Grouped Speculative Decoding for Autoregressive Image Generation

Junhyuk So[1] Juncheol Shin[2] Hyunho Kook[1] Eunhyeok Park[1,2]
[1]Department of Computer Science and Engineering, POSTECH
[2]Graduate School of Artificial Intelligence, POSTECH
{junhyukso,jchshin, kookhh0827, eh.park}@postech.ac.kr

Figure 1. Our ***Grouped Speculative Decoding*** (GSD) accelerates autoregressive image generation up to 4.3x without compromising quality.

Abstract

Recently, autoregressive (AR) image models have demonstrated remarkable generative capabilities, positioning themselves as a compelling alternative to diffusion models. However, their sequential nature leads to long inference times, limiting their practical scalability. In this work, we introduce Grouped Speculative Decoding (GSD), a novel, training-free acceleration method for AR image models. While recent studies have explored Speculative Decoding (SD) as a means to speed up AR image generation, existing approaches either provide only modest acceleration or require additional training. Our in-depth analysis reveals a fundamental difference between language and image tokens: image tokens exhibit inherent redundancy and diversity, meaning multiple tokens can convey valid semantics. However, traditional SD methods are designed to accept only a single most-likely token, which fails to leverage this difference, leading to excessive false-negative rejections. To address this, we propose a new SD strategy that evaluates clusters of visually valid tokens rather than relying on a single target token. Additionally, we observe that static clustering based on embedding distance is ineffective, which motivates our dynamic GSD approach. Extensive experiments show that GSD accelerates AR image models by an average of 3.7× while preserving image quality—all without requiring any additional training. The source code is avaliable at https://github.com/junhyukso/GSD.

1. Introduction

Recent advancements in Large Language Models (LLMs) [1, 3, 23] have demonstrated the remarkable ability of autoregressive (AR) models to capture highly complex distributions. This success has sparked widespread efforts to extend AR models to various domains [2, 13, 27]. In particular, AR image models [12, 19, 21] have achieved impressive generative performance, establishing themselves as a strong alternative to diffusion models [8, 15, 17]. Compared to diffusion-based approaches, AR image models offer several key advantages, including flexible resolution and seamless support for multimodal tasks.

This paper was accepted to the International Conference on Computer Vision (ICCV) 2025.

Despite these advantages, AR image generation faces a major challenge: *sequential token-by-token generation*. Generating a high-resolution image can require an AR model to produce **thousands** of tokens in sequence, whereas diffusion models typically need only **dozens** [17], despite their own limitations due to iterative refinement. This strict sequential dependency imposes a heavy computational burden, leading to significant latency that hinders the practical use of AR models in latency-critical scenarios. Furthermore, unlike text generation—where users can start reading as tokens appear—image generation demands that all pixels be rendered before any meaningful output is visible. This **all-or-nothing nature** of AR image generation results in a frustrating user experience, making it less suitable for interactive applications.

To tackle this challenge, various acceleration techniques have been explored [7, 28, 29]. One promising approach is Speculative Decoding (SD) based techniques [9, 22], originally developed and validated in the context of LLMs [10]. In its representative implementation, SD [10] leverages a lightweight draft model to propose token predictions rapidly, which are then verified in parallel by a larger base model. This allows for the simultaneous generation of multiple tokens, significantly boosting the AR generation speed of language tokens. However, when applied to AR image generation, existing SD-based methods have achieved only modest speed-ups [22] or required additional training for the draft model [9], limiting their practicality.

In this work, we conduct a thorough investigation into **why SD underperforms in AR image generation**. We identify that **the inherent redundancy and diversity among visual tokens** results in relatively **low and uniformly distributed next-token probabilities** during decoding, which significantly reduces SD's acceptance rate. Building on this insight, we propose **Grouped Speculative Decoding (GSD)**—a novel technique that performs SD at the level of semantically valid token groups rather than focusing on a single most-likely token. We theoretically and empirically demonstrate that GSD boosts the acceptance rate with a simple yet effective modification to the acceptance criterion. Additionally, we show that static clustering methods are suboptimal, leading us to introduce **Dynamic GSD**, which dynamically adjusts token grouping based on contextual information. Extensive experiments show that **GSD achieves an average 3.8× speedup** while maintaining high image quality, making it a practical and effective solution for accelerating AR image generation.

2. Preliminaries

2.1. Autoregressive Image model

The modern AR image models [12, 19, 21] typically consist of two components: Vector Quantizer (VQ)[25, 31] and an Autoregressive Transformer [5]. The VQ discretizes continuous images patches into discrete tokens through three main elements: an Encoder $\mathcal{E}$, a Decoder $\mathcal{D}$, and a Codebook $C = \{c_1, c_2, \ldots, c_n\}$, where each code $c_i \in \mathbb{R}^d$.

Formally, the **encoder** maps an input image $X \in \mathbb{R}^{H\times W\times C}$ to a latent representation $x \in \mathbb{R}^{h\times w\times d}$, where h and w denote the spatial dimensions of the latent space, and d is the feature dimension. Each of the d dimensional feature vectors in this latent representation is then quantized by mapping it to the nearest code c in the **codebook**, resulting in the quantized representation x_q. Finally, the **decoder** reconstructs the original RGB image from these quantized codes. The **encoder, codebook, and decoder** are jointly optimized by minimizing the reconstruction loss. This entire process can be summarized as follows:

$$\underbrace{X}_{\mathbb{R}^{H\times W\times C}} \xrightarrow{\mathcal{E}(X)} \underbrace{x \xrightarrow{\text{nearest}} x_q}_{\mathbb{R}^{h\times w\times d}} \xrightarrow{\mathcal{D}(x_q)} \underbrace{\hat{X}}_{\mathbb{R}^{H\times W\times C}}$$

Using this codebook C, the AR Image Model is trained to predict the probability distribution of the next token, just like text-based transformer models [5]. Recent studies [12, 21] have explored a unified generation of both image and text tokens within a single vocabulary, allowing for a more efficient serving system under a unified architecture.

Algorithm 1 Speculative Decoding [10]

Require: Draft Length L, Maximum Length N, Draft model q_θ, Target model p_θ, Initial context $X_{0:n_0}$

$n \leftarrow n_0$
while $n < N$ **do**
  for $j = 0$ to L **do** ▷ AR Draft
    $q_j \leftarrow q(\cdot \mid [X_{0:n}, \hat{X}_{0:j}])$, $\hat{X}_j \sim q_j(\cdot)$
  end for
  parallel for $j = 0$ to L ▷ Parallel Verify
    $p_j \leftarrow p(\cdot \mid [X_{0:n}, \hat{X}_{0:j}])$
  end for
  $(\hat{X}_{0:k},\ k) \leftarrow \texttt{VERIFY}(\hat{X}_{0:L}, p_{0:L}, q_{0:L})$
  $X_{n:n+k-1} \leftarrow \hat{X}_{0:k}$, $n \leftarrow n + k$ ▷ Accept
end while
return X

Algorithm 2 $\texttt{VERIFY}(X, p, q)$

Require: Draft $\hat{X}_{0:L}$, Verifier : $p_{0:L}(\cdot)$, Drafter : $q_{0:L}(\cdot)$

for $k = 0$ to L **do**
  if not $r \sim \mathcal{U}[0,1] \leq \min\left(1, \frac{p_k(\hat{X}_k)}{q_k(\hat{X}_k)}\right)$ **then**
    $x \sim [p_k - q_k]_+$, $\hat{X}_k \leftarrow x$, **break**.
  end if
end for
return $\hat{X}_{0:k}, k$

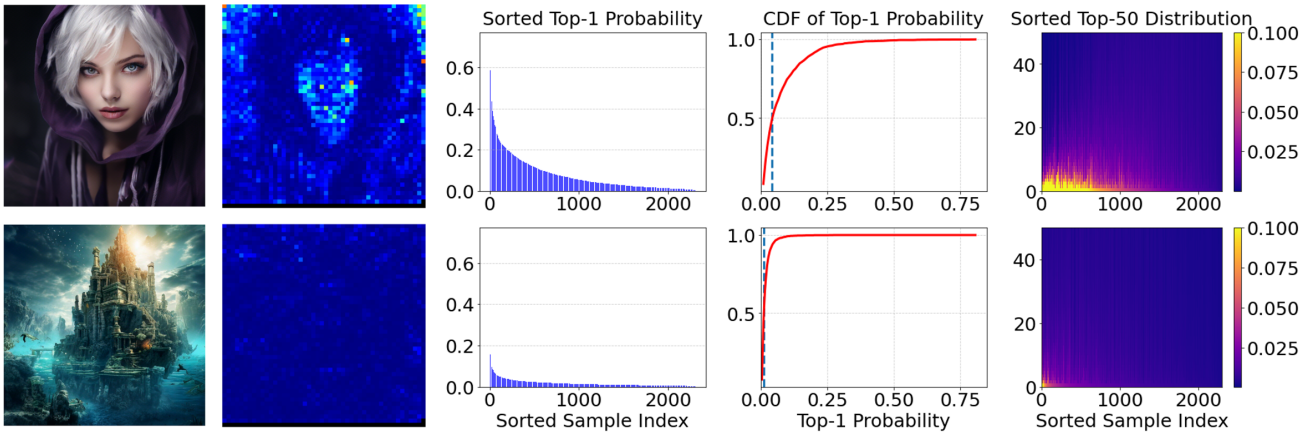


Figure 2. Each patch's p(x) visualization. We used Lumina mGPT with a standard decoding setting ($\tau = 1, K = 2000$)

2.2. Speculative Decoding

To accelerate AR decoding, SD [10] generates multiple tokens simultaneously by leveraging two distinct distributions: a precise expert (base) model $p_\theta(\cdot)$ and a more efficient draft model $q_\theta(\cdot)$. The process begins with the draft model $q(\cdot)$ autoregressively generating L tokens. The expert model $p(\cdot)$ then evaluates the exact probabilities of these tokens in parallel. Each token is sequentially accepted or rejected based on an acceptance probability of $\min\left(1, \frac{p(x)}{q(x)}\right)$. If a token is rejected, SD resamples from an adjusted distribution $[p - q]_+$, where $[\cdot]_+$ represents $\text{norm}(\max(0, \cdot))$. After this step, a new batch of L tokens is generated by $q(\cdot)$, and the process repeats. The complete procedure is outlined in Algorithm 1 and Algorithm 2. Notably, this approach ensures that the base distribution $p(\cdot)$ is precisely recovered while sampling from the draft distribution $q(\cdot)$, enabling significant acceleration of AR models without compromising output quality.

2.3. Speculative Jacobi Decoding

While the original SD [10] uses a separate, lightweight model to generate the draft, this is not a strict requirement. Since the correctness of the output distribution is ensured regardless of the choice of q, other methods with more efficient output estimation can also be used for drafting. From this perspective, Speculative Jacobi Decoding (SJD) [22] introduces an alternative approach by leveraging early predictions from a Jacobi iteration [18] as the draft distribution. Specifically, SJD starts by randomly initializing L tokens $x_{0:L}$ and performs a parallel forward pass—similar to the parallel computation in SD—to estimate the output distribution $p(x)_{0:L}$. The tokens are then sequentially verified using the same verification function, $\text{VERIFY}(\cdot)$. A detailed description of the algorithm is provided in Algorithm 4.

This approach is highly efficient because verification and drafting occur simultaneously within a single parallel inference step, eliminating the need for autoregressive inference of q or training a dedicated draft model. Furthermore, since rejected tokens are reused, they gradually converge toward their correct values over iterations, leading to a higher acceptance rate. As a result, SJD achieves up to a 2.1× speedup across various text-to-image AR models. Due to these advantages, **we use SJD as our baseline and introduce novel ideas on top of it throughout the rest of this paper.**

3. Low Acceptance Rate of SD in Image AR

Although SJD achieves a meaningful speedup, its acceptance rate in SD remains relatively low (around 40%) compared to that in typical text SD (around 70%) [10]. In this section, we provide an in-depth analysis of why the acceptance rate for SD in image AR is relatively low.

3.1. Characteristics of Image AR Decoding

The figure 2 provides a visual representation of *(a)* images generated by the AR model , *(b)* the top-1 probability values of the corresponding patchs and *(c)–(e)* further illustrate the sorted distribution of these top-1 probabilities across all tokens. As shown in the figure, the image AR model frequently assigns low top-1 score (below 5%) to 50%–95% of tokens, depending on the prompt. These probabilities are nearly uniformly distributed, suggesting that the model considers multiple tokens as equally plausible next steps. We attribute this phenomenon to two key factors:

- **Redundancy in visual tokens :** Unlike discrete text tokens, visual tokens are derived through vector quantization [25] from a continuous latent space. Although quan-

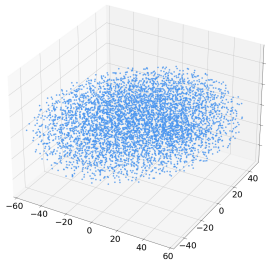


Figure 3. t-SNE[26] 3D visualization of the visual token embedding of [12]

Figure 4. Result with 50% random replacement during decoding, having CLIP_score = 32.089.

tized, these visual tokens still contain significant redundancy in low-frequency components, differing primarily in subtle, high-frequency details that are often imperceptible to the human eyes [34].

- **Diversity of image patches :** Unlike text, which is constrained by syntax and grammar, images can accommodate multiple valid visual patterns. For example, in Figure 2.(b), the model assigns lower top-1 probabilities to highly variable regions, such as hair, which can differ significantly in shape and texture. In contrast, more structured regions, like faces, tend to have higher top-1 probabilities due to their relatively consistent features.

In Figure 4, we present an additional experiment where 50% of the tokens are randomly replaced with their Top-100 candidates during decoding. As shown, the overall image quality remains largely unaffected, suggesting that the model indeed considers multiple valid next-step tokens and that substituting them has minimal impact. However, these findings raise a crucial question: Intuitively, if the model $p(x)$ is indifferent to many possible token choices, the draft model shouldn't need to be highly accurate, which should, in turn, lead to higher acceptance rates. **So why does speculative decoding still slow down in image AR?**

3.2. Total Variation Analysis

To explore this question, we start with following definition. From [10], SD's expected acceptance rate α is defined as :

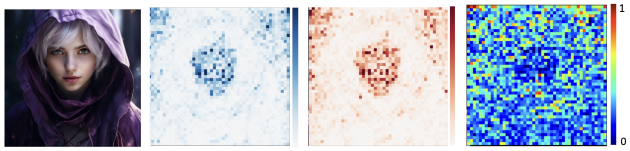
$$\textbf{Definition.}\ \alpha_{p,q} = \mathbb{E}_{x\sim q(x)}\left[\min\left(1, \frac{p(x)}{q(x)}\right)\right]. \quad (1)$$

This directly leads to the following proposition:

Proposition 1 (Acceptance and Total Variation)

let $\alpha_{p,q}$ be acceptance rate defined on Eq.1, and $TV(\cdot,\cdot)$ be Total Variation distance measure, then $\alpha_{p,q} = 1 - \frac{1}{2}\sum_x |p(x) - q(x)| \ = \ 1 - \mathrm{TV}(p,q)$.

Proof. *See Appendix.* A similar analysis has been explored in previous works [10, 30]. As shown, the expected acceptance rate for each token is determined by the absolute difference between the two probability distributions: the expert model p and the draft model q.



(a) Image (b) $p(x)$ Top-1 (c) $q(x)$ Top-1 (d) $TV(p,q)$

Figure 5. Visualization of image, $p(x)$ Top-1, $q(x)$ Top-1 and $TV(p,q)$. While both p, q have small Top-1, their TV is high.

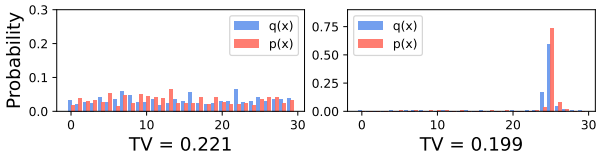


Figure 6. A toy example of the accumulation problem. While the left tends to have an almost uniform distribution, its total variation (TV) is larger than that of the right. This problem becomes more pronounced when the support is larger.

In Figure 5, we present *(a)* a generated image, *(b)* the top-1 probability of $p(x)$, *(c)* the top-1 probability of $q(x)$, and *(d)* the total variation $\mathrm{TV}(p,q)$ for each patch. Notably, even when both distributions $p(x)$ and $q(x)$ agree that multiple tokens are plausible—indicated by low top-1 probabilities—the total variation between them remains high. Conversely, regions with higher confidence (e.g., faces) exhibit relatively low TV values. These results suggest that although many tokens are valid as the next choice, **the accumulation of subtle differences between distributions p and q significantly increases total variation**, ultimately lowering the acceptance rate. To illustrate this concept more clearly, we provide a toy example in Figure 6. In practice, this effect is further amplified by the large vocabulary size—approximately 20,000 tokens—used in recent AR image models [12, 21].

4. Method: Grouped Speculative Decoding

To overcome this issue, we introduce a novel approach called *Grouped Speculative Decoding* (GSD), which bases acceptance decisions on **semantically meaningful tokens** rather than solely selecting the most likely token.

Specifically, let $\mathcal{X}$ represent the token vocabulary, and let $\mathrm{p}(x)$ and $\mathrm{q}(x)$ be the probability mass functions (p.m.f.) of the expert and draft models, respectively, defined over $\mathcal{X}$. We then partition $\mathcal{X}$ into *disjoint clusters*:

$$\mathcal{C} \ = \ \{C_1,\ C_2,\ \ldots,\ C_K\}, \quad (2)$$

where $C_i \cap C_j = \varnothing$ for $i \neq j$ and $\bigcup_{i=1}^{K} C_i = \mathcal{X}$. For each

cluster C_i, we define the grouped probability mass as:

$$\mathrm{p}'(C_i) = \sum_{x \in C_i} \mathrm{p}(x), \quad \mathrm{q}'(C_i) = \sum_{x \in C_i} \mathrm{q}(x). \tag{3}$$

Consequently, p' and q' are also p.m.f. defined over $\mathcal{C}$, satisfying $\sum_{C_i} \mathrm{p}'(C_i) = 1$, with a similar condition holding for q'. We denote by $C(\cdot)$ the mapping $\mathcal{X} \to \mathcal{C}$ that assigns each token x to its corresponding cluster C_i. The specific implementation of $C(\cdot)$ will be detailed in the next section.

In GSD, we use the identical SD verification procedure but decide to accept a token x if its cluster $C(x)$ satisfies:

$$\min\Big(1, \frac{\mathrm{p}'\big(C(x)\big)}{\mathrm{q}'\big(C(x)\big)}\Big) \geq r, \quad r \sim \mathcal{U}[0,1]. \tag{4}$$

If rejected, we follow the same resampling strategy as in standard SD. Intuitively, by summing individual masses within each cluster, we obtain a more coarse-grained probability distribution, smoothing out subtle differences between p and q. This reduces $\mathrm{TV}(\mathrm{p}', \mathrm{q}')$ and thereby improves the acceptance rate. We formally state this in the theorem below:

Theorem 1. Lower Bound on GSD Accept Rate

Let p, q be p.m.f defined over $\mathcal{X}$, let $\mathcal{C}$ is disjoint clstuer defined by Eq. (2) and let p', q' be the corresponding grouped p.m.f defined by Eq. (3). Then for any choice of $(p, q, \mathcal{C})$, the acceptance rate of GSD is bounded below by the acceptance rate of standard SD, meaning that $\alpha_{GSD} \geq \alpha_{SD}$

Proof Sketch. From Proposition 1, to show $\alpha_{GSD} > \alpha_{SD}$, we sufficient to show $\mathrm{TV}(p', q') \leq \mathrm{TV}(p, q)$. By Eq. (3), the total variation of p', q' and p, q can be expaned as :

$$\begin{aligned}
\mathrm{TV}(p', q') &= \frac{1}{2} \sum_{C_i} \big| p'(C_i) - q'(C_i) \big| \\
&= \frac{1}{2} \sum_{C_i} \Big| \sum_{x \in C_i} \big(p(x) - q(x) \big) \Big|. \\
\mathrm{TV}(p, q) &= \frac{1}{2} \sum_{x \in \mathcal{X}} \big| p(x) - q(x) \big| \\
&= \frac{1}{2} \sum_{C_i} \sum_{x \in C_i} \Big| p(x) - q(x) \Big|.
\end{aligned}$$

Because of the triangle inequality, for any C we have,

$$\Big| \sum_{x \in C_i} \big(p(x) - q(x) \big) \Big| \leq \sum_{x \in C_i} \big| p(x) - q(x) \big|.$$

Summing it over all C, we have $\mathrm{TV}(p', q') \leq \mathrm{TV}(p, q)$ □

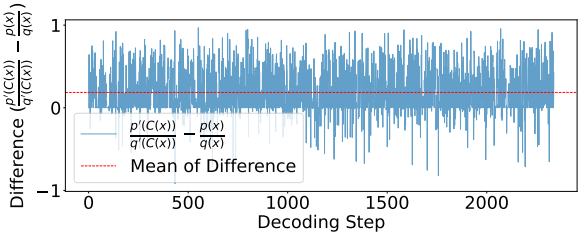


Figure 7. Diffrence of acceptance proability ($\frac{p'(C(x))}{q'(C(x))} - \frac{p(x)}{q(x)}$) during image AR decoding.

As shown, while $\frac{p'(C(x))}{q'(C(x))} \geq \frac{p(x)}{q(x)}$ may not hold for every individual token x, its expected rate is still higher. To support our theoretical anlysis, we depicts the probability difference during image AR decoding in Fig. 7. As shown, most tokens have greater acceptance probability in cluster level and mean difference is actually positive value, validating our theoretical result.

4.1. Context-aware Dynamic Clustering

The key takeaway from Theorem 1 is that GSD improves or at least preserves decoding speed regardless of how clusters are formed. However, designing an *effective* clustering strategy remains critical for maintaining output quality.

How can we form effective clusters? A straightforward approach is to cluster tokens based on pairwise distances among their corresponding codebook embeddings in the embedding space. However, we observed that this strategy frequently underperforms, as shown in Table 2. We attribute this primarily to two reasons:

- **Uniformity of the codebook embedding space :** Figure 3 shows a t-SNE [26] 3D visualization of the visual token embeddings obtained from [12]. As illustrated, embeddings are distributed nearly uniformly, making it difficult to construct semantically coherent clusters. This phenomenon becomes more pronounced as the codebook have higher utilization [34].
- **Impact of token context in image :** Figure 8 visualizes decoded results during generation using varying numbers of rows. Although tokens remain identical , their decoded RGB representations significantly differ, particularly in colors and fine details, highlighting the substantial influence of context. This effect becomes more pronounced when fewer tokens are present.

These observations offer an important insight: the raw embedding space does not fully represent a semantic manifold for image tokens. Instead, semantic meanings emerge when considering the token context, as thousands of tokens jointly input to the decoder.

Context-aware Dynamic GSD. Inspired by this insight, we propose leveraging the token probability $p(x)$ itself as a dynamic *measure* of token similarity. This choice is moti-

(a) 20% (b) 50% (c) 80%

Figure 8. Visualization of decoded results during generation with different numbers of rows. Even though tokens are identical, the resulting RGB images differ in details and color, indicating that accurate clustering should consider the context of nearby tokens.

vated by the fact that the model's predicted probability inherently reflects its perception of token similarity within the current decoding context.

Specifically, at each decoding step t, tokens are sorted according to their probability values, and clusters are formed by grouping the number of G nearest tokens. To avoid grouping tokens with significantly different semantics, we further exclude tokens whose embedding distances or probability differences exceed predefined thresholds d and δ. In Algorithm. 3, we show psuedo-code of our Dynamics GSD verification. The complete detailed algorithm for our final method is provided in the Algorithm. 5.

Importantly, as established by Theorem 1, even if the clustering function dynamically changes at each decoding step t, GSD still guarantees a higher acceptance rate than standard SD. This dynamic clustering not only improves the final image quality but also accelerates the decoding speed, as we will demonstrate in the next experimental section.

Algorithm 3 VERIFY_GSD(X, p, q, G)

Require: Draft $\hat{X}_{0:L}$, Verifier : $p_{0:L}(\cdot)$, Drafter : $q_{0:L}(\cdot)$, Group size G, Embedding distance matrix M_d, thresholds d, δ

for $k = 0$ to L **do**
  ▷ Dynamic Clustering with $p(\cdot)$
  $p\_sort_{vals}, p\_sort_{idx} \leftarrow \texttt{sort}(p_k)$
  $idx \leftarrow \texttt{find-idx}(p\_sort_{vals}, p_k(\hat{X}_k))$
  $C_{idxs} \leftarrow p\_sort_{idx}[idx - G//2 : idx + G//2]$
  $Cvals \leftarrow p_k[C_{idxs}]$

  ▷ Filter Outliers
  for cv, ci in $[Cvals, C_{idxs}]$ **do**
    if $|cv - p_k(\hat{X}_k)| > \delta$ then C_{idxs}.pop(ci)
    if $M_d[\hat{X}_k, ci] >$d then C_{idxs}.pop(ci)
  end for

  ▷ Verification with Cluster Probability
  $p'_C \leftarrow \texttt{sum}(p_k[C_{idxs}])$
  $q'_C \leftarrow \texttt{sum}(q_k[C_{idxs}])$
  if not $r \sim \mathcal{U}[0,1] \leq \min\left(1, \frac{p'_C}{q'_C}\right)$ **then**
    $x \sim [p_k - q_k]_+$, $\hat{X}_k \leftarrow x$, **break**.
  end if
end for
return $\hat{X}_{0:k}, k$

5. Experiments

For our experiments, we employed the SOTA text-to-image AR Image model, Lumina-mGPT [12]. Specifically, we used the standard 7B model and conducted experiments at a resolution of 768×768. In all experiments, we followed the default settings: Top-K sampling with $K = 2000$ and temperature $\tau = 1$. For GSD, we set $d = 0.5$, $\delta = 0.15$, and $L = 16$. We did not conduct experiments with the Greedy setting ($\tau = 0$), as it significantly degrades the quality of generated images [12, 22]. We used PyTorch 2.3 [14] on a high-performance server equipped with 8 RTX 3090 GPUs and an AMD EPYC 7402 processor.

Qualtitiatve Experiments We first present qualitative experiments comparing generated images from Vanilla AR, SJD [22], and our proposed method. To thoroughly evaluate the robustness and effectiveness of our approach, we carefully select five prompts designed to capture a wide range of environmental conditions and generation challenges.

Specifically, in Figure 9, we illustrate the following cases: (1) Unrealistic images, showcasing the model's ability to generate novel and imaginative content in a zero-shot manner. (2) Realistic images, which require precise rendering of complex contextual details. (3) Human faces, a particularly challenging category where even minor discrepancies in facial features are easily noticeable. (4) Animation-style illustrations, highlighting the method's adaptability to stylized visual content. Detailed prompts corresponding to each case are listed in the appendix. As clearly demonstrated in the figure, our proposed method consistently achieves superior image quality across all prompt categories while delivering an impressive 3.5× speedup compared to Vanilla AR decoding.

Quantitative Experiment To quantitatively evaluate both the generation quality and speed of our method, we conducted experiments on two datasets: Parti-Prompt [32] and MS-COCO [11]. For Parti-Prompt, we evaluated the generation quality using the CLIP Score [16], which measures the similarity of images to a given prompt, on a set of 1,600 text prompts. For MS-COCO, we used 5,000 prompts from the validation set and measured both the CLIP Score and the FID Score [11], where the latter was computed by comparing the quality of generated images against the validation set images. To evaluate generation speed, we measured the average latency required to generate a single image and the number of function evaluations (NFE), which

Figure 9. Qualitative experiment. Our GSD shows on average 3.6x NFE acceleration while maintaining image quality

indicates the number of model forward passes performed.

We benchmarked our method against three baselines: (A) *Vanilla AR*, (B) Lossless methods, including Jacobi Decoding [18] and SJD [22], and (C) Naive Lossy SD, such as Amplify $\left(\frac{k \cdot p}{q}\right)$ and Addition $\left(\frac{p+\epsilon}{q}\right)$ introduced in [10, 30], along with (D) **Ours**. For (C) and (D), we applied a lossy acceptance criterion atop SJD [22].

As shown in Table 1, when $G = 3$, our method achieves higher performance with fewer NFEs compared to both (A) Vanilla and (B) Lossless. This indicates that GSD's clustering effectively smooths minor fluctuations in the next-token distribution $p(x)$, positively influencing overall image quality by reducing the noise in model predictions. When we allow slight quality degradation ($G = 50$), our approach achieves a 3.6× speedup compared to Vanilla and a 1.63× speedup over SJD, enabling highly efficient speculative decoding in image AR. Furthermore, compared to naive lossy methods (C), which drastically degrade image quality, our method gives large acceleration with minimal quality loss.

5.1. Ablation Stuides

Pareto-front Comparison In Figure 10, we expand on the experiments presented in Table 1 by evaluating performance across a wider range of group sizes and visualizing these results using a Pareto front plot. As illustrated, our method achieves a superior Pareto front compared to naive lossy SD methods and notably also surpasses the lossless SJD by delivering higher performance with fewer NFEs. These findings indicate that GSD positively influences image quality by smoothing probabilities among similar tokens, thereby improving overall performance. Moreover, naive lossy methods suffer from drastic degradation due to bias or exploding behavior when $q(x)$ becomes small, as they only increase the numerator. In contrast, our method increases both p and q in an unbiased manner, effectively avoiding such issues and better preserving performance.

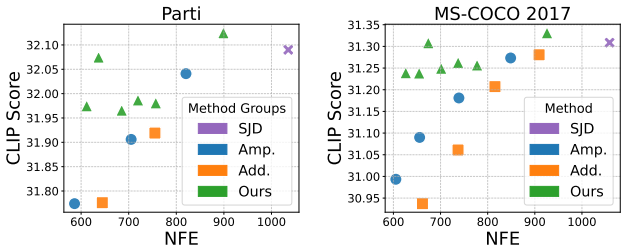


Figure 10. Pareto-front Comparision: NFE vs CLIP_score

Table 1. Quantitative evaluation on the MS-COCO 2017 and Parti-prompt.

Configuration		Latency (↓)	NFE (↓)	Acceleration (↑)		FID (↓)	CLIP-Score (↑)
				Latency	NFE		
Parti-prompt							
A	Lumina-mGPT [12]	112.29s	2392	1.00×	1.00×	–	32.091
B	Jacobi Decoding [18]	116.31s	2300.0	0.97x	1.04×	–	32.091
B	SJD [22]	52.34s	1035.3	2.15x	2.31×	–	32.090
C	Amplify (k=2)	37.48s	705.13	3.00x	3.39×	–	31.906
C	Amplify (k=3)	31.31s	586.23	3.59x	4.08x	–	31.774
C	Addition (ϵ=1e-1)	40.01s	755.05	2.81x	3.17x	–	31.919
C	Addition (ϵ=3e-1)	24.93s	644.57	4.50x	3.71x	–	31.776
D	Ours (G=3)	47.12s	**898.97**	**2.38x**	**2.66x**	–	**32.125**
D	Ours (G=50)	24.13s	**636.75**	**4.65x**	**3.76x**	–	**32.075**
D	Ours (G=100)	23.01s	**611.75**	**4.88x**	**3.91×**	–	**31.975**
MS-COCO 2017							
A	Lumina-mGPT [12]	122.45s	2379	1.00×	1.00×	30.79	31.308
B	Jacobi Decoding [18]	121.16s	2312	1.01x	1.03x	30.78	31.308
B	SJD [22]	55.26s	1058.6	2.22x	2.25×	30.78	31.308
C	Amplify (k=2)	35.98s	738.96	3.40x	3.22x	34.79	31.18
C	Amplify (k=4)	32.29s	635.17	3.79x	3.75x	40.05	30.99
C	Addition (ϵ=1e-1)	47.80s	909.12	2.56x	2.62x	32.75	31.2707
C	Addition (ϵ=3e-1)	31.20s	661.66	3.92x	3.60x	40.20	30.937
D	Ours (G=3)	48.28s	**925.89**	**2.54x**	**2.57x**	**31.50**	**31.33**
D	Ours (G=10)	33.79s	**701.35**	**3.62x**	**3.39x**	**33.12**	**31.25**
D	Ours (G=25)	32.52s	**674.04**	**3.77x**	**3.53×**	**33.55**	**31.24**

Table 2. Comparison of different clustering on Parti.

Method	NFE	CLIP Score
Baseline (SJD [22])	1035	32.090
(A) Embed distance	913.80	32.081
(B) draft $q(x)$	685.31	31.791
(C) expert $p(x)$	636.24	32.056
(D) Ours	636.75	32.075

Effect of Cluster In Table 2, we evaluate the performance of GSD using three different clustering methods. As shown, (A) clustering solely based on static embedding distance fails to achieve speedup, indicating that probability is not strongly correlated with static embedding distance. (B) Clustering based on the draft $q(x)$ leads to speedup but results in a significant decline in the CLIP score, indicating that q does not accurately capture token similarity when contexts are given. (C) When expert $p(x)$ is used for clustering, it successfully preserves the CLIP score while achieving acceleration. (D) Additionally, when filtering out tokens that exceed a certain threshold, performance improves.

6. Related Works: Speculative Decoding

After the pioneering work [10], numerous studies have attempted to improve acceptance rates in SD. Mainstream approaches enhance acceptance while maintaining sampling exactness by proposing multiple drafts in batches [20, 30] or structuring them as trees [4, 6]. Recently, a few studies [9, 24, 33] have explored relaxing the exact sampling constraints to further boost acceptance rates. Among these, LANTERN [9] proposes enlarging the numerator in the acceptance criterion by incorporating neighboring tokens' probabilities, similar to our approach. However, this method has several issues, such as requiring training and lacking a detailed analysis of image AR acceptance rates. Critically, since it increases only the numerator, it shares the same drawbacks as *naive lossy* methods—bias and the exploding problem—achieving only a 1.6× speed-up.

7. Conclusion

In this work, we propose **Grouped Speculative Decoding (GSD)**, a novel training-free SD method specifically designed for image AR. We first thoroughly analyze the pri-

mary challenge of SD in image AR and identify that the core issue arises from high-entropy predictions of the next token caused by the inherent variability of the images. Motivated by this, we introduce GSD, which effectively improves the acceptance rate of SD by smoothing the probability distributions through clustering semantically similar tokens. Furthermore, we observe that naive clustering relying on static embedding distances yields suboptimal outcomes, leading us to propose a dynamic clustering approach. Experimental results demonstrate that GSD achieves an average speed-up of $3.8\times$ while maintaining image quality.

Acknowledgments This work was supported by IITP and NRF grant funded by the Korea government(MSIT) (No. RS-2019-II191906, RS-2024-00415602) and Samsung Research Global AI Center.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1

[2] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025. 1

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 1

[4] Oscar Brown, Zhengjie Wang, Andrea Do, Nikhil Mathew, and Cheng Yu. Dynamic depth decoding: Faster speculative decoding for llms. *arXiv preprint arXiv:2409.00142*, 2024. 8

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2

[6] Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*, 2024. 8

[7] Zigeng Chen, Xinyin Ma, Gongfan Fang, and Xinchao Wang. Collaborative decoding makes visual auto-regressive modeling efficient. *arXiv preprint arXiv:2411.17787*, 2024. 2

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1

[9] Doohyuk Jang, Sihwan Park, June Yong Yang, Yeonsung Jung, Jihun Yun, Souvik Kundu, Sung-Yub Kim, and Eunho Yang. Lantern: Accelerating visual autoregressive models with relaxed speculative decoding. *arXiv preprint arXiv:2410.03355*, 2024. 2, 8

[10] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR, 2023. 2, 3, 4, 7, 8

[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014. 6

[12] Dongyang Liu, Shitian Zhao, Le Zhuo, Weifeng Lin, Yu Qiao, Hongsheng Li, and Peng Gao. Lumina-mgpt: Illuminate flexible photorealistic text-to-image generation with multimodal generative pretraining. *arXiv preprint arXiv:2408.02657*, 2024. 1, 2, 4, 5, 6, 8

[13] Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41 (8):1099–1106, 2023. 1

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[15] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 1

[16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 6

[17] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 1, 2

[18] Yang Song, Chenlin Meng, Renjie Liao, and Stefano Ermon. Accelerating feedforward computation via parallel nonlinear equation solving. In *International Conference on Machine Learning*, pages 9791–9800. PMLR, 2021. 3, 7, 8

[19] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024. 1, 2

[20] Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. Spectr: Fast speculative decoding via optimal transport. *Advances in Neural Information Processing Systems*, 36:30222–30242, 2023. 8

[21] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024. 1, 2, 4

[22] Yao Teng, Han Shi, Xian Liu, Xuefei Ning, Guohao Dai, Yu Wang, Zhenguo Li, and Xihui Liu. Accelerating auto-regressive text-to-image generation with training-free speculative jacobi decoding. *arXiv preprint arXiv:2410.01699*, 2024. 2, 3, 6, 7, 8, 11

[23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1

[24] Vivien Tran-Thien. An optimal lossy variant of speculative decoding, 2023. 8

[25] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 2, 3

[26] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (11), 2008. 4, 5

[27] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, et al. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*, 2023. 1

[28] Yuqing Wang, Shuhuai Ren, Zhijie Lin, Yujin Han, Haoyuan Guo, Zhenheng Yang, Difan Zou, Jiashi Feng, and Xihui Liu. Parallelized autoregressive visual generation. *arXiv preprint arXiv:2412.15119*, 2024. 2

[29] Zili Wang, Robert Zhang, Kun Ding, Qi Yang, Fei Li, and Shiming Xiang. Continuous speculative decoding for autoregressive image generation. *Advances in Neural Information Processing Systems*, 2024. 2

[30] Ming Yin, Minshuo Chen, Kaixuan Huang, and Mengdi Wang. A theoretical perspective for speculative decoding algorithm. *Advances in Neural Information Processing Systems*, 37:128082–128117, 2024. 4, 7, 8

[31] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021. 2

[32] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022. 6

[33] Meiyu Zhong, Noel Teku, and Ravi Tandon. Speeding up speculative decoding via approximate verification. *arXiv preprint arXiv:2502.04557*, 2025. 8

[34] Lei Zhu, Fangyun Wei, Yanye Lu, and Dong Chen. Scaling the codebook size of vqgan to 100,000 with a utilization rate of 99%. *arXiv preprint arXiv:2406.11837*, 2024. 4, 5

Grouped Speculative Decoding for Autoregressive Image Generation

Supplementary Material

S1. Proof of proposition 1

$$
\begin{aligned}
\alpha_{p,q} &= \mathbb{E}_{q(x)}\left[\min\left(1, \frac{p(x)}{q(x)}\right)\right] \\
&= \sum_x q(x) \min\left(1, \frac{p(x)}{q(x)}\right) \\
&= \sum_x \begin{cases} p(x), & \text{if } p(x) < q(x) \\ q(x), & \text{otherwise} \end{cases} \\
&= \sum_x \min(p(x), q(x)) \\
&= \sum_x \frac{p(x) + q(x) - |p(x) - q(x)|}{2} \\
&= \frac{1}{2}\left(\sum_x p(x) + \sum_x q(x) - \sum_x |p(x) - q(x)|\right) \\
&= \frac{1}{2}\left(1 + 1 - \sum_x |p(x) - q(x)|\right) \\
&= 1 - \frac{1}{2}\sum_x |p(x) - q(x)| \\
&= 1 - TV(p, q)
\end{aligned}
$$

S2. Full algorithm of GSD

In this section, we provide detailed pseudocode for the implementation of GSD in Algorithm 5 and 3. Since GSD is built upon SJD, it operates by simply replacing the VERIFY(·) part with our VERIFY_GSD(·). For a more detailed implementation, please refer to the source code.

Algorithm 4 Speculative Jacobi Decoding[22]

Require: Speculative Length L, maximum seq length N, expert model p_θ, initial context $X_{0:n_0}$

$k \leftarrow L$, $n \leftarrow n_0$
while $n < N$ **do**
  $q_{L-k:L}, \hat{X}_{L-k:L} \sim$ `Rand-init`$(\cdot)$
  parallel for $j = 0$ to L ▷ Parallel Verify
    $p_j \leftarrow p_\theta(\cdot \mid [X_{0:n}, \hat{X}_{0:j}])$
  end for
  $(\hat{X}_{0:k},\ k) \leftarrow$ `VERIFY`$(\hat{X}_{0:L}, p_{0:L},\ q_{0:L})$
  $X_{n:n+k-1} \leftarrow \hat{X}_{0:k}$ ▷ Accept.
  $q_{0:L-k} \leftarrow p_{k:L}$, $\hat{X}_{0:L-k} \leftarrow \hat{X}_{k:L}$ ▷ Draft update
  $n \leftarrow n + k$
end while
return X

Algorithm 5 Grouped Speculative Decoding

Require: Speculative Length L, maximum seq length N, expert model p_θ, initial context $X_{0:n_0}$, Group size G, Embedding distance matrix M_d, thresholds d, δ

$k \leftarrow L$, $n \leftarrow n_0$
while $n < N$ **do**
  $q_{L-k:L}, \hat{X}_{L-k:L} \sim$ `Rand-init`$(\cdot)$
  parallel for $j = 0$ to L ▷ Parallel Verify
    $p_j \leftarrow p_\theta(\cdot \mid [X_{0:n}, \hat{X}_{0:j}])$
  end for
  $(\hat{X}_{0:k},\ k) \leftarrow$ `VERIFY_GSD`$(\hat{X}_{0:L}, p_{0:L},\ q_{0:L}, G)$
  $X_{n:n+k-1} \leftarrow \hat{X}_{0:k}$ ▷ Accept.
  $q_{0:L-k} \leftarrow p_{k:L}$, $\hat{X}_{0:L-k} \leftarrow \hat{X}_{k:L}$ ▷ Draft update
  $n \leftarrow n + k$
end while
return X

S3. Additional Results

In this section, we present additional experiments expanding upon the visualizations discussed in the main text.

Top-1 probabilities In Fig. S1, we illustrate the visualization of Top-1 probabilities across a wider variety of images. As shown, regardless of the prompts, many images exhibit numerous tokens with low Top-1 probability distributions.

Visual quality comparison In Fig. S3, we visually illustrate the differences in generation quality among various methods compared in Table 1. As shown in the figure, our GSD achieves approximately a 4× speed-up while maintaining generation quality comparable to lossless methods such as vanilla AR and SJD. In contrast, the naive lossy method also achieves acceleration but significantly degrades generation quality.

GSD generation performance Fig. S2 presents further qualitative results of our method when accelerated by an average factor of 3.6. As demonstrated in the figure, our GSD significantly accelerates AR image decoding while maintaining generation quality across diverse prompts.

S4. Prompts on Qualititave Experiment

In Figure. 9 on main paper, the prompts for each images are as follows :

- *Rusty robot on a skateboard in the hallway of domitory, photography, 4k, realistic, detailed, bright*
- *Origami astronaut, waliking in the cloud, bright background, realistic, 4k, photography, bright color*
- *photography, realistic, White cute fluffy dog, skyblue background, very intricate, very detailed, realistic., bright*
- *color photo, photography, Face of a young man, very detailed, realistic. sharp, film grain, high contrast*
- *animation art work, cute, cat character, bright color pallette*

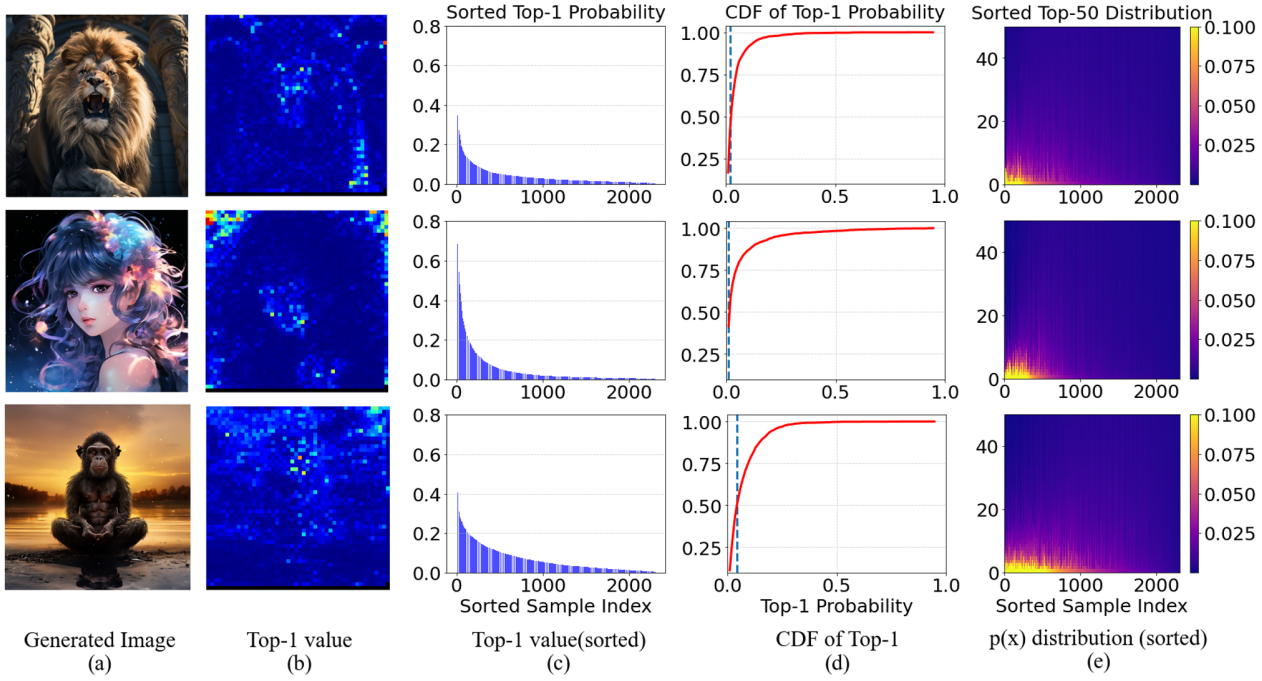


Figure S1. Additional p(x) visualization.

Figure S2. Qualtitiave experiment on various prompt. Our GSD shows on average 3.6x NFE acceleration while maintaing image quality

Figure S3. Qualitative comparison between methods in Table 1 of the main paper