



Hawk: Leveraging Spatial Context for Faster Autoregressive Text-to-Image Generation

Zhi-Kai Chen^{1,2} Jun-Peng Jiang^{1,2} Han-Jia Ye^{1,2} De-Chuan Zhan^{1,2*}

¹ School of Artificial Intelligence, Nanjing University, China

² National Key Laboratory for Novel Software Technology, Nanjing University, China
{chenzk, jiangjp, yehj, zhanc}@lamda.nju.edu.cn

Abstract

Autoregressive (AR) image generation models are capable of producing high-fidelity images but often suffer from slow inference due to their inherently sequential, token-by-token decoding process. Speculative decoding, which employs a lightweight draft model to approximate the output of a larger AR model, has shown promise in accelerating text generation without compromising quality. However, its application to image generation remains largely underexplored. The challenges stem from a significantly larger sampling space, which complicates the alignment between the draft and target model outputs, coupled with the inadequate use of the two-dimensional spatial structure inherent in images, thereby limiting the modeling of local dependencies. To overcome these challenges, we introduce Hawk, a new approach that harnesses the spatial structure of images to guide the speculative model toward more accurate and efficient predictions. Experimental results on multiple text-to-image benchmarks demonstrate a 1.71× speedup over standard AR models, while preserving both image fidelity and diversity.

1 Introduction

The field of image generation has witnessed significant progress [37, 20, 45], especially with the emergence of autoregressive (AR) models that demonstrate strong capabilities in generating high-quality images [48]. These models have been applied in various domains, including super-resolution [13], image completion [43], style transfer [7], image segmentation [28], and image editing [6].

Despite the impressive performance of autoregressive (AR) models in generating high-fidelity images, their slow inference speed remains a significant obstacle for real-world deployment. This limitation primarily stems from their inherently sequential nature—generating images token by token—which leads to low throughput and high latency. While existing approaches, such as non-autoregressive generation methods [14, 20, 38] and model compression techniques like pruning [44, 25] or quantization [2, 50], can accelerate inference, they often do so at the cost of degraded image quality. This raises the question: is it possible to speed up inference without compromising generation fidelity?

Speculative decoding has already demonstrated its effectiveness in accelerating autoregressive text generation models without compromising output quality [19, 33, 12, 10]. This technique employs a two-model framework: a lightweight draft model first generates a sequence of tokens, which are then partially or fully verified by a larger, high-quality target model. If the target model confirms the draft’s predictions, multiple tokens can be accepted in a single step, thus reducing the number of sequential forward passes and accelerating inference. Given its success in the text domain, it is natural to consider applying speculative decoding to AR image generation as a means to achieve faster inference without sacrificing image quality.

*Corresponding author, email: zhanc@lamda.nju.edu.cn.

Although AR-based text and image generation share many underlying principles, key differences between the two domains hinder the direct transfer of speculative decoding techniques to image generation. First, the significantly larger sampling space in image generation presents a key challenge for applying speculative decoding. To capture fine-grained visual details, image generation models typically sample 20 to 400 times more candidates from the vocabulary than text models [35, 23, 34], introducing greater variability in token prediction. This makes it much harder for the draft model to align with the target model, limiting the effectiveness of speculative decoding when applied directly.

From another perspective, unlike sequential text, images are naturally two-dimensional, exhibiting complex spatial dependencies that are largely absent in text generation. Our attention-sinking experiments [47] in Section 4.1 reveal that the generation of a pixel-level token is influenced not only by its horizontal neighbors within the same row, but also by vertical dependencies across rows. These findings suggest that both horizontal and vertical spatial contexts provide informative signals, which can be effectively leveraged to guide and accelerate AR image generation. However, existing speculative decoding algorithm designed for text tends to overlook the potential vertical information, and are not well suited to capture two-dimensional spatial context.

To address the aforementioned challenges, we introduce Hawk, a new framework based on Spatial Speculative Decoding that leverages the two-dimensional spatial structure of images to accelerate autoregressive (AR) image generation. By explicitly incorporating spatial dependencies—both horizontal and vertical—Hawk significantly improves alignment between the draft and target models. This not only expands the effective sampling space but also enables more informed speculative predictions, leading to faster generation without compromising image fidelity.

Specifically, Hawk introduces a pair of draft heads that independently speculate in both horizontal and vertical directions. Given the inherent asymmetry between these two spatial dimensions, the horizontal and vertical heads produce complementary speculative results, particularly enhancing diversity in edge contours and central image regions. We then integrate these outputs to construct a unified spatial sampling space, which serves as the foundation for candidate token selection. These candidates are subsequently verified by the target model. Empirically, we find that this merged spatial space aligns more closely with the target model’s sampling distribution, improving the acceptance rate of speculative tokens. By enabling multiple tokens to be generated and validated in a single pass, Hawk effectively speeds up inference without compromising generation fidelity. Experimental results on multiple text-to-image benchmarks demonstrate a $1.71\times$ speedup over standard AR models. The main contributions of this work are summarized as follows:

- We highlight the importance of accelerating image generation without compromising output quality, and analyze the unique challenges specific to image generation with speculative decoding.
- We propose Hawk, a novel strategy that combines speculative decoding with spatial information derived from the attention mechanism. To the best of our knowledge, this is the first work to leverage spatial information for accelerating autoregressive image generation.
- By integrating optimized Draft Heads and Spatial Speculative Decoding into a large-scale text-to-image model, Hawk achieves a $1.71\times$ speedup without compromising output quality.

2 Related Works

Autoregressive Text-to-Image Generation. Autoregressive (AR) models were initially developed for time series tasks. However, subsequent research revealed that AR models could also be applied to spatial data, such as image generation [31]. Early works, including PixelRNN [41] and PixelCNN [40], leveraged RNNs and CNNs to perform pixel-by-pixel image inference. However, this approach had its limitations. Later, methods such as vector quantization [11] were introduced to convert images into discrete tokens. Early approaches like VQ-VAE [42] explored this concept, and as the capabilities of autoregressive models became more apparent, research shifted toward developing more advanced models based on the autoregressive framework. Related works include [9, 30, 49].

In the era of large language models, multimodal generation, which combines both text and image generation, has become one of the key capabilities of large models. Chameleon [35] was proposed as a multimodal model capable of generating both images and text. Lumina-mGPT [23] is a fine-tuned model based on Chameleon. However, due to the reliance on AR inference, these methods inherently suffer from slow inference speeds.

Speculative Decoding. Speculative Decoding was initially proposed as an optimization strategy [3, 15] and later widely adopted as an acceleration technique in large language model inference [19, 5]. It achieves acceleration while maintaining output quality, making it a promising method for fast inference. Building on the speculative framework, several enhancements have been proposed to address the challenge of difficulty in obtaining drafter models. For instance, Medusa [4] and Eagle [21] introduced mechanisms such as draft heads, enabling the model to perform speculative generation and verification simultaneously within a single inference pass. Additionally, tree-based attention mechanisms [26] have been employed to increase the number of sampling candidates, enhancing the success rate of speculative decoding.

Further advancements include works like Hydra [1] and KOALA [51], which optimize the training of draft heads to enhance speculative decoding performance. Due to the lack of spatial information utilization in image generation, these acceleration algorithms did not achieve the expected performance gains. In our work, we adopt a similar draft head structure to that in Medusa, enhancing its effectiveness by incorporating spatial information.

Autoregressive Image Generation Acceleration. The use of speculative decoding to accelerate image generation remains an underdeveloped area, with no standardized methodology across existing works. Different approaches have been proposed in the literature. For example, the ZipAr [14] adopts a non-autoregressive generation strategy, avoiding the traditional raster scan order. In contrast, the LANTERN++ [18] employs a more relaxed speculative decoding scheme, increasing the acceptance probability of the small model’s speculative outputs. The SJD [36] integrates speculative sampling with Jacobi decoding to achieve acceleration.

In our Hawk method, we propose a spatial speculative sampling method. While preserving the autoregressive inference paradigm, we adopt the original speculative decoding strategy to maintain sampling performance, thereby achieving acceleration without compromising the model’s accuracy.

3 Preliminary

Autoregressive Image Generation. An Autoregressive (AR) model follows a next-token prediction process, where the prediction of x_t depends only on its preceding tokens $(x_1, x_2, \dots, x_{t-1})$. The probability of generating a sequence of discrete tokens can be formulated as: $p(x_1, x_2, \dots, x_T) = \prod_{t=1}^T p(x_t | x_1, x_2, \dots, x_{t-1})$.

In the context of image generation, vector quantization is employed to convert an image into a sequence of tokens. Vector quantization consists of an encoder and a quantizer. The encoder maps an image of shape $x \in \mathbb{R}^{h \times w \times 3}$ into a latent representation $z \in \mathbb{R}^{h \times w \times d}$, where d is the dimension of the latent space. Formally, this can be written as $z = E(x)$. In the process of computing the quantizer, we use a pretrained codebook $\mathcal{Z} = \{z_k\}_{k=1}^K$, consisting of K elements, each of dimension d . The quantizer then searches for the closest codebook vector for each feature element $z_{i,j}$. Specifically, for each feature point, the quantizer computes: $z_q = \arg \min_{z_k} \|z_{i,j} - z_k\|$, where z_k is a vector in the codebook $\mathcal{Z}$. The quantizer’s result is then derived by combining the indices of the nearest neighbors in the codebook for each element in the feature map.

Subsequently, 2D $n \times n$ image tokens are reshaped into a 1D sequence of length n^2 , enabling next-token generation. As a result, AR-based image generation typically follows a raster scan order, which involves sequential inference of the image, row by row and column by column. The most common pattern is from left to right and top to bottom.

Speculative Sampling. The core of speculative decoding lies in its use of the speculative sampling strategy to sample and validate the results generated by the draft model. Suppose the target model has a sampling distribution $P_{\text{target}}(x|x_{<t})$ and the draft model has a sampling distribution $P_{\text{draft}}(x|x_{<t})$ where t represents the current inference step. The process consists of two phases: drafting and verification. In the drafting phase, we use the draft model to generate $\gamma \in \mathbb{Z}^+$ tokens from the draft sampling space $P_{\text{draft}}(x_t|x_{<t})$. However, these tokens are not directly reliable. In the verification phase, we use a single forward pass of the target model to verify the speculative tokens $t+1 : t+n$ in parallel. For each speculative token, the target model has a probability of $\min(1, p_{\text{target}}(x)/p_{\text{draft}}(x))$ to accept the token. If no speculative tokens are accepted, the target model resamples the token with the probability $p'_{\text{target}}(x) = \text{norm}(\max(0, p_{\text{target}}(x) - p_{\text{draft}}(x)))$.

Utilization of Draft Heads Structure. Since speculative sampling relies on a smaller model for speculation, acquiring such a model can be challenging. The smaller model must maintain good consistency with the target model. However, not all models have smaller parameter versions available. In this context, the draft heads structure [4] serves as an alternative to the draft model. Draft heads refer to additional decoding heads incorporated after the final hidden layers to generate speculative outputs. In the Hawk method, we employ spatial draft heads: horizontal draft heads predict the token at position $(T + \text{Horizontal Depth})$, while vertical draft heads predict the token at position $(T + \text{Vertical Depth} \times \text{Image Width})$. Here, T represents the current position in the autoregressive (AR) decoding process, Horizontal Depth refers to the horizontal speculative distance, and Vertical Depth refers to the vertical speculative distance. By leveraging the draft heads structure, we eliminate the need for a separate, smaller speculative model, allowing us to complete both the draft and verification phases in a single forward pass.

Attention Sinking. The concept of “attention sinking” was first introduced in the context of large language decoding [47]. In their study, the authors observed that during the inference process of an autoregressive model, the model’s attention tends to concentrate on the first token and a few tokens close to the current inference point. This indicates that the first token and its neighboring tokens contain critical information for the model’s reasoning. The loss of this information can lead to a significant decline in the model’s performance during inference. To better understand the mechanism of attention sinking in image generation, we introduce two key concepts that will be referenced throughout this paper: inference-neighboring points and spatial-neighboring points. Inference-neighboring points are defined as the set of points that are sequentially adjacent to the current point in raster order (e.g. $x_{T-1}, x_{T-2}, x_{T-3}, x_{T-4}$). Spatial-neighboring points, on the other hand, are defined in terms of the geometric structure of the image. These include the points spatially adjacent to the current point, such as those directly above or to the left of it in the image space (e.g. $x_{T-1}, x_{T-\text{Image width}}, x_{T-2}, x_{T-2 \times \text{Image width}}$).

4 Spatial Speculative Decoding

In this section, we first analyze the unique characteristics of the attention sinking phenomenon in autoregressive image generation and observe its reliance on spatial information. We then introduce our proposed Hawk method.

4.1 Preliminary Experiments

To analyze the differences in inference dependencies between image and text autoregressive generation, we conduct an attention sinking experiment to compare the attention focus of different models during the generation process. This analysis helps us identify potential strategies for inference acceleration. Our experiment compares the average attention logits across different models to highlight their attention patterns during generation. For the Lumina-mGPT model, we generate a 48×48 image and examine the average attention logits at its central point. For the text model, we use LLaMA 2 [39], instructing it to generate the same number of tokens as the image central point while observing the average attention logits at the most recently generated token. Our findings reveal that, in image generation, attention is not solely concentrated on inference-neighboring points but also extends to Spatial-neighboring points within the image space. Figure 1 illustrates this relationship.

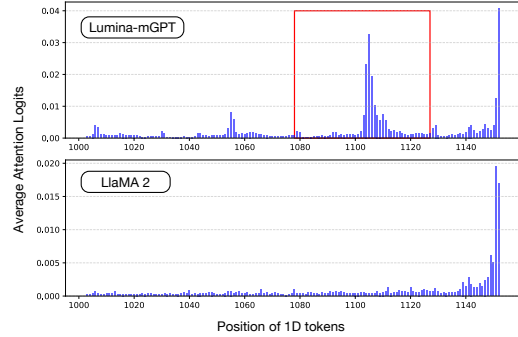


Figure 1: The difference in average attention logits between image generation and text generation, displayed with Lumina-mGPT in a 1D raster order (top), LLaMA 2 (bottom). The average attention logits of the image generation model exhibit a strong dependency on spatially neighboring points. The red box highlights the previous row in the image generation process.

The figure clearly shows that in text generation, attention primarily focuses on inference-neighboring points, with attention scores dropping sharply after approximately five tokens from the current

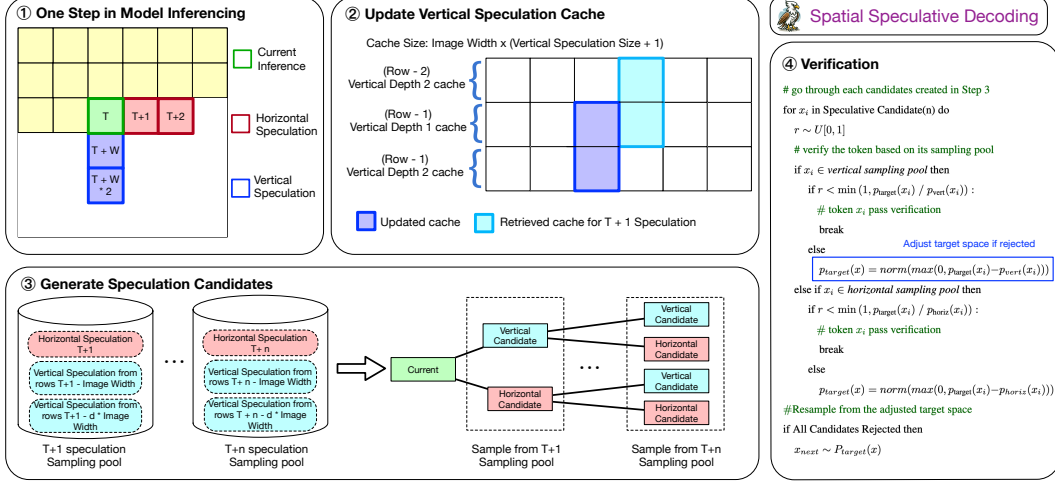


Figure 2: An overview of our Hawk method is presented. During each iteration of the inference process, horizontal and vertical speculations are generated using the draft head. The vertical speculations are stored in the Speculation Cache for future use when processing subsequent lines. Meanwhile, the horizontal speculations are combined with the previous vertical speculations to create the speculation sampling pool. From this pool, tree decoding candidates are generated, followed by a verification step akin to tree speculative decoding.

point. In contrast, for image generation, attention extends beyond the immediate tokens to include spatially adjacent tokens from previous rows, resulting in attention logits with multiple peaks. The red box in the figure highlights the previous row of the image generation. Notably, when inferring the current row, the average attention logits from the previous row surpass those from the current row, suggesting that attention from the preceding row plays a crucial role in image autoregressive inference, even contributing to the majority of the attention. This phenomenon underscores the distinctive dependencies in image generation as compared to text generation.

Speculative sampling employs speculation heads to predict future tokens, but these heads often result in a performance drop compared to the original model’s inference. We attribute this performance degradation to insufficient attention information. For text models, key attention information is primarily concentrated around the current inference point. In text-based speculative generation, when attempting to predict tokens farther away, much of the critical attention information is lost, leading to poor performance. In contrast, for image generation models, important information also resides in the spatially adjacent regions around the speculative point. Therefore, even when speculating on distant points (e.g., $T + \text{Image Width}$), the existing inference information from the current row (highlighted by the red box in our attention sinking experiment) helps mitigate the speculation loss. As a result, the draft heads can still maintain relatively good performance despite the absence of some neighboring information. Additional experiments on the attention sinking phenomenon are provided in Appendix B.

4.2 Enhancing Speculative Decoding with Spatial Information

Analysis. To achieve better speculative inference acceleration, we aim to expand the sampling space of the draft model, increasing its candidate selection options. With a larger sampling space, the likelihood of a smaller model successfully aligning with the sampling distribution of a larger model improves. Simply increasing the number of candidates for speculative decoding in a one-dimensional manner does not inherently enhance sampling diversity, as its effectiveness remains constrained by the limited information available within a single-dimensional space.

Our preliminary experiment demonstrates that directly speculating on tokens below can yield promising results. A potential approach to expand the sampling direction of draft heads is to introduce additional draft heads in speculative sampling, enabling the prediction of the next row or even multiple subsequent rows. By adopting this approach, we effortlessly double the sampling space dimension of the draft model.

Specifically, our strategy consists of three key steps: First, we use dual-direction draft heads to generate an additional sampling space and cache the vertical sample space for use in subsequent inferences. During the speculation process, we combine the cached vertical sampling space from the previous row with the horizontal sampling space at the current speculative point to form a spatial sampling pool, from which we generate our speculative candidates. In this generation process, a tree structure is employed to expand the number of candidates that can be verified simultaneously. Finally, speculative sampling and tree decoding are applied to verify whether the forward pass produces an acceptable speculative result. Figure 2 provides an overview of the method.

Dual Direction Drafting Heads. To more effectively explore the target model’s sampling space, we believe incorporating spatial information is crucial. We use Dual Direction Drafting Heads to simultaneously speculate in both vertical and horizontal directions, thereby expanding the sampling capacity of our draft heads and improving the success rate of our speculation. Let T represent the current inference position along the horizontal axis (e.g., the current image token in a row). Instead of speculating horizontally on positions $T + 1, T + 2, \dots, T + n$, where n is the horizontal speculation length, Hawk also makes predictions for positions further down the vertical axis. Specifically, we introduce an additional drafter heads that target positions of the form:

$$T + IW \times VSD \quad (1)$$

Where *Vertical Speculation Depth* (VSD) refers to the number of rows below the current position that are explored, and *Image Width* (IW) represents the number of tokens per row in the generated image. In practical usage, we need to follow the raster order for image generation. As a result, the outputs from the vertical draft heads are not utilized immediately; instead, we maintain a cache of size

$$IW \times (VSD \times (VSD + 1)) / 2 \quad (2)$$

to store vertically speculative predictions, which will be utilized in the generation of vertical sampling space for subsequent inferences.

Spatial Sampling Pool. To optimize our draft sampling space and better align it with the target sampling space, we build a candidate sampling pool using two-dimensional speculating information at each speculation point. We reuse vertical predictions already computed from cache when speculating horizontally to point $T + n$ in the candidates generation step.

When the model computes speculative outputs for $T + n$, it utilizes vertical predictions from up to Vertical Speculation Length subsequent rows at $T + n - IW \times \Delta_v$ position, where Δ_v represents the subsequent row number. The prediction results at these positions will be combined to form our vertical sampling space.

$$\text{VertSpec}(T + n) = \{\text{Cache}(T + n, \Delta_v, d) \mid d = \Delta_v, \Delta_v \in \{1, 2, \dots, VSD\}\} \quad (3)$$

Here, d represents the depth index of the vertical speculation. Specifically, for the vertical speculation results of the previous k_{th} row, we utilize the speculative result with a depth of k at that position. Although these speculative results have different speculation depths and come from different previous rows, all of them point to the same current inference point. These predictions are retrieved from the cache, which we maintained in the previous inference. For constructing the horizontal sampling space, we directly use the speculation from the current point to the $T + n$ point.

$$\text{HorizSpec}(T + n) = P_{\text{horiz}}(x_{T+n} | x \leq T) \quad (4)$$

When speculation point $T + n$, we combine the vertical sampling space and the horizontal sampling space to form a two-dimensional spatial sampling pool of next-step candidates. We merge:

$$\underbrace{\text{HorizSpec}(T + n)} + \underbrace{\text{VertSpec}(T + n)} \quad (5)$$

This merged set forms the new candidate pool at the speculation location $T + n$, incorporating the candidate tree expansion. The candidate tree is generated using the Cartesian product of the sampling results from the sampling pool at each speculation position. By adopting this approach, each newly expanded node in the candidate tree benefits from both the current horizontal speculation and the cached vertical speculation, enabling a more comprehensive exploration of the image space and resulting in richer, more diverse candidate generation.

Sampling and Verification. The core principle of speculative decoding lies in preserving the distribution of the original model. Our algorithm focuses on validating this design through a

verification-based decoding process. Specifically, we employ a tree-structured decoding strategy (step-3 in Figure 2) to generate and verify speculative drafts.

For each draft result, candidates at the same depth (e.g., $t+1$, $t+2$, $\dots$, $t+n$) are verified sequentially. If a candidate passes verification, decoding proceeds to the next depth; otherwise, the target distribution is updated iteratively. Specifically, if k candidates are consecutively rejected, we perform k updates:

$$p_{\text{target}}^{(i+1)}(x) = \text{norm}(\max(0, p_{\text{target}}^{(i)}(x) - q_{\text{draft}}(x))), \quad i = 0, \dots, k-1, \quad (6)$$

and resample from the last refined distribution $p_{\text{target}}^{(k)}$, where q_{draft} may come from either the horizontal or vertical draft head. Despite involving two distinct draft sampling distributions, our approach still preserves the statistical behavior of the original model (see Appendix A for proof). Formally, our final formulation for approximating the original distribution is expressed as:

$$p(x) = \sum_{i=1}^{a+b} \left(\prod_{j=1}^{i-1} (1 - \alpha_j(x)) \cdot q_i(x) \cdot \alpha_i(x) \right) + \left(\prod_{j=1}^{a+b} (1 - \alpha_j(x)) \right) \cdot p_{\text{resid}}(x) \quad (7)$$

where a and b denote the numbers of samples drawn from the two sampling spaces, and α represents the acceptance rate at each verification step.

Here, $p_{\text{resid}}(x)$ denotes the residual probability of failure after all candidates have been evaluated. In conventional speculative sampling, multiple candidates are drawn from a single latent space, yielding identical candidate distributions. As a result, when all candidates fail verification, the residual probability $p_{\text{resid}}(x)$ cannot be further reduced. In contrast, by introducing multiple sampling heads that operate on distinct spaces, our approach diversifies the verification distributions, thereby mitigating the residual probability and improving overall sampling efficiency.

5 Experiment

Model and Benchmark. We evaluate our proposed Hawk method on the Lumina-mGPT model [23]. For testing, we generate 768×768 images and use top- k sampling with $k = 2000$ and a temperature of 1.0. As for how these parameters affect the generated images, we provide a discussion in Appendix D. We also employ classifier-free guidance [8] with a guidance scale of 3.0. For the benchmark, we use the test sets from COCO 2017 [22] and Flickr30K [29]. From these datasets, we sample 500 examples from each for evaluation.

Metrics. We evaluate inference acceleration using two metrics: Inference Acceleration Rate (*original inference time / accelerated inference time*), where higher is better, and Accept Length, the average number of successful speculative predictions per step. For non-speculative baselines, the accept length is 1. To evaluate image quality, we employed two widely used metrics. The FID [17] score measures the similarity between the generated images and real images, where lower scores indicate better quality. Additionally, the CLIP [16] score assesses the semantic alignment between the generated images and their corresponding text prompts, with higher scores reflecting better alignment. All experiments are conducted on RTX 3090 GPUs.

Training. To train our Spatial Draft Head, we largely follow the same procedure used for Medusa and Lumina-mGPT. We only train the draft head, while keeping the rest of the model frozen. We use the AdamW [24] optimizer with a weight decay of 0.1 and $\beta = (0.9, 0.95)$. The base learning rate is set to 2×10^{-5} . We set the draft head balance weight λ_k to 1. Structural details of the draft heads and memory considerations are provided in Appendix C. We use 6,000 images sampled from the LAION aesthetic training set [32] to fine-tune our drafter head. This training procedure typically takes 8–12 hours to converge on a **single** RTX 3090 GPU.

5.1 Quantitative and Qualitative Evaluation

We evaluate five different methods by analyzing their performance across various test datasets: vanilla autoregressive decoding, Medusa (horizontal draft heads), Hawk with vertical draft heads, Medusa with LANTERN++, and Hawk with spatial draft heads. In this context, spatial draft heads refer to the combination of both vertical and horizontal draft heads.

Table 1: Experiment result for different speculation methods on COCO2017 and Flickr30K validation dataset. We sampled 500 images from the validation set and evaluated the hardware environment of a single RTX 3090 GPU. Spatial Draft heads: a combination of vertical draft heads and horizontal draft heads. For LANTERN++, we use the hyperparameter setting of ($k = 10$, $\lambda = 2$).

Dataset	Method	Speed up	Accepted Length	FID	CLIP Score
COCO2017	Vanilla AR	1.00×	-	90.68	33.43
	Medusa [4]	1.58×	1.728	92.51	33.34
	Hawk (Vertical Draft Heads)	1.58×	1.726	91.46	33.33
	LANTERN++ [18]	1.69×	1.863	93.22	33.08
	Hawk (Spatial Draft Heads)	1.71×	1.890	90.71	33.39
Flickr30K	Vanilla AR	1.00×	-	104.80	34.49
	Medusa [4]	1.55×	1.704	103.51	34.48
	Hawk (Vertical Draft Heads)	1.59×	1.738	104.21	34.45
	LANTERN++ [18]	1.67×	1.842	106.10	34.25
	Hawk (Spatial Draft Heads)	1.69×	1.871	104.50	34.45

Table 1 presents the quantitative results comparing different approaches in the image speculative decoding task. Our vertical and horizontal draft heads demonstrate varying effectiveness depending on the dataset. This discrepancy is likely attributed to the distinct image distribution characteristics of each dataset. Due to varying spatial dependencies within the images across datasets, performance differences arise between the different draft heads. Our Hawk method achieves a $1.71\times$ speed-up compared to the baseline method, aligning with our expectations. Since our sampling space is a combination of vertical and horizontal sampling spaces, its overall performance is constrained by the summation of the performance gains from both heads.

From the perspective of FID (lower is better) and CLIP score (higher is better), our approach achieves substantial acceleration while effectively preserving the image quality of the original model. We compare our method with LANTERN++, an existing multi-draft image speculative decoding acceleration strategy, and integrate their algorithm into our Medusa-based framework for a fair comparison. Our approach achieves a consistently higher accepted length during speculative verification, indicating more efficient and stable speculative decoding. Although the overall acceleration gain is moderate due to the additional computation from dual-directional draft heads, this improvement is obtained without relaxing verification thresholds, maintaining strict adherence to the verification criteria.

In contrast, LANTERN++ adopts a relaxed speculative decoding strategy from $\min\left(1, \frac{q(\hat{x}|s)}{p(\hat{x}|s)}\right)$ to $\min\left(1, \frac{\sum_{x \in A_k} \delta(\hat{x}) q(x|s)}{p(\hat{x}|s)}\right)$ that substantially lowers the verification threshold, increasing the acceptance rate of inaccurate predictions from the small model. This often results in degraded generation quality, as reflected in the fact that LANTERN++ exhibits higher FID and lower CLIP scores compared to Hawk. By maintaining a more conservative and accurate verification process, Hawk achieves superior overall performance in both efficiency and output fidelity.

For the qualitative evaluation, we visually compared images generated by different methods to identify any noticeable differences in quality. This analysis aimed to assess whether the accelerated methods introduced any degradation compared to the baseline or other approaches. Given that our speculative sampling strategy is mathematically grounded, it theoretically preserves the same sampling space as the target model. As shown in Figure 3, we conducted extensive observational experiments on our test set and found no significant degradation in image quality. These results suggest that our Hawk method produces images that maintain the same quality as the original model.

5.2 Effectiveness of Vertical Draft Head

We aim to demonstrate that, compared to traditional horizontal speculative sampling, vertical heads introduce new information that improves the spatial sampling space. We tested performance differences between vertical and horizontal draft heads during training and inferencing.

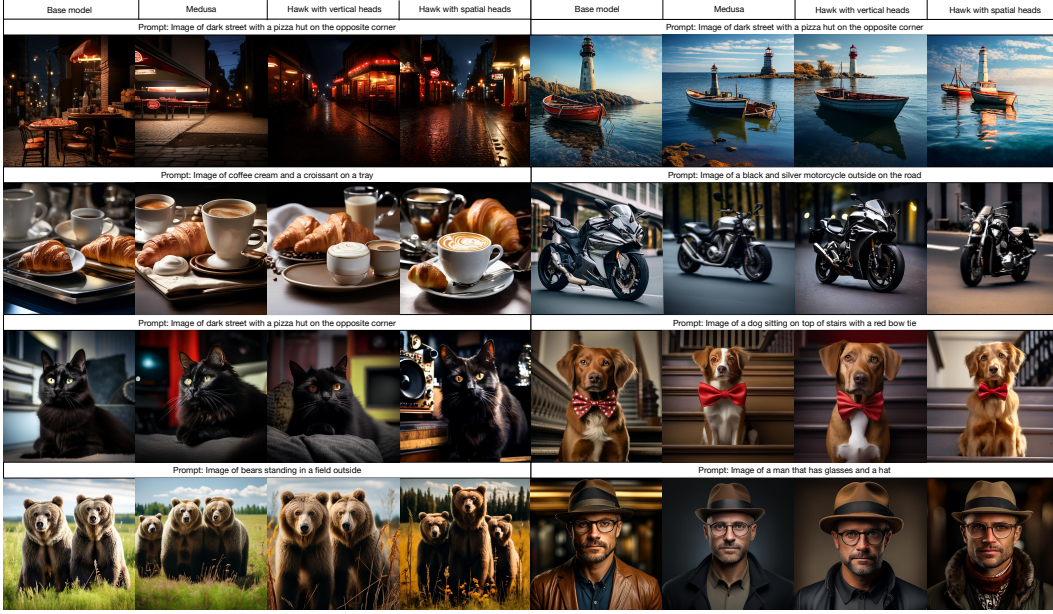


Figure 3: Quality evaluation of Hawk method. For each image type, the images, from left to right, are generated by the base model, Medusa, Hawk with vertical heads, and Hawk with spatial heads. The Hawk method maintains the performance of the baseline model while enhancing inference speed.

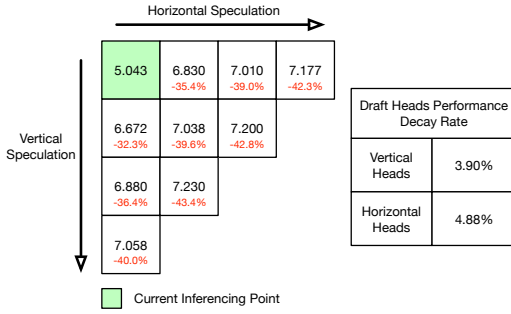


Figure 4: The training loss of draft heads at different locations is related to the current decoding point. Vertical heads experience relatively less performance decay as the speculation depth increases.

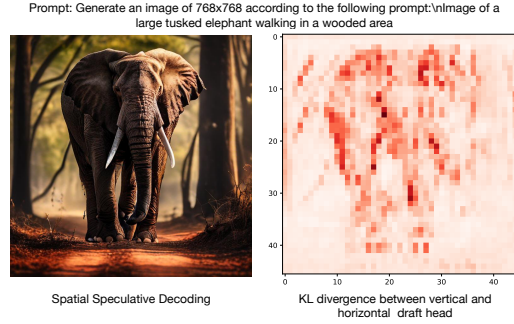


Figure 5: The KL divergence between the vertical and horizontal draft heads during speculative decoding for the given prompt. The difference between the vertical and horizontal draft heads is more pronounced when generating complex areas of the image.

During training, we observed a discrepancy in the speculative capabilities of vertical and horizontal heads. We compared the training losses of draft heads at different positions. Intuitively, since the vertical speculation head is positioned at a distance equal to the image width from the current decoding point—much larger than that of the horizontal head—we would expect the vertical draft head to have a higher training loss. However, our experiments yielded the opposite result. In Figure 4, we show comparisons of training losses of draft heads at various positions.

The green sections represent the fine-tuning loss of the model on the training dataset, while the other sections show the training loss of the speculative draft heads. For speculative points at the same Euclidean distance from the current decoding position, the vertical draft head demonstrates lower training loss. The performance decay rate shown in the figure illustrates this phenomenon. With increasing speculative depth, the vertical direction shows a smaller performance decay rate (3.90%) compared to the horizontal direction (4.88%). Our experiments reveal that, ideally, vertical heads should produce better speculative results than horizontal heads, even though they involve a significantly larger speculative distance.

In Figure 5, we compare the Kullback–Leibler (KL) divergence between horizontal and vertical draft heads during inferencing. The plot reveals that the divergence gap becomes more pronounced when generating fine-grained details, such as an elephant’s eyes, nose, teeth, and overall contour. These regions, due to their inherent ambiguity, permit multiple potential outcomes, resulting in a broader sampling space for the target model and greater KL differences between vertical and horizontal heads. Conventional draft-based speculative sampling struggles to align with the target model in such cases. In contrast, our Hawk method leverages integrated sampling spaces, with additional contributions from the vertical head, which broadens the range of speculative choices and significantly increases the acceptance rate of our sampling strategy.

To further quantify the benefits of this increased diversity among draft heads, we analyze its theoretical impact on sampling efficiency. As shown in Figure 6, we illustrate the theoretical speedup effects introduced by the diversity among different draft heads. We evaluate the theoretical speculative rejection probability $p_{\text{resid}}(x)$ (see Eq. 7), where a lower $p_{\text{resid}}(x)$ indicates greater potential for acceleration. The term $p_{\text{resid}}(x)$ represents the residual probability of resampling after rejection.

As defined in Eq. 6, the residual distribution $p_{\text{target}}^{(i+1)}(x)$ is recursively updated by subtracting the draft distribution $q_{\text{draft}}(x)$ from the target distribution. When the same draft distribution $q_{\text{draft}}(x)$ is repeatedly used across iterations, the residual mass converges to a fixed point where further subtraction yields no meaningful reduction—hence $p_{\text{resid}}(x)$ cannot further converge. However, by introducing multiple sampling spaces with distinct draft distributions, the residuals at each step are computed with respect to different $q_{\text{draft}}^{(j)}(x)$, enabling additional reduction of the overall $p_{\text{resid}}(x)$ and thus yielding greater theoretical speedup.

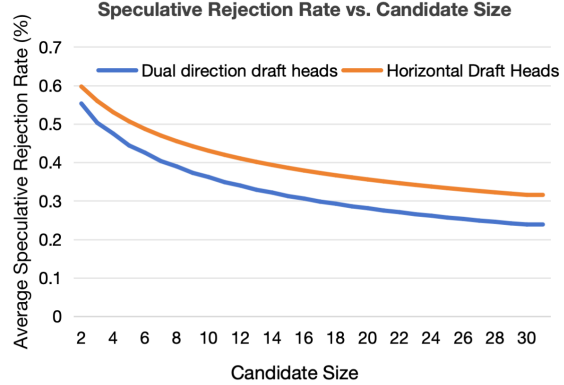


Figure 6: Comparison of dual-direction and horizontal draft heads across varying numbers of candidates. The dual-direction design yields a lower speculative decoding rejection probability, indicating more efficient utilization of draft tokens.

6 Conclusion

In this work, we introduced Hawk, an acceleration framework for autoregressive image generation that leverages spatially aware speculative decoding. By integrating spatial information into the sampling process, Hawk effectively enlarges the candidate space and achieves substantially higher acceptance rates, leading to faster inference without compromising image quality. Extensive experiments demonstrate that Hawk maintains visual fidelity comparable to the baseline while delivering notable speedups. Future work includes integrating Hawk with more advanced speculative backbones (e.g., Eagle [21] or Hydra [1]) to further enhance efficiency and performance.

7 Acknowledgements

This work was supported by the National Key R&D Program of China (No. 2024YFE0202800), the National Natural Science Foundation of China (NSFC) under Grant No. 62476123, and NSFC under Grant No. 62376118.

References

- [1] Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusingha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. Hydra: Sequentially-dependent draft heads for medusa decoding. *arXiv preprint arXiv:2402.05109*, 2024.
- [2] Aishwarya Bhandare, Vamsi Sripathi, Deepthi Karkada, Vivek V. Menon, Sun Choi, Kushal Datta, and Vikram A. Saletore. Efficient 8-bit quantization of transformer neural machine language translation model. *ArXiv*, abs/1906.00532, 2019.
- [3] F Warren Burton. Speculative computation, parallelism, and functional programming. *IEEE Transactions on Computers*, 100(12):1190–1193, 1985.
- [4] 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.
- [5] Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.
- [6] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. In *European Conference on Computer Vision*, pages 88–105. Springer, 2022.
- [7] Yingying Deng, Fan Tang, Weiming Dong, Chongyang Ma, Xingjia Pan, Lei Wang, and Changsheng Xu. Stytr2: Image style transfer with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11326–11336, 2022.
- [8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.
- [9] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, and Jie Tang. Cogview: mastering text-to-image generation via transformers. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS ’21, Red Hook, NY, USA, 2024. Curran Associates Inc.
- [10] Jiale Fu, Yuchu Jiang, Junkai Chen, Jiaming Fan, Xin Geng, and Xu Yang. Speculative ensemble: Fast large language model ensemble via speculation. *arXiv preprint arXiv:2502.01662*, 2025.
- [11] R. Gray. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1984.
- [12] Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*, 2017.
- [13] Baisong Guo, Xiaoyun Zhang, Haoning Wu, Yu Wang, Ya Zhang, and Yan-Feng Wang. Lar-sr: A local autoregressive model for image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1909–1918, 2022.
- [14] Yefei He, Feng Chen, Yuanyu He, Shaoxuan He, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Zipar: Accelerating auto-regressive image generation through spatial locality, 2024.
- [15] John L Hennessy and David A Patterson. *Computer architecture: a quantitative approach*. Elsevier, 2011.
- [16] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.
- [17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.
- [18] 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, 2025.
- [19] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, ICML’23. JMLR.org, 2023.
- [20] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *CoRR*, abs/2406.11838, 2024.
- [21] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning*, ICML’24. JMLR.org, 2024.
- [22] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

- [23] 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.
- [24] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [25] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.
- [26] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, ASPLOS ’24, page 932–949. ACM, Apr. 2024.
- [27] James M Ortega and Werner C Rheinboldt. *Iterative solution of nonlinear equations in several variables*. SIAM, 2000.
- [28] Yassine Ouali, Céline Hudelot, and Myriam Tami. Autoregressive unsupervised image segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pages 142–158. Springer, 2020.
- [29] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.
- [30] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.
- [31] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR, 18–24 Jul 2021.
- [32] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.
- [33] Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.
- [34] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation, 2024.
- [35] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.
- [36] 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 jacobin decoding, 2025.
- [37] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *arXiv preprint arXiv:2404.02905*, 2024.
- [38] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.
- [39] 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. arXiv preprint arXiv:2302.13971*, 2023.
- [40] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.
- [41] Aaron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International conference on machine learning*, pages 1747–1756. PMLR, 2016.
- [42] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.
- [43] Ziyu Wan, Jingbo Zhang, Dongdong Chen, and Jing Liao. High-fidelity pluralistic image completion with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4692–4701, 2021.

- [44] Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2020.
- [45] Yongliang Wu, Shiji Zhou, Mingzhuo Yang, Lianzhe Wang, Heng Chang, Wenbo Zhu, Xinting Hu, Xiao Zhou, and Xu Yang. Unlearning concepts in diffusion model via concept domain correction and concept preserving gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 8496–8504, 2025.
- [46] Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*, 2024.
- [47] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.
- [48] Jing Xiong, Gongye Liu, Lun Huang, Chengyue Wu, Taiqiang Wu, Yao Mu, Yuan Yao, Hui Shen, Zhongwei Wan, Jinfa Huang, et al. Autoregressive models in vision: A survey. *arXiv preprint arXiv:2411.05902*, 2024.
- [49] 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.
- [50] Ali Hadi Zadeh, Isak Edo, Omar Mohamed Awad, and Andreas Moshovos. Gobo: Quantizing attention-based nlp models for low latency and energy efficient inference. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, page 811–824. IEEE, Oct. 2020.
- [51] Kaiqi Zhang, Jing Zhao, and Rui Chen. Koala: Enhancing speculative decoding for llm via multi-layer draft heads with adversarial learning. *arXiv preprint arXiv:2408.08146*, 2024.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: Yes, our abstract and introduction summarize the main contributions of our work.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: Yes, we added an limitation section at the end of the paper.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: Yes, we have complete proof for each theoretical result.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: Yes, the paper has contain all the details to reproduce the result.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code for this paper will be released upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Yes, we have included the hyperparameters of our training and evaluation details.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Papers on large language models (LLMs) typically do not report error bars in their evaluation results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Yes, we have included the training and inferencing computation resources.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: Yes, we follow the code of Ethics

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This paper presents work whose goal is to advance the field of Machine Learning, especially image generation acceleration. There are many potential societal consequences in our work, none of which we feel must be specifically highlighted here.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper has no safeguards risks

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Yes, we have cite all the original owners of assets.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: The code for this paper will be released upon acceptance.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

A Theoretical Justification

Assume we are given two speculative sampling spaces: one from the vertical draft head and the other from the horizontal draft head. Let a denote the number of speculative tokens sampled from the vertical head, and b denote the number of speculative tokens sampled from the horizontal head. In the simplest case, we set $a = 1$ and $b = 1$, meaning that we draw one candidate token from each head for validation. We adopt a sequential validation strategy in which the token proposed by the vertical head is evaluated first. If it is rejected, we then proceed to validate the token from the horizontal head. Under this setup, we define the following notation: $p(x)$ be the true distribution defined by the base AR model. $q_{\text{vert}}(x)$ and $q_{\text{horiz}}(x)$ be the proposal distributions from the vertical and horizontal draft heads, respectively.

$p'_{\text{vert}}(x)$ is the adjusted base model distribution after vertical head rejection. Calculated by

$$\text{norm}(\max(0, p(x) - q_{\text{vert}}(x))) = \frac{p(x) - \min(q_{\text{vert}}(x), p(x))}{\sum_{x'} [p(x') - \min(q_{\text{vert}}(x'), p(x'))]} \quad (8)$$

$p'_{\text{vert-horiz}}(x)$ is the residual distribution after both vertical and horizontal rejections. Calculated by

$$\text{norm}(\max(0, p'_{\text{vert}}(x) - q_{\text{vert}}(x))) = \frac{p'_{\text{vert}}(x) - \min(q_{\text{vert}}(x), p'_{\text{vert}}(x))}{\sum_{x'} [p'_{\text{vert}}(x') - \min(q_{\text{vert}}(x'), p'_{\text{vert}}(x'))]} \quad (9)$$

α_{vert} is the acceptance probability of the vertical head. Calculated by

$$1 - \sum_{x'} (p(x') - \min(q_{\text{vert}}(x'), p(x'))) \quad (10)$$

$\alpha_{\text{vert-horiz}}$ is the acceptance probability of the horizontal head conditioned on vertical rejection. Calculated by

$$1 - \sum_{x'} (p_{\text{vert}}(x') - \min(q_{\text{vert}}(x'), p'_{\text{vert}}(x'))) \quad (11)$$

Our ultimate goal is to approximate the original model's distribution $p(x)$ by combining samples from two draft heads operating along different spatial dimensions. For a given point x' , our proof proceeds as follows:

$$\begin{aligned} p(x') &= \Pr(\text{vertical accept}, x = x') + \Pr(\text{vertical reject, horizontal accept}, x = x') \\ &\quad + \Pr(\text{vertical reject, horizontal reject}, x = x') \\ &= q_{\text{vertical}}(x') \cdot \min\left(1, \frac{p(x')}{q_{\text{vertical}}(x')}\right) \\ &\quad + (1 - \alpha_{\text{vertical}}) \cdot q_{\text{horizontal}}(x') \cdot \min\left(1, \frac{p'_{\text{vertical}}(x')}{q_{\text{horizontal}}(x')}\right) \\ &\quad + (1 - \alpha_{\text{vertical}})(1 - \alpha_{\text{vertical-horiz}}) \cdot p'_{\text{vertical-horiz}}(x') \\ &= q_{\text{vertical}}(x') \cdot \min\left(1, \frac{p(x')}{q_{\text{vertical}}(x')}\right) \\ &\quad + (1 - \alpha_{\text{vertical}}) \cdot q_{\text{horizontal}}(x') \cdot \min\left(1, \frac{p'_{\text{vertical}}(x')}{q_{\text{horizontal}}(x')}\right) \\ &\quad + (1 - \alpha_{\text{vertical}}) \cdot (p'_{\text{vertical}}(x') - \min(q_{\text{horizontal}}(x'), p'_{\text{vertical}}(x')))) \\ &= q_{\text{vertical}}(x') \cdot \min\left(1, \frac{p(x')}{q_{\text{vertical}}(x')}\right) + (1 - \alpha_{\text{vertical}}) \cdot p'_{\text{vertical}}(x') \\ &= p(x') \end{aligned} \quad (12)$$

Let $\pi = [s_1, s_2, \dots, s_{a+b}]$ denote an ordered sequence of speculative tokens, where each s_i is generated from either a vertical or horizontal draft head. The total number of speculative tokens is $a + b$, with a sampled from horizontal heads and b from vertical heads. Each sample result s_i corresponds to a proposal possibility $q_i(x)$. At each step i , the acceptance probability is defined

as $\alpha_i = 1 - \sum_{x'} (p^{(i)}(x') - \min(q_i(x'), p^{(i)}(x')))$ where $p^{(i)}(x)$ represents the residual target density after the first $i - 1$ proposals have been rejected. It is recursively defined as: $p^{(i)}(x) = \text{norm}(\max(0, p^{(i-1)}(x) - q_{i-1}(x)))$, The sampling probability at a specific point $x = x'$ under the generalized formulation is given by:

$$p(x) = \sum_{i=1}^{a+b} \left(\prod_{j=1}^{i-1} (1 - \alpha_j(x)) \cdot q_i(x) \cdot \alpha_i(x) \right) + \left(\prod_{j=1}^{a+b} (1 - \alpha_j(x)) \right) \cdot p_{\text{resid}}(x) \quad (13)$$

This section provides the theoretical justification needed to address the reviewer’s concern about potential distributional shift. It shows that our dual-direction speculative decoding preserves the original output distribution.

B More Attention Sinking Experiment

We conducted further analysis of attention sinking in image generation. In Figure 7, we reshaped the 1D image attention logits into a 2D spatial representation for visualization. The results reveal that during the generation of an image, significant attention is concentrated on the first token of the image and the tokens spatially adjacent to the current inference point. Additionally, tokens at the end of each row tend to have slightly higher weights, while the rest of the attention is evenly distributed across the entire image. Although image generation still follows a raster order, it inherently encodes spatial information.

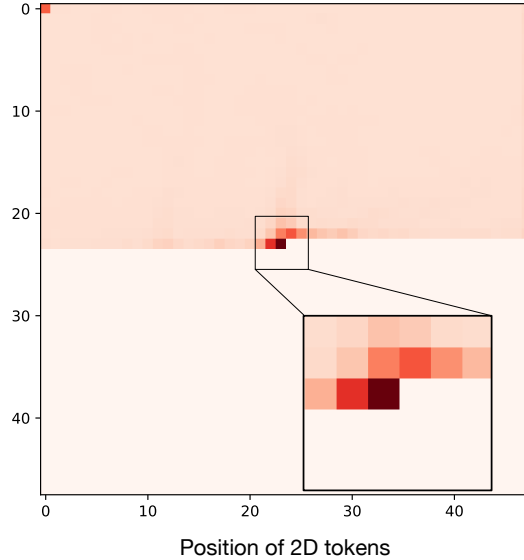


Figure 7: Lumina-mGPT average attention logits converted into a 2D spatial representation.

C Draft Heads Structure and Memory Overhead

For a single Medusa head, we employ a linear layer with both input and output dimensions equal to the model’s hidden size, followed by a SiLU activation function to introduce non-linearity.

The original Medusa strategy introduces only minimal memory overhead (less than 0.5%). Our proposed SSD strategy results in a relatively small increase in the total number of parameters, accounting for less than 1% of the base model (Lumina-mGPT). During inference, the actual memory overhead ratio is even lower due to the use of KV caching, which helps reduce the memory cost of additional components. In the case of FP16, our draft heads occupy approximately 134 MB of memory.



Figure 8: The performance of the base model under different top-k parameters is as follows: from top to bottom, the values of top-k are 2000, 1000, 200, 50, 10, and 1.

D Different Generation Parameters Affect Image Quality

In AR text generation, the top-k parameter is typically set between 5 and 50, representing the maximum number of elements considered during sampling at each step. However, in AR image generation, the choice of top-k has a significantly different impact. Larger top-k values generally result in more detailed images and higher image quality. However, as top-k increases, speculative sampling becomes more challenging: fewer speculative predictions pass the model’s validation, leading to slower speculative decoding speeds. Figure 8 compares how different top-k values affect the image quality of the baseline model.

Temperature plays a crucial role in autoregressive image generation, as the quality of the generated images varies significantly with changes in temperature. In Figure 9, we tested the performance of the base model under different temperature settings. Due to the strong impact of temperature on image quality, certain algorithms, such as jacobi decoding [27] (which is only applicable in greedy decoding), are not suitable for accelerating autoregressive image generation.

E Limitation and Future works

In our current implementation, we employ Medusa as the baseline for speculative decoding. However, according to prior evaluations [46], Medusa is not among the most efficient or state-of-the-art algorithms for text speculative decoding. Consequently, our reported acceleration results may underestimate the potential performance achievable with more advanced speculative decoding methods. In future work, we plan to integrate stronger baseline algorithms to further improve inference efficiency and validate the generality of our proposed approach.

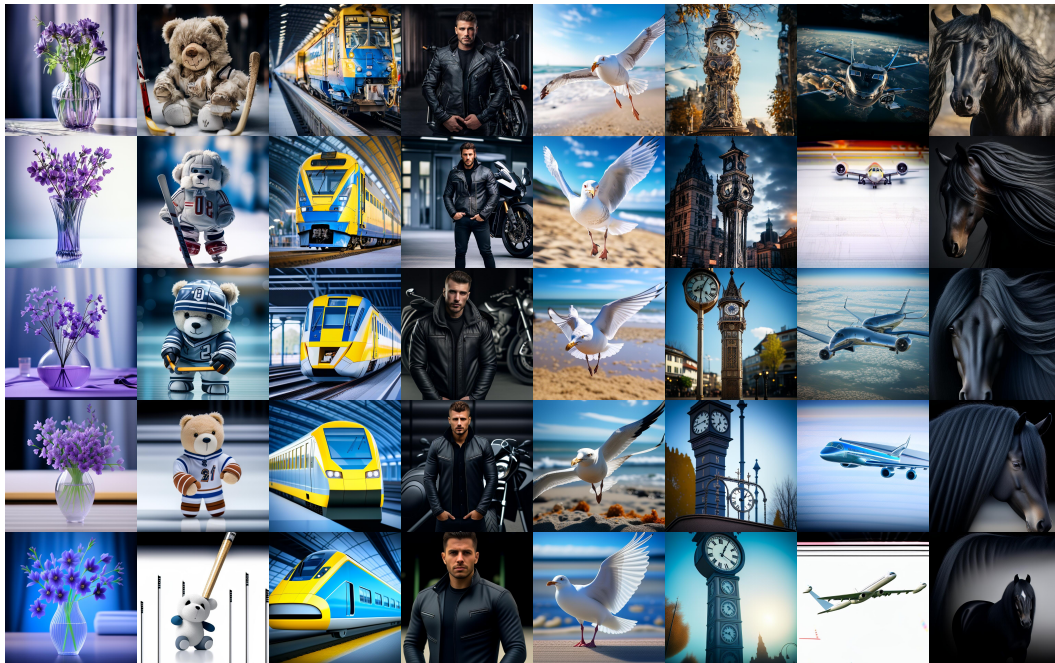


Figure 9: The performance of the base model under different temperature parameters is as follows: from top to bottom, the values of temperature are 1, 0.8, 0.6, 0.4, 0.2.