	59.3	76.1	86.0	72.8
	2	57.1	76.0	85.2	71.0
Confident Decoding	Acc	62.5	77.1	86.4	73.1
	Steps	2.87 ($\times 2.8$)	1.41 ($\times 5.7$)	1.25 ($\times 6.4$)	1.46 ($\times 5.5$)

Table 4: Comparison of confident decoding with other decoding algorithms. Confident decoding significantly reduces the number of decoding steps while maintaining nearly the same performance, thereby improving decoding efficiency.

accuracy.

5 Generation Behaviors

In this section, we provide a showcase to demonstrate that dMLLMs exhibit fundamentally different decoding behaviors compared to autoregressive models. Unlike autoregressive models, discrete diffusion models offer the unique capability of **explicitly controlling the content** generated at arbitrary positions. This allows us to specify certain tokens as fixed in the final answer prior to generation. We refer to such tokens as **structure priors**.

We present three illustrative examples. Each includes the input image, the question, and the generated response from Dimple. For each case, we annotate: the **response length** (i.e., the total number of tokens in the final answer), the **number of remaining tokens** that need to be generated after applying the structure prior, and the **actual num-**

ber of decoding steps (actual iteration) used in generation. Furthermore, we visualize the **generation history**, which records at which iteration each token was generated. Tokens provided by structure priors are marked with “-”, and special tokens (e.g., padding) are excluded from the generation history but counted in the actual number of decoding steps. To enhance interpretability, we colored the response: tokens decoded earlier are shown in blue, while those generated later appear in red. Tokens with similar decoding steps share similar colors. All examples leverage the Prefilling technique and proposed Confident Decoding strategy.

5.1 Example 1 (Tab. 5): Structured Reasoning and Early Answering

In this example, the input consists of two images, and the model is asked to identify the common object present in both. We inject three structure priors into the expected answer: “In the first image, there ”, “In the second image, there ” and “The common item in the two images is”. These priors guide the model to follow a reasoning trajectory resembling structured thought. Unlike the chain-of-thought (CoT) strategies used in autoregressive models, which rely on indirect guidance, structure prior enables direct and precise control over intermediate reasoning steps.

Remarkably, the correct answer “scissors” is decoded at the 10th iteration, prior to the completion of the full response. The subsequent steps focus on completing the thinking trajectory. This illustrates

Question: What is the common item in the two images?

Response: In the first image, there is a pair of black scissors. In the second image, there is a pair of black scissors and a pink plate with a pig face on it. The common item in the two images is the scissors.

Response Length / #Remaining Tokens / Actual Iterations:

64/42/30



Generation History:

In	the	first	image	,	there		is	a	pair	of	black	scissors	.		In	the
-	-	-	-	-	-	-	2	1	4	5	16	5	3	4	-	-
second	image	,	there		is	a	pair	of	black	scissors	and	a	pink	plate		
-	-	-	-	-	2	2	13	12	15	14	17	18	19	23		
with	a	pig	's	face	on	it	.	The	common	item	in	the	two	images	is	
22	24	25	26	27	20	21	4	5	-	-	-	-	-	-	-	-
the	scissors	.														
11	10	9														

Table 5: Example: Structured Reasoning and Early Answering.

Question: Extract the information in the image.

Response: {"Date": "Apr 18, 2018", "Time": "12:00 AM",}

Response Length/#Remaining Tokens/Actual Iterations: 32/22/7

Generation History:

{	"	Date	:	"	Apr	18	,	2018	,	"	Time	:	"	12:00	AM	",	}
-	-	-	-	-	5	1	1	1	1	1	1	1	2	1	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	3	3	4	3	4
-	-	-	-	-	-	-	-	-	-	-	-	-	6	7	-	-	-

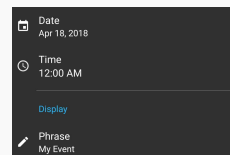


Table 6: Example: Structured output and automatic semantic clustering during the generation process.

a key strength of diffusion models: they can arrive at the final answer earlier in the generation process. In contrast, autoregressive models must complete the full sequence before producing the answer.

5.2 Example 2 (Tab. 6): Structured Output

In this example, the task involves extracting visual information from an image. The question does not specify the desired output format and the attributes required. All of these are indicated by the structure prior. In this example, the structure prior is “{date:”, “time:” and ”}”. These priors define a JSON-like response layout, and the model successfully generates structured content accordingly.

This example also provides direct evidence of the effectiveness of **Confident Decoding**. The number of tokens decoded per iteration varies dynamically. For instance, 9 tokens corresponding to the date and year are decoded simultaneously in a single step. Despite a total of 22 tokens to be generated, the model completes generation in only 7 iterations, one-third of the total number of tokens. This parallelism significantly accelerates inference.

5.3 Example 3 (Tab. 7): Length Control

Controlling output length is difficult in autoregressive generation. However, discrete diffusion models inherently support this capability. In this case, we specify structure priors at the end of the sequence. Specifically, we force tokens at positions $[-12 : -4]$ to be: “Thus, the answer is \box{””. This allows us to strictly control where the model concludes its generation and places the final answer. We demonstrate two configurations with response lengths of 16 and 32. In each case, the model adjusts its reasoning span to fill the available token budget appropriately.

6 Related Work

In the supplementary material, we also discuss the differences between Dimple and other approaches based on parallel decoding and multimodal perception models.

The first problem when introducing diffusion process into language decoding, is the choice of a continuous diffusion space or discrete one. There are explorations in the continuous spaces. DiffuSeq (Yuan et al., 2023) use a partial noising and con-

Question: How many objects are preferred by more than 90 percent of people in at least one category? Please think step by step.

Response: Nothing is preferred by more than 90 percent of people in any category. Thus, the answer is \box{Zero}.

Response Length/#Remaining Tokens/Actual Iterations:
32/23/16

Generation History:

Nothing	is	preferred	by	more	than		9	0	percent	of	people	in	any	category	.	Thus
17	16	15	14	13	12	11	10	10	10	9	8	7	6	5	3	-
,	the	answer	is		box	{	Zero	}	.							
-	-	-	-	-	-	-	4	2								

Response: All of the objects in the chart have a maximum of 90 percent, but no object actually reaches 90. Therefore, none of the objects are preferred by more than 90 percent of people in at least one category. Thus, the answer is \box{Zero}.

Response Length/#Remaining Tokens/Actual Iterations: 64/55/37

Generation History:

All	of	the	objects	in	the	chart	have	a	maximum	of		9	0	percent	,	but
20	17	15	16	18	19	38	21	22	26	23	25	28	24	27	29	33
no	object	actually	reaches		9	0	.	Therefore	,	none	of	the	objects	are	preferred	
34	35	36	37	31	32	30	13	14	12	11	11	10	8	7	6	
by	more	than	9	0	percent	of	people	in	at	least	one	category	.	Thus	,	
6	6	6	6	6	6	6	6	5	5	5	5	4	5	9	-	-
the	answer	is		box	{	Zero	}	.								
-	-	-	-	-	-	-	3	2								

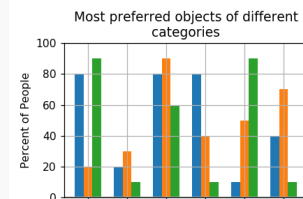


Table 7: Example: Length Control.

ditional denoising process to model sequence-to-sequence diffusion in continuous space. Argmax Flows and Multinomial Diffusion propose continuous relaxations for categorical distributions for better generation stability (Hoogeboom et al., 2021). SED (Strudel et al., 2022) directly use the embedding space for the diffusion of natural language tokens. For the exploration in the discrete space, Structured Denoising Diffusion Models (SDDM) introduce discrete diffusion tailored to linguistic structures (Austin et al., 2023); RDM (Zheng et al., 2024) reparameterizes the backward discrete diffusion into a step sampling process to provide more flexible and unified framework. For training diffusion language model, loss function is the key. MDLM (Sahoo et al., 2024a), MD4 (Shi et al., 2025b) and SEDD (Lou et al., 2024) show that both a simple weighted mask language modeling loss and a score entropy loss, which is adopted from the score matching loss, are suitable for training. In this work, we follow the thread of masked language modeling scheme for the language diffusion process. In this direction, Diffusion-LM (Xu, 2022) applies diffusion to masked language modeling to enhance controllable text generation; DiffusionBERT (Sahoo et al., 2024b) uses pretrained BERT as an initialization of the diffusion training to accelerate convergence and improve perfor-

mance. Besides using a pure diffusion language model, auto-regressive hybrids like AR-Diffusion (Li et al., 2023b) and Semi-autoregressive methods like SSD-LM (Han et al., 2023) aim to combine the benefits of diffusion and autoregressive modeling. SSD-LM also shows that with a simplex projection the diffusion process can also be conducted in the natural vocabulary space. Recently, diffusion language models have been scaled to show comparable performance with the SOTA auto-regressive model, to learn instruction following ability and to have even better planning ability (Nie et al., 2025; Gong et al., 2025; Ye et al., 2025a,b).

7 Conclusion

In this work, we introduce Discrete Diffusion Multimodal Large Language Models **Dimple** and **Dimple+**, both of which match the performance of their autoregressive baselines while offering unique advantages such as parallel decoding and structure-aware generation. The main limitation of Dimple is that Dimple-7B is trained on less than 1/20th of the data used by state-of-the-art models such as Qwen2.5-VL and Eagle2. Consequently, while Dimple demonstrates competitive performance among models of similar scale, it still falls short compared to the most powerful models.

Acknowledgment

This project is supported by the Ministry of Education, Singapore, under the Academic Research Fund Tier 1 (FY2026, WBS: A-8004345-00-00), the National Research Foundation, Singapore, and Cyber Security Agency of Singapore under its National Cybersecurity R&D Programme and CyberSG R&D Cyber Research Programme Office (Award: CRPO-GC1-NTU-002).

References

- Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2023. [Structured denoising diffusion models in discrete state-spaces](#). *Preprint*, arXiv:2107.03006.
- Mozhgan Nasr Azadani, James Riddell, Sean Sedwards, and Krzysztof Czarnecki. 2025. [Leo: Boosting mixture of vision encoders for multimodal large language models](#). *Preprint*, arXiv:2501.06986.
- Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*.
- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. [Qwen2.5-vl technical report](#). *Preprint*, arXiv:2502.13923.
- Zalán Borsos, Matt Sharifi, Damien Vincent, Eugene Kharitonov, Neil Zeghidour, and Marco Tagliasacchi. 2023. [Soundstorm: Efficient parallel audio generation](#). *Preprint*, arXiv:2305.09636.
- Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. 2022. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, volume 35, pages 28266–28279. Curran Associates, Inc.
- Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. 2024. [Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design](#). *Preprint*, arXiv:2402.04997.
- Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T. Freeman, Michael Rubinstein, Yanzhen Li, and Dilip Krishnan. 2023. [Muse: Text-to-image generation via masked generative transformers](#). *Preprint*, arXiv:2301.00704.
- Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. 2022. [Maskgit: Masked generative image transformer](#). *Preprint*, arXiv:2202.04200.
- Guo Chen, Zhiqi Li, Shihao Wang, Jindong Jiang, Yicheng Liu, Lidong Lu, De-An Huang, Wonmin Byeon, Matthieu Le, Tuomas Rintamaki, Tyler Poon, Max Ehrlich, Tuomas Rintamaki, Tyler Poon, Tong Lu, Limin Wang, Bryan Catanzaro, Jan Kautz, Andrew Tao, and 2 others. 2025. [Eagle 2.5: Boosting long-context post-training for frontier vision-language models](#). *Preprint*, arXiv:2504.15271.
- Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, Ji Ma, Jiaqi Wang, Xiaoyi Dong, Hang Yan, Hewei Guo, Conghui He, Botian Shi, Zhenjiang Jin, Chao Xu, and 16 others. 2024a. [How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites](#). *Preprint*, arXiv:2404.16821.
- Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. 2024b. [Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks](#). *Preprint*, arXiv:2312.14238.
- Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, and 1 others. 2024. [Vlmevalkit: An open-source toolkit for evaluating large multi-modality models](#). In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11198–11201.
- Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. 2024. Discrete flow matching. In *Advances in Neural Information Processing Systems*, volume 37, pages 133345–133385. Curran Associates, Inc.
- Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. 2025. [Scaling diffusion language models via adaptation from autoregressive models](#). *Preprint*, arXiv:2410.17891.
- Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, and 1 others. 2024. [Scaling diffusion language models via adaptation from autoregressive models](#). *arXiv preprint arXiv:2410.17891*.
- Jarvis Guo, Tuney Zheng, Yuelin Bai, Bo Li, Yubo Wang, King Zhu, Yizhi Li, Graham Neubig, Wenhui Chen, and Xiang Yue. 2025. [Mammoth-vl: Eliciting multimodal reasoning with instruction tuning at scale](#). *Preprint*, arXiv:2412.05237.
- Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023. [Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control](#). *Preprint*, arXiv:2210.17432.

- Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. [Argmax flows and multinomial diffusion: Learning categorical distributions](#). *Preprint*, arXiv:2102.05379.
- Kuan-Po Huang, Shu wen Yang, Huy Phan, Bo-Ru Lu, Byeonggeun Kim, Sashank Macha, Qingming Tang, Shalini Ghosh, Hung yi Lee, Chieh-Chi Kao, and Chao Wang. 2025. [Impact: Iterative mask-based parallel decoding for text-to-audio generation with diffusion modeling](#). *Preprint*, arXiv:2506.00736.
- Bo Li, Kaichen Zhang, Hao Zhang, Dong Guo, Renrui Zhang, Feng Li, Yuanhan Zhang, Ziwei Liu, and Chunyuan Li. 2024a. [Llava-next: Stronger llms supercharge multimodal capabilities in the wild](#).
- Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. 2024b. [Lmms-eval: Accelerating the development of large multimodal models](#).
- Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. 2025a. [Lavida: A large diffusion language model for multimodal understanding](#). *Preprint*, arXiv:2505.16839.
- Tianhong Li, Huiwen Chang, Shlok Kumar Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. 2023a. [Mage: Masked generative encoder to unify representation learning and image synthesis](#). *Preprint*, arXiv:2211.09117.
- Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. 2024c. [Autoregressive image generation without vector quantization](#). *Preprint*, arXiv:2406.11838.
- Yifan Li, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. [Diffusion models for non-autoregressive text generation: A survey](#). *Preprint*, arXiv:2303.06574.
- Zhiqi Li, Guo Chen, Shilong Liu, Shihao Wang, Vibashan VS, Yishen Ji, Shiyi Lan, Hao Zhang, Yilin Zhao, Subhashree Radhakrishnan, Nadine Chang, Karan Sapra, Amala Sanjay Deshmukh, Tuomas Rintamäki, Matthieu Le, Ilia Karmanov, Lukas Voegtli, Philipp Fischer, De-An Huang, and 8 others. 2025b. [Eagle 2: Building post-training data strategies from scratch for frontier vision-language models](#). *Preprint*, arXiv:2501.14818.
- Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023a. [Improved baselines with visual instruction tuning](#). *Preprint*, arXiv:2310.03744.
- Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. [Llava-next: Improved reasoning, ocr, and world knowledge](#).
- Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. Visual instruction tuning. In *Conference on Neural Information Processing Systems (NeurIPS)*.
- Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. [Discrete diffusion modeling by estimating the ratios of the data distribution](#). *Preprint*, arXiv:2310.16834.
- Gen Luo, Xue Yang, Wenhan Dou, Zhaokai Wang, Jiawen Liu, Jifeng Dai, Yu Qiao, and Xizhou Zhu. 2025. [Mono-intervl: Pushing the boundaries of monolithic multimodal large language models with endogenous visual pre-training](#). *Preprint*, arXiv:2410.08202.
- Chenlin Meng, Kristy Choi, Jiaming Song, and Stefano Ermon. 2022. Concrete score matching: Generalized score matching for discrete data. In *Advances in Neural Information Processing Systems*, volume 35, pages 34532–34545. Curran Associates, Inc.
- Fnu Mohbat and Mohammed J. Zaki. 2024. [Llava-chef: A multi-modal generative model for food recipes](#). *Preprint*, arXiv:2408.16889.
- 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](#). *Preprint*, arXiv:2502.09992.
- Mihir Prabhudesai, Mengning Wu, Amir Zadeh, Kateřina Fragkiadaki, and Deepak Pathak. 2025. [Diffusion beats autoregressive in data-constrained settings](#). *Preprint*, arXiv:2507.15857.
- Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024a. [Simple and effective masked diffusion language models](#). *Preprint*, arXiv:2406.07524.
- Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024b. [Simple and effective masked diffusion language models](#). *Preprint*, arXiv:2406.07524.
- Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. 2025a. [Simplified and generalized masked diffusion for discrete data](#). *Preprint*, arXiv:2406.04329.
- Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. 2025b. [Simplified and generalized masked diffusion for discrete data](#). *Preprint*, arXiv:2406.04329.
- Robin Strudel, Corentin Tallec, Florent Althé, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. 2022. [Self-conditioned embedding diffusion for text generation](#). *Preprint*, arXiv:2211.04236.
- Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. 2023. Score-based continuous-time discrete diffusion models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

- Alexander Swerdlow, Mihir Prabhudesai, Siddharth Gandhi, Deepak Pathak, and Katerina Fragkiadaki. 2025. [Unified multimodal discrete diffusion](#). *Preprint*, arXiv:2503.20853.
- Shitong Xu. 2022. [Clip-diffusion-lm: Apply diffusion model on image captioning](#). *Preprint*, arXiv:2210.04559.
- Ling Yang, Ye Tian, Bowen Li, Xincheng Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. 2025. [Mmada: Multimodal large diffusion language models](#). *arXiv preprint arXiv:2505.15809*.
- Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025a. [Beyond autoregression: Discrete diffusion for complex reasoning and planning](#). *Preprint*, arXiv:2410.14157.
- Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025b. [Dream 7b](#).
- Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. 2025. [Llada-v: Large language diffusion models with visual instruction tuning](#). *Preprint*, arXiv:2505.16933.
- Runpeng Yu, Qi Li, and Xinchao Wang. 2025. [Discrete diffusion in large language and multimodal models: A survey](#). *Preprint*, arXiv:2506.13759.
- Weihaoyu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. 2023. [Mm-vet: Evaluating large multimodal models for integrated capabilities](#). *Preprint*, arXiv:2308.02490.
- Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. 2023. [Seqdiffuseq: Text diffusion with encoder-decoder transformers](#). *Preprint*, arXiv:2212.10325.
- Haoyu Zhang, Yangyang Guo, and Mohan Kankanhalli. 2024a. [Joint vision-language social bias removal for clip](#). *Preprint*, arXiv:2411.12785.
- Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. 2024b. [Lmms-eval: Reality check on the evaluation of large multimodal models](#). *Preprint*, arXiv:2407.12772.
- Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. 2024. [A reparameterized discrete diffusion model for text generation](#). *Preprint*, arXiv:2302.05737.
- Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. [Llada 1.5: Variance-reduced preference optimization for large language diffusion models](#). *Preprint*, arXiv:2505.19223.
- Alon Ziv, Itai Gat, Gael Le Lan, Tal Remez, Felix Kreuk, Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2024. [Masked audio generation using a single non-autoregressive transformer](#). *Preprint*, arXiv:2401.04577.

- Supplementary Material -

S1 Training and Evaluation

S1.1 Training Configuration and Hyperparameters

Training Phase	1st: Autoregressive Alignment	2nd: Autoregressive Instruction Tuning	3rd: Diffusion Instruction Tuning
Training Method	Autoregressive	Autoregressive	Diffusion
Training Data	LLaVA-CC3M-Pretrain	LLaVA-NEXT Instruction Tuning	
#Training Data	559k	739k	
learning rate	0.001	5e-6	5e-7
batch size	256	128	128
weight decay	0	0	0
warmup ratio	0.03	0.03	0.03
lr schedule	linear	linear	linear
max grad norm	1	1	0.1
epoch	1	1	1
optimizer	AdamW	AdamW	AdamW

Table S1: Training Configuration and Hyper-Parameters.

The training of Dimple can be divided into three stages. The first two stages involve autoregressive alignment and instruction-following tuning, while the final stage utilizes diffusion-based instruction-following tuning. The specific training configurations and hyperparameters for each stage are detailed in Tab. S1. The training hyperparameters of Dimple+ are the same as those used in the DT stage of Dimple. The training of Dimple and Dimple+ was conducted on H100 clusters, requiring approximately 100 GPU hours each. Experiments in Tab. 3 is conducted on one single H100 GPU.

The number of training iterations for the Dimple AR baseline equals the sum of the iterations used in the autoregressive training and diffusion training stages of Dimple. The AR baseline for Dimple+ is trained with the same number of iterations as Dimple+. For both cases, the training hyperparameters of the AR baselines are aligned with those used in the autoregressive training stage of Dimple.

S1.2 Masking Strategy

During training, the Autoregressive Alignment and Autoregressive Tuning phases use the standard next-token prediction loss from AR training. In these stages, input samples are not masked. In contrast, during the Diffusion Tuning stage, the response parts of the input samples are masked. The masking strategy follows a linear schedule, where the proportion of masked tokens is proportional to the diffusion time step. Let $r \in [0, 1]$ and $t \in [0, 1]$ denote the masking ratio and the time index, respectively. We define $r = t$. That is, when $t = 0$, no tokens are masked, and when $t = 1$, all tokens are masked. This linear masking strategy is also commonly used in prior discrete diffusion LLM training (Ye et al., 2025b; Gong et al., 2024; Campbell et al., 2024).

S1.3 Training Sample Padding

Unlike autoregressive models, dMLLMs generate answers with a given response length and do not rely on a [EOS] token to terminate the generation. To accommodate the diffusion generation process, we replace the [EOS] token in the original training dataset with a random number of padding tokens specific to Dream’s tokenizer. These tokens are used by the Dream model to learn to adaptively control the actual length of the generated answers.

The Dream model relies on a predefined response length to determine the length of generated outputs, ensuring that the number of textual tokens does not exceed this length. When the number of textual tokens is less than the response length, Dream generates additional [Padding] tokens so that the total number of tokens equals the predefined response length. Therefore, the model does not rely on a [EOS]

Text Input for Autoregressive Training	
User:	[BOS]<image>\nWhat type of item is holding the napkin?\nAnswer the question with GPT-T-COCO format.[EOS]
Assistant:	[BOS]A clear napkin holder is holding the napkin.[EOS]
User:	[BOS]What color is the napkin in the napkin holder?[EOS]
Assistant:	[BOS]The napkin inside the napkin holder is blue-green.[EOS]
Text Input for Diffusion Training	
User:	[BOS]<image>\nWhat type of item is holding the napkin?\nAnswer the question with GPT-T-COCO format.[EOS]
Assistant:	[BOS]A clear napkin holder is holding the napkin. $\underbrace{\text{[Padding]} \dots \text{[Padding]}}_{n_1 \text{ [Padding] Tokens}}$
User:	[BOS]What color is the napkin in the napkin holder?[EOS]
Assistant:	[BOS]The napkin inside the napkin holder is blue-green. $\underbrace{\text{[Padding]} \dots \text{[Padding]}}_{n_2 \text{ [Padding] Tokens}}$

Table S2: Training Examples Used for Autoregressive Training and Diffusion Training. n_1 and n_2 are random integers that enhance the model’s robustness to varying response lengths.

token to terminate generation. Instead, [EOS] tokens in the original training data should be replaced with [Padding] tokens.

To enhance the Dimple model’s robustness to response length, we randomly replace each single [EOS] token with n [Padding] tokens. The max and min value of n is selected based on the length of the ground truth response. For a ground truth answer with the length of l , the derivation of n can be written as follows.

$$n \sim \text{RandomInteger}(n_{min}, n_{max}),$$

$$(n_{min}, n_{max}) = \begin{cases} (l + 1, 2^{\lceil \log_2(l+1) \rceil}), & \text{if } n < 16, \\ (l + 1, 16 \cdot \lceil \frac{l+1}{16} \rceil), & \text{otherwise} \end{cases}$$

Tab. S2 show an example of input sample for diffusion training.

S1.4 Evaluation Configuration

We conducted benchmark evaluations using the Imms-eval library. Both the evaluation metrics and prompts were based on the default configuration provided by Imms-eval. Throughout all evaluations, we employed the MaskGIT decoding algorithm from Dream, with the algorithm temperature set to 0 and confident decoding disabled. The overall decoding temperature was set to 0, and the top-p value was fixed at 0.01, resulting in a largely deterministic generation process.

The most influential parameter affecting evaluation performance is the response length. For all datasets except MMVet, we experimented with response lengths of 4, 8, and 16. Table 1 reports the best results among these three settings. The MMVet dataset is an exception due to the high variability in the ground truth response lengths—ranging from a single word to over 500 tokens. For MMVet, we fixed the token length for all responses to 64. While this is not an optimal choice, as it limits the model’s capacity to fully address long-response questions and wastes tokens on short responses, it provides a uniform evaluation setting. We consider adaptive response length selection as a direction for future research. During evaluation, the number of generation steps was set equal to the response length, enforcing the model to decode exactly one token per step.

Following the default configuration, the evaluation metrics used for each benchmark are listed as follows.

- **Accuracy:** GQA, MMBench, POPE, MMMU, SQA, AI2D, ChartQA, TextQA
- **Number of Correct Answer:** MME, OCRBench
- **GPT-Based Accuracy:** MathVista, MMVet

We collect the number in the Table 1 as follows. Dimple’s, Dimple+’s and their AR Baselines’ performance is obtained using Imms-eval library, which is also used in LLaVA (Li et al., 2024b; Zhang et al., 2024b). For Qwen2.5-VL, results are collected from their official reports and VLMEvalKit (Duan et al., 2024). For LLaVA-NEXT, results on OCRBench are taken from (Azadani et al., 2025); results on MathVista test-mini, ChartQA, and AI2D are obtained using Imms-eval. Other results for LLaVA-NEXT and the results for other models are collected from their official reports. The training data information is collected from the respective official reports.

S1.5 Evaluation of Model after Alignment

To evaluate the captioning capabilities of the model after alignment, we employed GPT-4o for assessment. The prompt used for evaluation is shown in Tab. S3.

S1.6 Model and Data Used

The Qwen2.5 and Qwen2.5-VL series is released with explicit documentation regarding model scale, training data composition, language coverage, and licensing conditions. The model family provides broad multilingual support, including Chinese and English. Most open-weight Qwen2.5 and Qwen2.5-VL models are released under the Apache 2.0 License.

The LLaVA-CC3M-Pretrain dataset is a visual instruction pretraining dataset derived from CC3M and related captioning resources. Its dataset card identifies the modality as image-text data and the language as English. The dataset documentation states that the data is intended to support research on large multimodal models and to facilitate reproduction of the LLaVA work. It also specifies that users must comply with the original licenses and terms of the underlying CC3M and BLIP resources. CC3M permits use of the dataset while requesting acknowledgement of Google LLC as the data source and providing the data on an “as is” basis. In this paper, our use of LLaVA-CC3M-Pretrain Data is solely for academic research and is consistent with the stated purpose and licensing requirements of the dataset.

The LLaVA-NeXT instruction tuning data is associated with the LLaVA-NeXT project, which emphasizes that its datasets and checkpoints are subject to the licenses and terms of their original sources. These include, where applicable, the OpenAI Terms of Use, LLaMA community or research licenses, Tongyi Qianwen research license terms, and the licenses of underlying vision-language datasets. The LLaVA-NeXT documentation further describes related evaluation and recaptioned data, including English and Chinese image captioning examples and video captioning data derived from ActivityNet, while noting that some evaluation data is intended for internal development rather than formal benchmarking. In this work, our use of LLaVA-NeXT instruction tuning resources follows the relevant original license conditions and remains within the academic research context for which these resources were released.

S2 Confident Decoding

We propose **Confident Decoding**, a token selection strategy designed for diffusion-based generation. Unlike traditional decoding methods that fix the number of tokens to be updated per iteration, Confident Decoding dynamically selects positions to decode based on the model’s confidence in its predictions.

Overview. At each decoding step, the model evaluates the confidence of its predictions across all masked positions. If any of the confidence scores exceed a predefined threshold γ , the corresponding tokens are decoded all together. Otherwise, a fallback strategy is used to ensure generation continues in accordance with the total decoding budget. That is, if no confidence score exceeds the threshold, the second stage of selection is triggered, and another decoding algorithm can be invoked. Thus, the complete decoding pipeline can combine Confident Decoding with MaskGIT, Random selection, or other strategies. Alg. 1 presents an example of the pseudocode for Confident Decoding. This mechanism enables adaptive decoding, leading to both improved generation quality and computational efficiency.

Confidence Estimation. Confident Decoding does not impose any constraints on the method for computing confidence. It can use the maximum prediction probability as the confidence value, as used in MaskGIT, the entropy of the prediction distribution, or the margin between the highest and second-highest

Algorithm 1: Confident Decoding

```
// Take temperature as an example of probability adjustment.
// Confidence is calculated in a MaskGIT way.
// Use random selection callback.
Input: Input token sequence  $\mathbf{x}_t$ , logits  $\mathbf{z}_t$ , temperature  $\tau$ , threshold  $\gamma$ , number of positions  $K$  to
        select if the fallback is triggered, number of masked tokens  $N$ 
Output: Updated token sequence  $\mathbf{x}_{t+1}$ 

// Step 1: Sample token predictions and compute confidence
// Pre-revision probabilities
 $\mathbf{p}_t \leftarrow \text{softmax}(\mathbf{z}_t)$ ;
// Post-revision probabilities
 $\tilde{\mathbf{p}}_t \leftarrow \text{softmax}(\mathbf{z}_t/\tau)$ ;
foreach  $i \in \{1, \dots, N\}$  do
    // MaskGIT confidence
     $c^{(i)} = \max(\mathbf{p}_t^{(i)})$ ;
    // sampling
     $\tilde{x}_t^{(i)} \sim \text{Categorical}(\tilde{\mathbf{p}}_t^{(i)})$ ;
// Step 2: Position selection
 $x_{t+1} \leftarrow x_t$ ;
if  $\exists i, c^{(i)} \geq \gamma$  then
    // Step 2.1: Confident decoding
     $\mathcal{I} \leftarrow \{i \mid c^{(i)} \geq \gamma, i \in \{1, \dots, N\}\}$ ;
    foreach  $i \in \mathcal{I}$  do
         $x_{t+1}^{(i)} \leftarrow \tilde{x}_t^{(i)}$ ;
else
    // Step 2.2: Fallback: decoding via random selection
    Randomly select  $K$  positions:  $\mathcal{I} \leftarrow \text{RandomSample}(\{1, \dots, N\}, K)$ ;
    foreach  $i \in \mathcal{I}$  do
         $x_{t+1}^{(i)} \leftarrow \tilde{x}_t^{(i)}$ ;
return  $x_{t+1}$ 
```

prediction probabilities. In general, the confidence can be expressed as a function $c = \phi(p)$, where p is a prediction probability vector of length V , and the output is a scalar confidence score.

Since the prediction probabilities in LLMs can be revised by decoding parameters such as temperature, top-p and top-k, prior work has considered two options for choosing the input probability p : either before (Nie et al., 2025) or after such revisions (Ye et al., 2025b). In our experiments, we use the prediction probabilities before these revisions to compute confidence. This choice is motivated by the fact that adjustments, like temperature and top-p, are applied in a position-wise manner, without accounting for inter-positional relationships or preserving the relative confidence across different positions. For example, in high-temperature or low top-p settings, the revised probability may become almost deterministic, where one position has a probability of approximately 1 and all others are 0. Such a probability is uninformative for computing confidence, as all positions would yield identical confidence scores.

S3 Ablation on Dimple Training

Following experiments demonstrates the advantages of the proposed autoregressive-then-diffusion hybrid training paradigm over the pure diffusion training. The pure diffusion pipeline consists of 2 phases: Diffusion Alignment (DA) and Diffusion Tuning (DT). Our hybrid approach incorporates 3 phases:

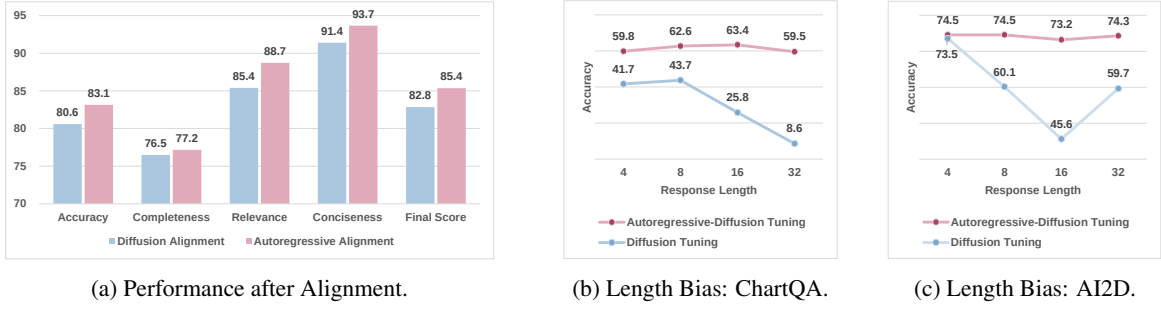


Figure S1: Comparison of Captioning Capabilities After Alignment (Fig. S1a), and the Length Bias Phenomenon on the ChartQA and AI2D Datasets After Instruction Tuning (Figs. S1b and S1c).

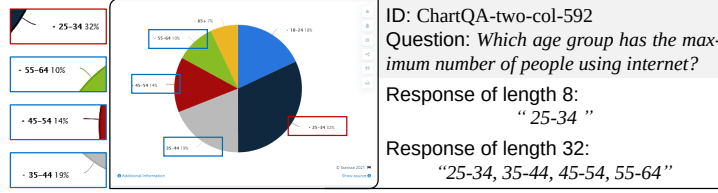


Figure S2: An example of Length Bias Phenomenon Collected from ChartQA. The response is generated by the model after 2-epoch diffusion tuning. The red and blue boxes on the left indicate the source locations of the numbers in the response.

Autoregressive Alignment (AA), Autoregressive Tuning (AT), and Diffusion Tuning (DT).

After Alignment To evaluate the captioning capability of the aligned models, we randomly sampled 128 images from the LLaVA-CC3M dataset (Liu et al., 2023b) (excluded from training) and generated captions using the aligned models. The model trained with DA used the diffusion generation pipeline, while the model trained with AA used the autoregressive generation pipeline. Given an image and its caption, we prompted GPT-4o-2024-11-20 to rate the captions on four dimensions—accuracy, completeness, relevance, and conciseness—on a scale of 1–100, with the final score being the average of the four dimensions. The results, summarized in Fig. S1a, show that the model aligned using AA outperforms the one aligned using DA across all metrics.

After Instruction Tuning We compared six different instruction tuning strategies:

- Strategies A, B: Continue with DT for 1 or 2 epochs after DA.
- Strategies C, D: Perform 1 epoch of AT with two different learning rates after AA.
- Strategies E, F: Continue with 1 epoch of DT using two different learning rates after C and D.

Tab. S4 presents the performance of all six strategies. Since the final goal is to obtain a model capable of diffusion generation, all models were evaluated using the diffusion generation pipeline. First, comparing strategies A and D, which are both trained with the same learning rate for 1 epoch, AT outperforms DT on 7 out of 9 benchmarks. However, models trained with AT alone suffer from a gap between autoregressive training and diffusion-based inference. Then, comparing strategies B and F, the AT-then-DT strategy outperform the pure diffusion tuning strategy across all benchmarks.

Length Bias Another notable weakness of models trained purely with diffusion is the presence of severe length bias, making them highly sensitive to the response length hyperparameter. This issue is especially pronounced in chart-based QA tasks. Figs. S1b and S1c illustrate the relationship between response length and accuracy on AI2D and ChartQA. The AT+DT model maintains relatively stable performance across varying lengths, whereas the pure diffusion model shows dramatic performance drops as length increases. For example, in ChartQA, accuracy drops from 42.7% to 8.6% as the response length increases from 8 to 32. Fig. S2 shows an example from ChartQA, where the pure diffusion model exhibits a tendency to translate all mask tokens into normal textual tokens. This behavior leads the model to continuing to generate text even after producing a correct answer, as it lacks the ability to “stop” appropriately.

S4 Sensitivity Analysis on Generation Parameters

We conducted a sensitivity analysis on the following hyperparameters: Response Length (L), Number of Steps (k), Confidence Threshold (γ), Temperature (t), and Top-p (p). **1. Response Length (L)**

Fixing $k = L$, $\gamma = 1$, $t = 0$, and $p = 0.01$, we analyzed the sensitivity to response length. While response length does affect performance, its impact is generally minor across most datasets.

2. Steps (k)

Fixing $L = 8$, $\gamma = 1$, $t = 0$, and $p = 0.01$, we varied k and observed that reducing k degrades performance, indicating a trade-off between decoding speed and performance.

3. Confident Decoding Threshold (γ)

Fixing $L = 8$, $t = 0$, and $p = 0.01$, we evaluated the effect of confident decoding. Within the range $\gamma \in [0.8, 1.0]$, confident decoding remains stable and does not significantly impact performance across most datasets.

4. Temperature (t)

Fixing $L = k = 8$, $\gamma = 1$, and $p = 0.95$, we varied the temperature. The model performs best when $t < 0.2$; higher values of t significantly harm performance.

5. Top P (p)

Fixing $L = k = 8$, $\gamma = 1$, and $t = 0.2$, we varied the top-p value and found that performance is largely unaffected. This suggests the model is not highly sensitive to p .

S5 Further Discussion on Related Works

S5.1 Multimodal Large Language Models

Multimodal large language models (MLLMs) have rapidly progressed with several open-source series such as LLaVA, Eagle, InternVL, and Qwen-VL. LLaVA aligns vision encoders with LLMs via visual instruction tuning and demonstrates competitive performance using efficient training and external tool use (Liu et al., 2023b,a; Li et al., 2024a; Liu et al., 2024). Eagle models explore vision-language integration through multiple encoders and long-context strategies, notably Automatic Degrade Sampling and Image Area Preservation (Azadani et al., 2025; Li et al., 2025b; Chen et al., 2025). InternVL focuses on scaling vision-language models, evolving from large vision encoders to joint multimodal pretraining, and achieves state-of-the-art results with models like InternVL2.5 and InternVL3 (Chen et al., 2024b,a; Mohbat and Zaki, 2024; Luo et al., 2025). Qwen-VL introduces multilingual vision-language models with dynamic resolution processing and bounding-box generation, achieving competitive performance across diverse tasks (Bai et al., 2023; Zhang et al., 2024a; Bai et al., 2025).

With the advancement of discrete large language models (dLLMs), discrete diffusion has also been applied to multimodal understanding tasks. Concurrent with Dimple, models such as UniDisc (Swerdlow et al., 2025), MMaDA (Yang et al., 2025), LLaDA-V (You et al., 2025), and LaViDA (Li et al., 2025a) have emerged. Among them, UniDiscc (Swerdlow et al., 2025) and MMaDA (Yang et al., 2025) are unified discrete diffusion models that jointly model multimodal and text generation within the discrete diffusion framework. Dimple, LLaDA-V (You et al., 2025), and LaViDA (Li et al., 2025a) are all multimodal understanding models. In terms of training, LLaDA-V (You et al., 2025) successfully scales the training dataset to 17 million samples. LaViDA (Li et al., 2025a) introduces complementary masking strategies for training data, while Dimple proposes a hybrid training method called “First-Autoregressive-then-Diffusion”. For inference, both Dimple and LaViDA (Li et al., 2025a) demonstrate the effectiveness of pre-filling techniques in discrete diffusion models. Additionally, Dimple introduces the Confident Decoding algorithm, the first method capable of dynamically controlling the number of generation steps in a discrete diffusion model, thereby balancing generation speed and quality.

S5.2 Parallel-Decoding

Parallel decoding techniques have been widely used in image generation (Chang et al., 2022, 2023; Li et al., 2023a, 2024c) and audio generation (Huang et al., 2025; Borsos et al., 2023; Ziv et al., 2024) to accelerate the generation process. There are two main approaches for selecting which token positions to decode. One approach relies on the magnitude of confidence score: these methods first predict tokens at

all positions, compute confidence score for each position, and then retain the predictions at positions with high confidence scores (Chang et al., 2022, 2023; Li et al., 2023a; Borsos et al., 2023; Ziv et al., 2024). Another approach randomly selects a subset of positions to decode (Li et al., 2024c; Huang et al., 2025).

In these works, the number of positions decoded at each timestep is determined by a pre-defined scheduling function. For example, the commonly used cosine schedule imposes a fixed slow-to-fast generation pattern. A limitation of this strategy is that the top-k tokens with the highest confidence scores are always decoded, even if some of them have low absolute confidence. For example, if the top two confidence scores are 0.95 and 0.1, both tokens will be decoded, despite the second one being highly uncertain.

To address this limitation, our confident decoding algorithm differs from previous works by allowing the model to dynamically determine the number of tokens to decode at each timestep based on its own confidence score. Specifically, we set a confidence threshold and only decode tokens whose predicted confidence exceeds this threshold. This prevents the model from decoding uncertain tokens and allows it to adaptively adjust decoding speed depending on the input query. While both our approach and prior works make use of confidence scores, our method introduces greater flexibility by not fixing the number of tokens decoded per step.

S5.3 Diffusion Multimodal Large language Models

In the main-text, we provide the discussion and performance comparison among dMLLMs. Here, we provide Tab. S10 for a summarization of the differences among these models. Because UniDisc uses Held-out evaluation, it trains one model for each dataset. So we leave its #Training Data as blank.

S6 Discussions

Limitations. Dimple-7B is trained on less than 1/20th of the data used by state-of-the-art models such as Qwen2.5-VL and Eagle2. Consequently, while Dimple demonstrates competitive performance among models of similar scale, it still falls short compared to the most powerful models. Therefore, scaling up training data and model parameters is a key future direction for dMLLM. Although confident decoding improves efficiency, the full attention mechanism in discrete diffusion still incurs quadratic memory and computational complexity, limiting scalability to longer sequences and low-resource environments. Thus, further investigation into optimizing dMLLM inference is essential.

Broader Impact Dimple’s safety concerns arise from both the underlying DLM model and Dimple’s vision encoding process. Like other MLLMs, it is also susceptible to issues such as hallucination and generation bias.

Safeguards. The following Usage Disclosure is included in the Dimple document. “We release the Dimple model for academic and non-commercial research purposes only. Users are strongly discouraged from using the model to generate misleading or harmful content, including but not limited to disinformation, impersonation, or harassment. Dimple is not designed or validated for use in safety-critical domains such as medical decision-making, autonomous vehicles, or legal advice systems. ”

S7 Examples

In this section, we provide additional examples of Dimple’s generations to qualitatively showcase its capabilities. A typical phenomenon is that the existence of randomness in the decoding algorithm affects the length of the meaningful text generated. Here, meaningful text refers to tokens excluding special tokens such as padding token and mask token. Given a response length, shorter outputs indicate a higher proportion of meaningless padding tokens, whereas longer outputs imply fewer such tokens. Here, the decoding algorithm specifically refers to the method used to select which tokens are decoded, such as MaskGIT. The randomness in the decoding algorithm is affected by two factors: first, whether token selection is performed via sampling or by greedily choosing the position with the highest confidence; second, the value of the temperature parameter. When there is no randomness, Dimple tends to generate shorter responses. In contrast, with randomness, Dimple is more likely to produce longer responses.

Not all questions exhibited the aforementioned phenomenon. Some questions are inherently unaffected. For instance, as shown in Tab. S11, mathematical reasoning tasks naturally require multi-step reasoning, resulting in relatively long answers regardless of the presence of randomness. In contrast, in Tabs. S12 and S13, tasks such as pure OCR recognition or counting—which involve only perception without the need for deeper understanding or analysis—tend to yield short answers, regardless of the decoding strategy.

However, for tasks where the response length is not intrinsically fixed, the presence or absence of randomness in decoding has a significant impact. For example, in Tabs. S14 to S16, tasks that require both perception and visual understanding, or involve describing images, clearly show that decoding with randomness tends to generate much longer responses.

In Tabs. S14 to S16, the response length is set to be 128 and the decoding algorithm is MaskGIT. For the “without randomness” setup, the temperature of the MaskGIT is 0. For the “with randomness” setup, the temperature is 0.4.

Caption Evaluation Instructions

You will be given one image and one generated visual caption describing the image. Your task is to carefully evaluate how well the caption reflects the content of the image based on four dimensions:

1. Accuracy: Does the caption correctly describe the objects, actions, or scene in the image?
2. Completeness: Does the caption capture the essential and important aspects of the image, without omitting critical content?
3. Relevance: Is all the information in the caption relevant to the image? Are there any hallucinated, fabricated, or irrelevant elements?
4. Conciseness: Is the caption free from redundancy or unnecessary elaboration?

For each dimension, assign a score from 0 to 100 and present it in the following format:

““

****Accuracy****

Justification: {{your explanation here}}

Accuracy score: {{score}}

****Completeness****

Justification: {{your explanation here}}

Completeness score: {{score}}

****Relevance****

Justification: {{your explanation here}}

Relevance score: {{score}}

****Conciseness****

Justification: {{your explanation here}}

Conciseness score: {{score}}

””

Then, provide the average score over the four dimensions. Justify your overall score in a paragraph and finish with the sentence:

““The final score is {{score}}.””

Image: [image]

Generated caption: [caption]

Table S3: Instructions used when Evaluating Captions for the Models after Alignment.

		TextVQA	SQA	MMMU	MME	POPE	MMVet	ChartQA	AI2D	
A	DT (lr=5e-6, 1 Epoch)	38.4	70.2	45.1	1420	360	83.9	35.2	40.8	71.3
B	DT (lr=5e-6, 2 Epoch)	36.6	70.6	44.0	1508	397	84.7	34.9	43.7	73.5
C	AT (lr=2e-5, 1 Epoch)	49.2	79.5	38.1	1298	403	84.3	32.9	53.7	70.5
E	+ DT (lr=5e-6, 1 Epoch)	49.8	81.9	42.6	1483	426	85.3	34.1	53.9	74.4
D	AT (lr=5e-6, 1 Epoch)	49.8	83.7	43.3	1404	422	82.6	36.3	55.6	71.8
F	+ DT (lr=5e-7, 1 Epoch)	49.4	84.4	45.2	1514	432	85.1	41.2	58.4	74.4

Table S4: Performance Comparison Between Autoregressive Tuning (AT) and Diffusion Tuning (DT). The results show that the best performance is achieved by first applying autoregressive tuning followed by diffusion tuning.

L	CharQA	SQA	POPE	AI2D
16	63.3	76.7	86.3	73
8	63.0	77.2	86.1	72.9
4	56.4	77.5	86.2	72.9

Table S5: Sensitivity analysis of L .

k	CharQA	SQA	POPE	AI2D
8	63.0	77.2	86.1	72.9
4	57.2	76.1	85.9	72.8
2	52.2	76.2	85.5	71.1

Table S6: Sensitivity analysis of k .

γ	CharQA	SQA	POPE	AI2D
1	63.0	77.2	86.1	72.9
0.9	62.5	77.0	86.4	73.1
0.8	61.9	77.1	86.3	73.1
0.7	60.1	76.8	85.9	72.8

Table S7: Sensitivity analysis of γ .

t	CharQA	SQA	POPE	AI2D
2	48.5	57.4	77.9	55.6
1	57.9	74.4	83.1	70.3
0.2	63.0	76.9	86.3	73.4
0	63.0	77.2	86.1	72.9

Table S8: Sensitivity analysis of t .

p	CharQA	SQA	POPE	AI2D
0.95	63.6	76.9	86.2	73.4
0.5	63.0	77.2	86.3	72.7
0.1	62.9	77.2	86.1	73.0

Table S9: Sensitivity analysis of p .

	Dimple	UniDisc	MMaDA	LLaDA-V	LaViDA-L	LaViDA-D
Model Size	7B	1.4B	8B	8B	8B	7B
#Training Data	1.3M	*	$\sim 2M$	17M	1.6M	1.6M
Training Loss	AR then Diffusion	Diffusion	Diffusion	Diffusion	Diffusion	Diffusion
Reinforcement Learning			✓			
Complementary Masking Training					✓	✓
Inference with Prefilling	✓				✓	✓
Confident Decoding with Dynamic Inference Scheduling	✓					

Table S10: Technical comparison between Dimple and other approaches, highlighting its AR-then-diffusion training paradigm and the use of Prefilling and Confident Decoding for faster inference.

Mathematical Reasoning

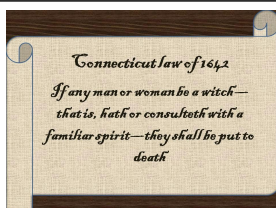
$$(x+3)^2=4$$

User: First please perform reasoning, and think step by step to provide best answer to the following question: What is x in the equation?

Dimple: The equation in the image is $(X+3)^2 = 4$. To find the value of x, we need to solve the equation. First, we take the square root of both sides to get rid of the square. This gives us $x + 3 = \pm 2$. Then, we subtract 3 from both sides to isolate x. This gives us $x = -3 \pm 2$. This results in two possible values for x: $x = -3 + 2 = -1$ and $x = -3 - 2 = -5$. So, the value of x is either -1 or -5.

Table S11: Example of Dimple Generation. The image and question are collected from the MMVet dataset (Yu et al., 2023). This example demonstrates Dimple’s capabilities in OCR and mathematical reasoning.

OCR



User: What are all the scene text in the image?

Dimple: Connecticut law of 1642, If any man or woman be a witch- that is, hath or consulteth with a familiar spirit—they shall be put to death

Table S12: Example of Dimple Generation. The image and question are collected from the MMVet dataset (Yu et al., 2023). This example demonstrates Dimple’s capabilities in pure OCR.

Counting



User: How many tomatoes are there?

Dimple: Five

Table S13: Example of Dimple Generation. The image and question are collected from the MMVet dataset (Yu et al., 2023). This example demonstrates Dimple’s capabilities in counting.

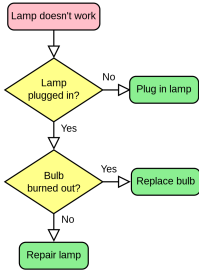
Chart Understanding	
 <pre> graph TD Start([Lamp doesn't work]) --> D1{Lamp plugged in?} D1 -- No --> A1([Plug in lamp]) D1 -- Yes --> D2{Bulb burned out?} D2 -- Yes --> A2([Replace bulb]) D2 -- No --> A3([Repair lamp]) </pre>	
User:	What is the solution if the lamp is plugged in and the bulb has burned out?
Dimple (w/o Randomness):	The solution is to replace the bulb.
Dimple (w Randomness):	If the lamp is not plugged in and the bulb has burned out, the solution would be to replace the bulb. This is because if the bulb is dead and it's not plugged into the power source, then replacing the bulb is necessary to restore the functionality of the lamp.

Table S14: Example of Dimple Generation. The image and question are collected from the MMVet dataset (Yu et al., 2023). This example demonstrates Dimple’s capabilities in OCR and chart understanding.

Recognition	
	
User:	First please perform reasoning, and think step by step to provide best answer to the following question: Is this airplane taking off or landing?
Dimple (w/o Randomness):	The airplane is taking off.
Dimple (w Randomness):	The airplane is taking off because, as can be seen in the image, the landing gear is off the ground, and the lower part of the wing is visible. This indicates the airplane is not yet on the runway, and it is in the air.

Table S15: Example of Dimple Generation. The image and question are collected from the MMVet dataset (Yu et al., 2023). This example demonstrates Dimple’s capabilities in recognition.

Chicken Nugget Map Meme

Sometimes I just look at pictures of the earth from space and I marvel at how beautiful it all is.



User:	Can you explain this meme in detail?
Dimple (w/o Randomness):	The meme features a tray of golden chicken nuggets, arranged to resemble a world map.
Dimple (w Randomness):	The meme is a humorous image featuring a tray of golden breaded or fried food on a baking sheet, along with the text "Sometimes I just look at the pictures of the Earth from space and I marvel at how beautiful it is." The meme appears to be comparing the appearance of the food on the tray to a stunning view of a planet from space, emphasizing the beauty of the Earth in relation to the food. The contrast between the food on the tray and the celestial image adds to the comparison in a playful and entertaining way. The overall message of the meme is to highlight the wonder and beauty of the natural world.

Table S16: Example of Dimple Generation. The image and question are collected from the LLaVA Paper (Liu et al., 2023b). This is a typical example demonstrates the model’s ability to understand memes and the abstract humor embedded within them.

GSM8K Question: Direct Instruction

User:	The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have? Directly answer the question.
Dimple :	9

Table S17: Example of Dimple Generation with explicit instruction.

GSM8K Question: In-Context Learning	
User:	Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now? A: <think> Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. </think> The answer is 11. Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?
Dimple :	<think> They started with 23 apples and used 20 so $23 - 20 = 3$ apples. They bought 6 more so they have $3 + 6 = 9$ apples. </think> The answer is 9.

Table S18: Example of Dimple Generation with In-Context Learning.