

Inference-Time Scaling for Diffusion Models beyond Scaling Denoising Steps

Nanye Ma^{*, †, 1}, Shangyuan Tong^{*, ‡, 2}, Haolin Jia³, Hexiang Hu³, Yu-Chuan Su³, Mingda Zhang³, Xuan Yang³, Yandong Li³, Tommi Jaakkola², Xuhui Jia³ and Saining Xie^{1,3}

^{*}Equal contribution, ¹NYU, ²MIT, ³Google, [†]Work done during an internship at Google

Generative models have made significant impacts across various domains, largely due to their ability to scale during training by increasing data, computational resources, and model size, a phenomenon characterized by the scaling laws. Recent research has begun to explore inference-time scaling behavior in Large Language Models (LLMs), revealing how performance can further improve with additional computation during inference. Unlike LLMs, diffusion models inherently possess the flexibility to adjust inference-time computation via the number of denoising steps, although the performance gains typically flatten after a few dozen. In this work, we explore the inference-time scaling behavior of diffusion models *beyond* increasing denoising steps and investigate how the generation performance can further improve with increased computation. Specifically, we consider a search problem aimed at identifying better noises for the diffusion sampling process. We structure the design space along two axes: the verifiers used to provide feedback, and the algorithms used to find better noise candidates. Through extensive experiments on class-conditioned and text-conditioned image generation benchmarks, our findings reveal that increasing inference-time compute leads to substantial improvements in the quality of samples generated by diffusion models, and with the complicated nature of images, combinations of the components in the framework can be specifically chosen to conform with different application scenario.

1. Introduction

Generative models have transformed various fields, including language [1, 78, 80], vision [60, 61], and biology [86], by learning to sample from the underlying data distributions. A key factor in their success is their ability to scale up during training by increasing data volumes, computational resources, and model sizes. This training-time scaling behavior, often described as *Scaling Laws* [27, 31], predicts how performance improves as the models grow larger, consume more data, and are trained for longer time, guiding the development of increasingly capable generative models.

Recently, in Large Language Models (LLMs), the study on scaling has expanded to inference-time [7, 69, 90]. By allocating more compute during inference, often through sophisticated search processes, these works show that LLMs can produce higher-quality and more contextually appropriate responses [20, 74, 85, 87, 95]. Inference-time scaling opens new avenues for improving model performance when additional resources become available after training.

Diffusion models [26, 70, 72], trained to remove noises from data, are a class of generative models that dominates the continuous data domains such as images [15], audio [67], and videos [56]. To generate a single sample, their generation process usually starts from pure noise and requires multiple forward passes of trained models to denoise and obtain clean data. These forward passes are thus dubbed *denoising steps*. Since the number of denoising steps can be adjusted to trade sample quality for computational cost, the generation process of diffusion models naturally provides flexibility in allocating inference-time computation budget. Under the context of generative models, such computation budget is also commonly

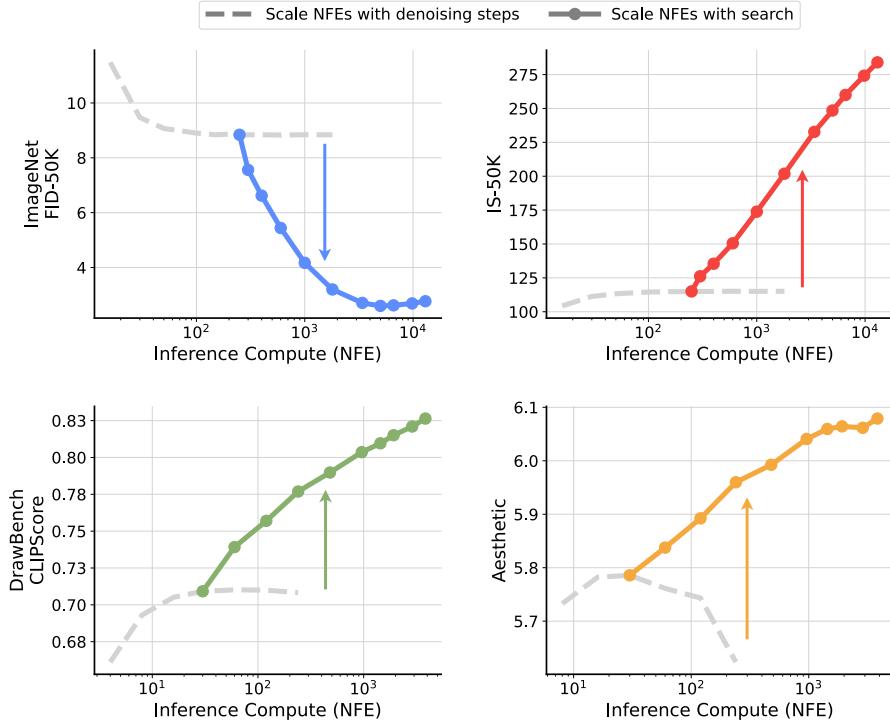


Figure 1 | Inference scaling beyond increasing denoising steps. We demonstrate the performance with respect to FID \downarrow , IS \uparrow on ImageNet, and CLIPScore \uparrow , Aesthetic Score \uparrow on DrawBench. Our search framework exhibits substantial improvements in all settings over purely scaling NFEs with increasing denoising steps.

measured by the *number of function evaluations* (NFE), to ensure a reasonable comparison with other families of models that use iterative sampling processes but without denoising capabilities [79, 96],

Empirical observations [32, 71, 72] have indicated that by investing compute into denoising steps alone, performance gains tend to plateau after a certain NFEs, limiting the benefits of scaling up computation during inference. Therefore, previous work on diffusion models has long focused on maintaining high performance while making NFEs as small as possible for efficiency during inference time [64, 73]. We, on the other hand, are interested in the opposite frontier.

Compared to LLMs, diffusion models work with explicit randomness that comes from the noises injected either as initial samples or during the sampling process [72, 93]. It has been shown that these noises are not created equal [2, 57], i.e., some lead to better generations than others. This observation extends an additional dimension to scale NFEs other than increasing denoising steps - searching for better noises in sampling. Rather than solely allocating NFEs for denoising steps, which often leads to a quick performance plateau, this work investigates methods to effectively utilize compute during inference through search, thereby improving the performance and scalability of diffusion models at inference time (Figure 1). We primarily consider two design axes in our search framework: *verifiers* used to provide feedback in search, and *algorithms* used to find better noise candidates, following terminologies used in LLMs [69].

For *verifiers*, we consider the three different settings, which are meant to simulate three different use cases: (1) where we have privileged information about how the final evaluation is carried out; (2) where we have conditioning information for guiding the generation; (3) where we have no extra information available. For *algorithms*, we consider (1) Random Search, which simply selects the best from a fixed

set of candidates; (2) Zero-Order Search, which leverages verifiers feedback to iteratively refine noise candidates; (3) Search over Paths, which leverages verifiers feedback to iteratively refine diffusion sampling trajectories.

We first walk over these design choices in the relatively simple setting of class-conditioned generation on ImageNet and demonstrate their effectiveness, providing an instantiation of our framework. Then we carry these design choices over to the larger-scale text-conditioned generation setting and evaluate our proposed framework. Due to the complex nature of images and the rich information text conditionings contain, more holistic evaluations of generation quality are required [42]. We therefore employ multiple verifiers for scaling inference-time compute in search. This also enables us to probe into the “biases” each verifier possesses, and how well they are aligned with the generation tasks. To alleviate overfitting to a single verifier, we also experiment with an ensemble of verifiers and showcase its good generalizability across different benchmarks.

Our contributions are summarized as follows:

- We propose a fundamental framework for inference-time scaling of diffusion models. We show that scaling NFEs through search can lead to substantial improvement across generation tasks and model sizes beyond increasing denoising steps. In addition, we conduct a comprehensive empirical analysis of how inference-time compute budgets affect scaling performance.
- We identify two key design axes in the proposed search framework: verifiers, which provide feedback, and algorithms, which find better noise candidates. We examine how different verifier-algorithm combinations perform across various tasks, and our findings indicate that no single configuration is universally optimal; each task instead necessitates a distinct search setup to achieve the best scaling performance.
- We conduct extensive analysis on the alignment between verifiers and different generation tasks. Our results shed light on the biases embedded inside different verifiers and the necessity for a specifically designed verifier in each distinct vision generation task.

2. Background and Motivation

Diffusion models. Diffusion models [26, 70, 72] and more generally flow-based models [3, 46, 47] are a family of generative models that learn to reverse a reference “noising” process. We follow the notations presented in EDM [32], and let the data distribution we want to model be $p_{\text{data}}(\mathbf{x})$ with standard deviation σ_{data} . We consider a reference process that injects different levels of i.i.d. Gaussian noise to the clean data, specified by its standard deviation σ , and we denote these mollified distributions as $p(\mathbf{x}; \sigma)$. The terminal noise level $\sigma_{\max} \gg \sigma_{\text{data}}$ ¹ lets this reference process destroy practically all information of the data $p(\mathbf{x}; \sigma_{\max}) \approx \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \mathbf{I})$. Generation then starts from a pure noise, and simulates some differential equation to progressively denoise the sample to a clean one. Specifically for diffusion models, the underlying vector field is closely related to the score functions $\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma)$ at different noise levels. Often an ordinary differential equation (ODE) [72] or a stochastic differential equation (SDE) [4] is used during the sampling process:

$$\begin{aligned} \text{ODE} \quad & d\mathbf{x} = -\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t))dt \\ \text{SDE} \quad & d\mathbf{x} = -2\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t))dt + \sqrt{2\dot{\sigma}(t)\sigma(t)}dW_t \end{aligned}$$

where $\sigma(t)$ is the σ schedule w.r.t. time, and W_t is the standard Wiener process. Diffusion models D_θ are trained to approximate the ground truth score functions.

¹Flow-based models effectively have $\sigma_{\max} = \infty$.

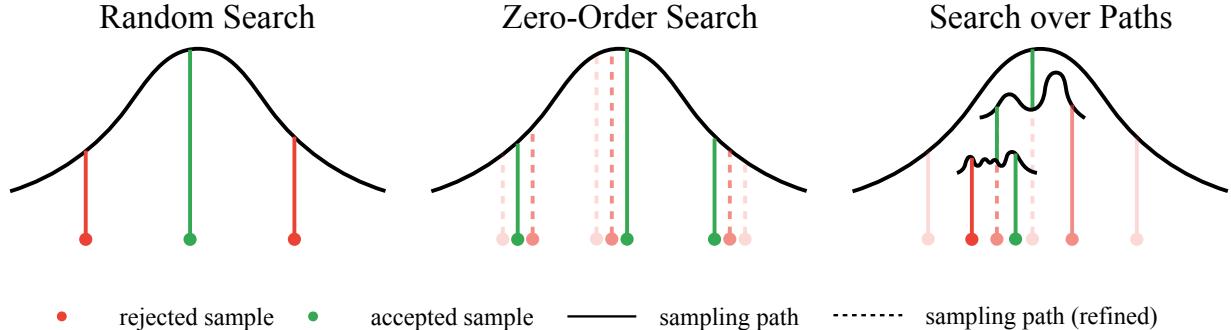


Figure 2 | *Illustration of Search Algorithms.* **Left:** Random Search selects the best sample according to the verifier score and rejects the rest. **Center:** Zero-Order Search samples N candidates in the neighborhood of the pivot noise at each step, and selects the best one according to the verifier to continue the search from. **Right:** Search over Paths sample noises at intermediate sampling steps to add to current samples to expand the sampling trajectories, and select the best one to continue the search.

Innate scaling at inference time. One remarkable property of diffusion models is their innate flexibility to allocate varied compute at inference time for the same task. Because they are trained to approximate the underlying vector field, diffusion models are evaluated multiple times at different noise levels for a single generation. Effectively, the sampling process can be understood as a rolled-out, much larger model, that is stably trained only parts at a time. This mismatch in capacity between training and inference time is one of the key characteristics that separate diffusion models and other generative models like GANs [22] and VAEs [38]. Investing more compute to denoising steps generally leads to better generations, but with diminishing benefits, due to the accumulation of both approximation and discretization errors [93]. Therefore, for diffusion models to scale more at inference time, a new framework needs to be designed.

Randomness from noise. In theory, there is explicit randomness in the sampling of diffusion models: the randomly drawn initial noise, and the optional subsequent noise injected via procedures like SDE [72] and Restart Sampling [93]. Nonetheless, because the model evaluations are inherently deterministic, there is a fixed mapping from these noises to the final samples². It has been shown that some noises are better than others [2, 57], suggesting that it is possible to push the inference time scaling limit by devoting more NFEs to finding the more preferable noises for sampling.

3. How to Scale at Inference Time

With the insights described in Section 2, we now present our framework on inference-time scaling for diffusion models. We formulate the challenge as a search problem over the sampling noises; in particular, how do we know which sampling noises are good, and how do we search for such noises?

On a high-level, there are two design axes we propose to consider:

- **Verifiers** are used to evaluate the goodness of candidates (Section 3.1). These typically are some pre-trained models that are capable of providing feedback; concretely, verifiers are functions

$$\mathcal{V} : \mathbb{R}^{H \times W \times C} \times \mathbb{R}^d \rightarrow \mathbb{R} \quad (1)$$

that takes in the generated samples and optionally the corresponding conditions, and outputs a scalar value as the score for each generated sample.

²Technically we also need to fix the same NFEs in denoising steps, but in practice this requirement is often quite loose (see Section 5 for detail). 4

-
- **Algorithms** are used to find better candidates based on the verifiers scores (Section 3.2). Formally defined, algorithms are functions

$$f : \mathcal{V} \times D_\theta \times \{\mathbb{R}^{H \times W \times C} \times \mathbb{R}^d\}^N \rightarrow \mathbb{R}^{H \times W \times C} \quad (2)$$

that takes in a verifier \mathcal{V} , a pre-trained Diffusion Model D_θ , and N pairs of generated samples and corresponding conditions, and outputs the best initial noises according to the deterministic mapping between noises and samples. Throughout this search procedure, f typically performs multiple forward passes through D_θ (see Section 3.2). We refer to these additional forward passes as the *search cost*, which we measure in terms of NFEs as well.

To give a concrete instantiation of our framework, we present a design walk-through of class-conditional ImageNet [13] generation task. We take a SiT-XL [50] model pre-trained on ImageNet with resolution of 256×256 and perform sampling with a second-order Heun sampler [32], *i.e.*, no other source of randomness but the initial noise used in sampling. We measure inference compute budget with the total NFEs used with *denoising steps* and *search cost*. The denoising steps is fixed to the optimal setting of 250 NFEs [50], and we primarily investigate the scaling behavior with respect to the NFEs devoted to search. Unless specified otherwise, we use classifier-free guidance (cfg) [25] weight of 1.0, focusing on the simple conditional generation task without guidance.

We start with the simplest search algorithm, where we randomly sample Gaussian noises, generate samples from them with ODE, and select those that correspond to the best verifier score (Figure 2). We denote such algorithm **Random Search**, which is essentially a Best-of-N strategy applied once on all noise candidates. Here, the primary axis for scaling NFEs in search is simply the number of noise candidates to select from. For verifiers, we start with the “best” one, an **Oracle Verifier**, which we assume to have full privileged information about the final evaluation of the selected samples. For ImageNet, since FID [24] and IS [65] are typically used as evaluation metrics, we directly take them as the oracle verifiers.

For IS, we select the samples with the highest classification probability output by a pre-trained InceptionV3 model [75] of the conditioning class. For FID, we use the pre-calculated ImageNet Inception feature statistics (mean and covariance) as references, and we greedily choose the sample that minimizes the divergence against the ground-truth statistics. More details are included in Appendix A.

As shown in Figure 3, the straightforward strategy in Random Search is highly effective across all guidance weights. As the NFEs invested in search increases, both FID and IS enjoy substantial improvements with their corresponding oracle verifiers. However, it is important to point out that in most cases it is impractical to directly employ the oracle verifier, since the specifics of the final evaluation procedures are generally not available. Therefore, this setting and the results are merely proof-of-concept, which serves as a confirmation that it is possible to invest compute into search and scale significantly at inference time, provided that the verifiers are chosen appropriately.

3.1. Search Verifiers

In more realistic setups, verifiers could have access to the conditioning used for generation and some pre-trained models that are not explicitly aligned with the final evaluation procedures. In this scenario, the verifiers would evaluate the candidates based on both the quality of the samples and their alignment with the specified conditioning inputs. We denote such family the **Supervised Verifiers**.

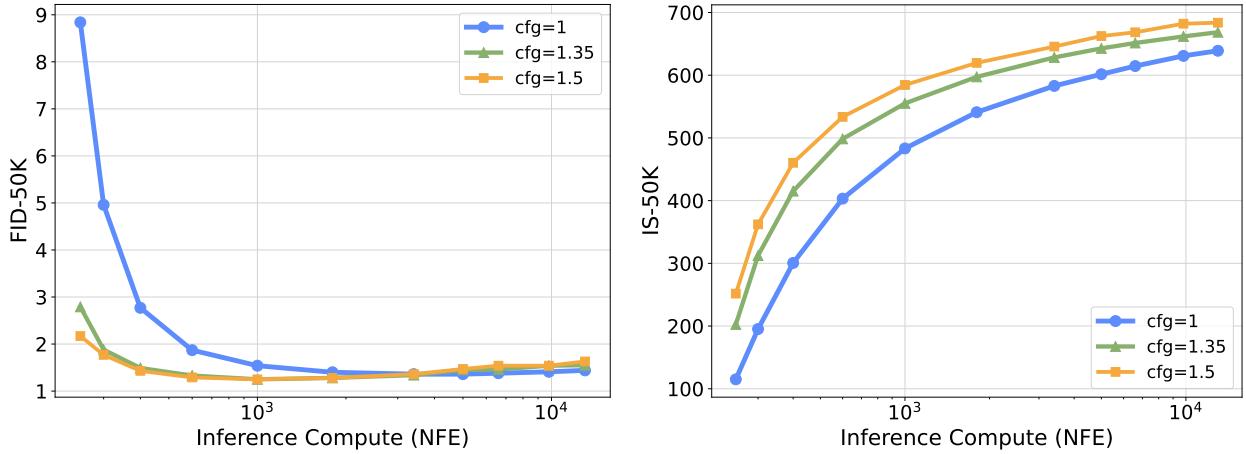


Figure 3 | **Performances of Oracle Verifiers.** Random Search with FID and IS on ImageNet. **Inference Compute** is given by the total NFEs devoted to denoising steps and search; the starting points of all curves in each and the following figures denote only devoting NFEs to denoising steps and 0 NFEs in search.

While scaling NFEs with search demonstrates impressive performance with the oracle verifiers as displayed in Figure 3, the key question is whether its effectiveness can be generalized to supervised verifiers with more accessible pre-trained models designed for various vision tasks. To investigate this, we take two models with good learned representations, CLIP [58] and DINO [53]. Since we only have class labels as the conditioning information on ImageNet, we utilize the classification perspective of the two models. For CLIP, we follow Radford et al. [58] and use the embedding weight generated via prompt engineering³ as a zero-shot classifier. For DINO, we directly take the pre-trained linear classification head. During search, we run samples through the classifiers and select the ones with the highest logits corresponding to the class labels used in generation. We include more settings in Appendix A.

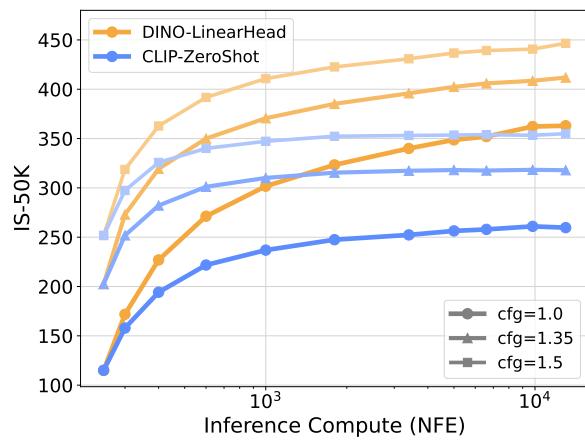


Figure 4 | **Performances of Supervised Verifiers.** Random Search with CLIP and DINO on ImageNet across different classifier-free guidance weights. **CLIP-ZeroShot** refers to using the logits output by the CLIP zero-shot classifier formulated with Prompt Engineering, and **DINO-LinearHead** refers to using the pre-trained linear classifier provided by Oquab et al. [53].

As shown in Figure 4, this strategy also effectively improves the model performance on IS compared to the purely scaling NFEs with increased denoising steps (Figure 1). Nevertheless, we note that, as these classifiers operate point-wise, they are only partially aligned with the goal of FID score (see Appendix B). Specifically, the logits they produce only focus on the quality of a single sample without taking population diversity into consideration, which leads to a significant reduction in sample variance and eventually manifests as mode collapse as the compute increases. The random search algorithm is also to blame due to its unconstrained search space, which accelerates the converging of search towards the bias of verifiers. Such phenomenon is similar to *reward hacking* in reinforcement learning [11, 55], and thus we term it as *Verifier Hacking*.

³See https://github.com/openai/CLIP/blob/main/notebooks/Prompt_Engineering_for_ImageNet.ipynb

Although conditioning information is essential in real-world generation tasks, we discover that it is not necessary for the verifiers to guide the search effectively. As shown in Figure 5, we find that there exists a strong correlation between the logits output by the DINO / CLIP classifiers, and the feature space (extracted by DINO / CLIP, respectively) cosine similarity of the model’s x-prediction at a low noise level ($\sigma = 0.4$) and the final generated clean sample ($\sigma = 0$). Therefore, we proceed to use this similarity score as a surrogate for classification logits, and denote such family **Self-Supervised Verifiers**, given that they do not require extra condition information. We again observe effective scaling behavior in Figure 5. This result is encouraging for use cases where conditioning information is not available or hard to obtain, like the task of medical imaging generation [36]. As these limitations are uncommon in real-world settings, we leave further investigation of Self-Supervised Verifiers to future work.

3.2. Search Algorithms

Our previous explorations have predominantly considered a simple random search setup, which is a one-time best-of-N selection strategy on a randomly chosen fixed set of candidates. Our findings in Section 3.1 indicate that this approach can lead to *verifier hacking*: since the random search operates on the entire Gaussian space, it can quickly overfit to the “bias” of verifiers and lead to failure of our intended goal [21]. This realization motivates us to investigate more nuanced search algorithms that leverage verifiers’ feedback to iteratively refine candidates, only slightly each time, thus mitigating the overfitting risks. Specifically, we consider a **Zero-Order Search** approach:

1. we start with a random Gaussian noise \mathbf{n} as pivot.
2. find N candidates in the pivot’s neighborhood. Formally, the neighborhood is defined as $S_n^\lambda = \{\mathbf{y} : d(\mathbf{y}, \mathbf{n}) = \lambda\}$, where $d(\cdot, \cdot)$ is some distance metric.
3. run candidates through an ODE solver to obtain samples and their corresponding verifier scores.
4. find the best candidates, update it to be the pivot, and repeat steps 1-3.

Much like Zero-Order optimization [19], Zero-Order Search does not involve expensive gradient calculation; instead, it approximates the gradient direction via multiple forward function evaluations inside the neighborhood. As with standard first-order methods [62], we deem the number of iterations (i.e., how many times the algorithm runs through steps 1-3) to be the primary axis for scaling NFEs in search. When N gets larger, the algorithm will locate a more precise local “optimum”, and when λ increases, the algorithm will have a larger stride and thus traversing the noise space more quickly. In practice, we fix the value of λ and investigate the scaling behavior w.r.t N . We abbreviate the algorithm as **ZO-N**.

We note that since many verifiers are differentiable, first-order search with true gradient is technically possible and has seen applications in practice [5, 52]. However, it requires back-propagating through the entirety of the sampling process, which is typically prohibitively costly in terms of both time and

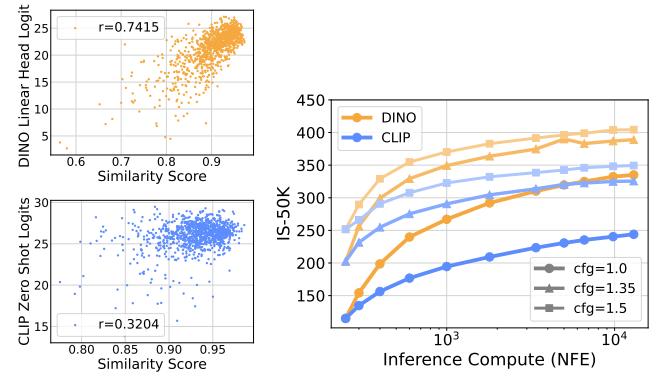


Figure 5 | *Performances of Self-Supervised Verifiers*. Left: correlation between CLIP and DINO feature similarity score and their classification logits; Right: Random Search with CLIP and DINO feature similarity score as verifiers across different classifier-free guidance weight.

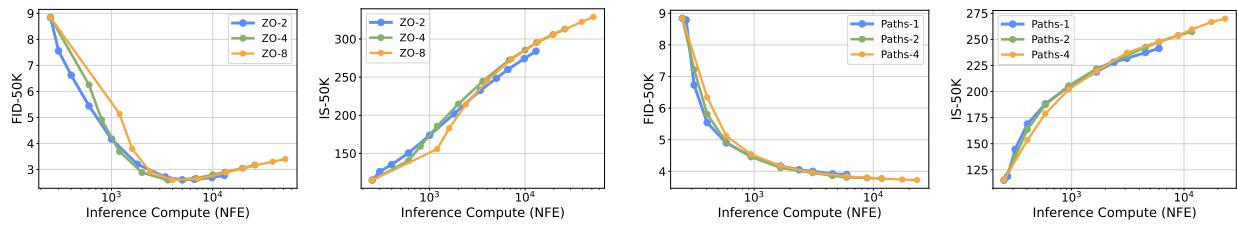


Figure 6 | Performances of Search Algorithms. We fix the verifier to be DINO-LinearHead and investigate the FID and IS of Zero-Order Search and Search over Paths on ImageNet. For each algorithm, we further demonstrate the relationship between N and their performances.

space complexity, especially when scaling large models. In practice, we find that first-order search does not demonstrate a significant advantage over zero-order search on ImageNet despite its higher cost and worse scalability. We include the comparisons in Appendix C.

The iterative nature of diffusion sampling processes yields other possibilities for designing local search algorithms, and it is feasible to search along the sampling trajectories over the noises injected. We propose **Search over Paths** to explore one of such possibilities. Specifically,

1. sample N initial i.i.d. noises and run the ODE solver until some noise level σ . The noisy samples x_σ serve as the search starting point.
2. sample M i.i.d noises for each noisy samples, and simulate the forward noising process from σ to $\sigma + \Delta f$ to produce $\{x_{\sigma+\Delta f}\}$ with size M .
3. run ODE solver on each $x_{\sigma+\Delta f}$ to noise level $\sigma + \Delta f - \Delta b$, and obtain $x_{\sigma+\Delta f - \Delta b}$. Run verifiers on these samples and keep the top N candidates. Repeat steps 2-3 until the ODE solver reaches $\sigma = 0$
4. run the remaining N samples through random search and keep the best one.

To ensure the iteration terminates, we strictly require $\Delta b > \Delta f$. Also, since the verifiers are typically not adapted to noisy input, we perform one additional denoising step in step 3 and use the clean x-prediction to interact with the verifiers. Here, the primary scaling axis is the number of noises M added in step 2, and in practice, we investigate the scaling behavior with different numbers of initial noises N . We thus term the algorithm **Paths- N** . Both algorithms are illustrated in Figure 2, from which we can see that compared to Random Search, both Zero-Order Search and Search over Paths retain very strong locality: the former operates in the neighborhood of the initial noise, and the latter searches in the intermediate steps of the sampling process.

We show the performance of these algorithms in Figure 6. Due to the locality nature of the two algorithms, both of them manage to alleviate the diversity issue of FID to some extent while maintaining a scaling Inception Score. For Zero-Order Search, we note that the effectiveness of increasing N is marginal, and $N = 4$ seems to already be a good estimation of the local optimum. For Search over Paths, we see that different values of N lead to different scaling behavior, with small N being compute efficient in small generation budget, and large N having an advantage when scaling up compute more.

4. Inference-Time Scaling in Text-to-Image

With the instantiation of our search framework in Section 3, we proceed to examine its inference-time scaling capability in larger-scale text-conditioned generation tasks, and study the alignment between verifiers and specific image generation tasks.

Datasets. For a more holistic evaluation of our framework, we use two datasets: (1) DrawBench, introduced in Saharia et al. [63], consists of 200 prompts spanning 11 different categories. It aims to evaluate text-to-image models’ ability to handle complex prompts and generate realistic and high-quality images. During evaluations, we generate one image per prompt. (2) T2I-CompBench [30] is a benchmark designed for evaluating attribute binding, object relationships, and complex compositions. We generate two images per prompt and use the 1800 prompts from the validation set for evaluation.

Models. We use the newly released FLUX.1-dev model [41] as our backbone, which is currently at the frontier of text-to-image generation and representative of the capabilities of many contemporary text-conditioned diffusion models. For detailed sampling settings, see Appendix A.

Verifiers. Due to the inherently sophisticated nature of text-conditioned image generation, a more comprehensive and fine-grained evaluation is required [42]. We therefore expand the choice of supervised verifiers to assess a broader range of aspects in the generated images: *Aesthetic Score Predictor*⁴ [68], *CLIPScore* [23], and *ImageReward* [92]. Relying on a large amount of human-annotated data, these verifiers capture human preferences from different perspectives: *Aesthetic Score Predictor* is trained to predict the human rating of synthesized images’ visual quality; *CLIPScore* aligns visual and text features via 400M human labeled (image, text) pair data; and lastly, *ImageReward* learns to capture more general preferences via carefully curated annotation pipeline, including rating and ranking samples on text-image alignment, aesthetic quality, and harmlessness. Therefore, *ImageReward* has the larger capacity and can capture the evaluative aspects of *Aesthetic Score* and *CLIPScore* to some extent. We include more discussion and results in Section 4.1.

Additionally, we combine these three verifiers to create a fourth verifier, referred to as the *Verifier Ensemble*, to further expand the capacity of verifiers across the evaluative aspects. Since the metrics produced by these verifiers operate on substantially different scales, instead of the absolute scores, we record the relative rankings of metrics across samples, configure the Verifier Ensemble to assess the unweighted average ranking of the three metrics for each sample, and select the sample with the highest ranking.

We find that self-supervised verifiers are less effective in text-to-image settings. We attribute this to two main factors: (1) self-supervised verifiers focus on the visual quality of images but overlook essential textual information, and (2) the large-scale pre-training and extensive fine-tuning might make text-to-image models attain very different sampling dynamics compared to small class-conditioned models trained on ImageNet. We include the performance and more detailed analysis in Appendix D.

Metrics. On DrawBench, we use all verifiers not employed in the search process as primary metrics to provide a more comprehensive evaluation. Considering the usage of Verifier Ensemble, we additionally introduce an LLM as a neutral evaluator for assessing sample qualities.

The extensive pretraining and substantial model capacity of LLMs and Multimodal Large Language Models (MLLM) endow them with exceptional image-text understanding and generalization capabilities, making them highly effective evaluators for assessing the quality of synthesized images across diverse aspects [76]. In fact, many prior works either adopt the VQA approach with LLMs as evaluation models [29, 49, 88], or leverage MLLMs as annotators to obtain feedback on various aspects of the generated images [10, 89]. Inspired by these approaches, we prompt the Gemini-1.5 flash model (via Gemini-1.5-Flash-002 API⁵) to assess synthesized images from five different perspectives: Accuracy to Prompt, Originality, Visual Quality, Internal Consistency, and Emotional Resonance. Each perspective is rated on a scale from 0 to 100, and the averaged overall score is used as the final metric. We denote such evaluator as *LLM Grader*, and include the prompting and evaluation setup in Appendix A.

⁴Though not taking in condition information, the aesthetic predictor is considered to be supervised by the annotated aesthetic scores on LAION. 9
⁵See model details on <https://ai.google.dev/gemini-api/docs/models/gemini#gemini-1.5-flash>

SiT-XL [50]

Increasing NFEs in denoising steps



Increasing NFEs in search



“Lighthouse.” (437)



“Hourglass.” (604)

PixArt- Σ [8]



“A person playing saxophone.”

FLUX-1.dev [41]



“A laptop on top of a teddy bear.”



“Photo of an athlete cat explaining it’s latest scandal at a press conference to journalists.”

Figure 7 | *Visualizations of Scaling Behaviors.* Each row is constructed as follows: **left three:** sampled with increasing NFEs in denoising steps; **right four:** sampled with increasing NFEs in search. First two rows are sampled from SiT-XL [50] with DINO-LinearHead, third row is sampled from PixArt- Σ [8] with Verifier Ensemble, and last two rows are sampled from FLUX-1.dev [41] with Verifier Ensemble.

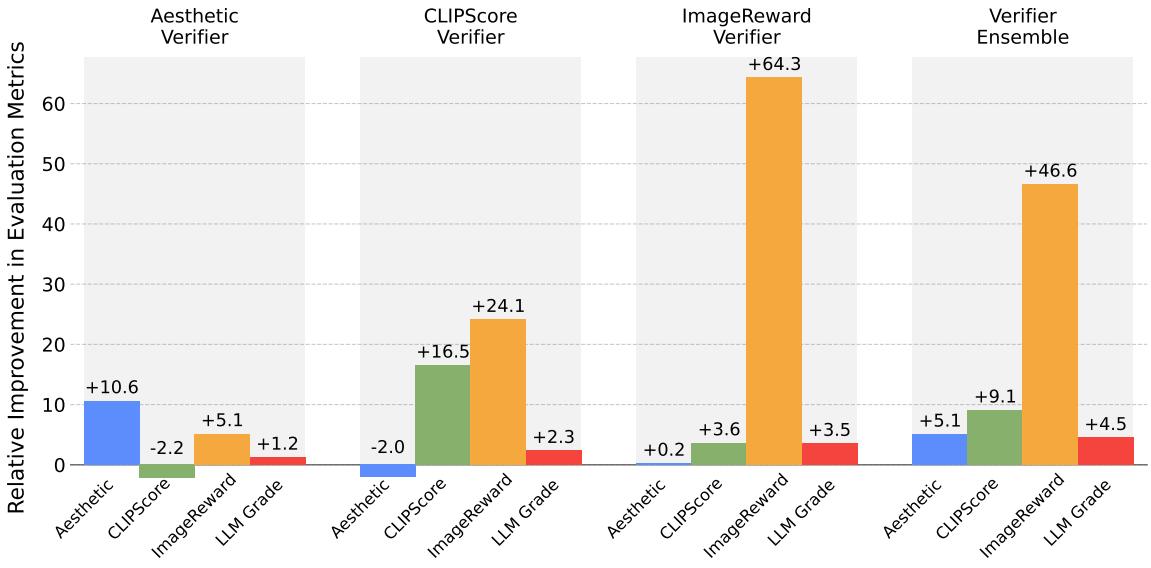


Figure 8 | *Performances of Search with FLUX.1-dev at inference-time.* We fix the search budget to be 3840 NFEs with random search, and demonstrate the relative performance gain (%) with generation without any search budget.

Lastly, on T2I-CompBench, we use the evaluation pipeline provided by Huang et al. [30] to assess the performance of our framework in compositional generation tasks. The pipeline utilizes the BLIP-VQA model [43] for attribute binding evaluation, UniDet model [98] for spatial relationship evaluation, and finally weighted average of BLIP, UniDet, and CLIP Score for evaluating complex compositions.

4.1. Analysis Results: Verifier-Task Alignment

We now present our results comparing combinations of verifiers and algorithms on different datasets.

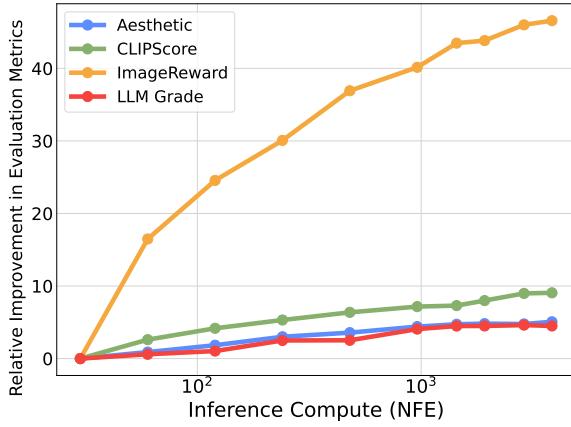


Figure 9 | *Scalability of search with FLUX.1-dev on DrawBench.* We use random search with Verifier Ensemble to obtain the results, and demonstrate the relative performance gain (%) with generation without any search budget. Similar scaling behavior to ImageNet setting is observed across different metrics.

demonstrates our claim that search setups can be specifically chosen to conform to different application

DrawBench. DrawBench is a highly general-purpose dataset containing text prompts from diverse categories. We argue that evaluating generation tasks on such dataset requires assessing a wide range of aspects rather than focusing narrowly on specific criteria (e.g., aesthetic, text alignment). Given its comprehensive pretraining knowledge and the diverse evaluative aspects we established, the LLM Grader serves as an effective surrogate for human preferences on this benchmark. By leveraging it, we can assess how well a verifier aligns with the broad requirements of generation tasks on DrawBench.

As illustrated in Figure 8, and as indicated by the LLM Grader, searching with all verifiers generally improves sample quality, while specific improvement behaviors vary across different setups. This

scenarios. For instance, ImageReward and Verifier Ensemble, which possess more nuanced evaluative aspects and closely align with human preferences, consistently improve scores across all evaluation metrics, making them suitable for the generalized generation tasks on DrawBench. In contrast, Aesthetic and CLIP Verifiers are less desirable for tasks requiring satisfying performances across multiple evaluative aspects, due to the effect of *Verifier hacking*.

From the left two columns in Figure 8, we see that searching with Aesthetic and CLIP Verifier overfit to their inherent biases, negatively impacting each other. We conjecture that both verifiers suffer from major misalignment in evaluation: the Aesthetic Score focuses solely on visual quality, often favoring highly stylized images deviating from their text prompt, whereas CLIP prioritizes visual-text alignment at the expense of visual quality [11, 83, 94]. As a result, exploiting the biases of one verifier (e.g. Aesthetic Score) during search degrades the evaluation metrics assessed by the other verifier (e.g. CLIP). This aligns with observations by Clark et al. [11], who noted that extensive fine-tuning with Aesthetic or CLIP rewards can cause the model distribution to collapse to a single high-reward mode.

However, we point out that our search method does not modify the model’s learned score function and will preserve its pretrained behavior on individual samples. Consequently, unlike the complete collapsing in single-sample quality observed by Clark et al. [11], the samples selected by our search method remain within the learned data distribution, only with their mode shifted towards one of the verifiers (say Aesthetic) and slightly away (only $\sim -2\%$ in performance) from the other (say CLIP). This is further supported by the evaluation results from the LLM Grader, that searching for Aesthetic or CLIP Score can still improve the overall preference scores, despite trade-offs between the two. Importantly, since searching with Aesthetic and CLIP Score does not lead to a total collapse in sample quality and leverages their unique strengths in aesthetic quality and text faithfulness, they can be well-suited for tasks that require a focus on specific attributes such as visual appeal or textual accuracy, rather than maintaining general-purpose performance.

Lastly, from Figure 9, we observe similar scaling behavior of evaluation metrics with respect to increasing search budget, similar to the ImageNet settings.

T2I-CompBench. Unlike DrawBench, the evaluation pipeline on T2I-CompBench primarily emphasize correctness in relation to the text prompt [30], such as accurately generating colors, object relationships, and overall compositions, without prioritizing pure visual quality. These different goals effectively call for a different search setup, and the results from Table 1 support this claim. We have observed that searching with Aesthetic Scores leads to minimal improvements and even degradation in metrics. Although all three remaining verifiers account for text faithfulness to some extent, they demonstrate varying degrees of improvement. Notably, ImageReward outperforms Verifier Ensemble across all evaluation categories, while CLIP provides only marginal gains. This can be attributed to the fact that CLIP lacks the nuanced evaluative aspects aligned with human preferences, and Verifier Ensemble includes Aesthetic Score, which negatively impacts evaluation performance on this task.

These contrasting behaviors of verifiers on DrawBench and T2I-CompBench highlight that the effectiveness

Verifier	Color	Shape	Texture	Spatial	Numeracy	Complex
-	0.7692	0.5187	0.6287	0.2429	0.6167	0.3600
Aesthetic	0.7618	0.5119	0.5826	0.2593	0.6159	0.3472
CLIP	0.8009	0.5722	0.7005	0.2988	0.6457	0.3704
ImageReward	0.8303	0.6274	0.7364	0.3151	0.6789	0.3810
Ensemble	0.8204	0.5959	0.7197	0.3043	0.6623	0.3754

Table 1 | Performance of search with FLUX.1-dev on T2I-CompBench. We use random search with Verifier Ensemble to obtain the samples; for evaluation, we use the pipeline provided in T2I-CompBench. The first row denotes the performance without search where we fix the denoising budget to be 30 NFEs, and for the rest, we fix the search budget to be 1920 NFEs.

Verifier	Aesthetic	CLIPScore	ImageReward	LLM Grader
-	5.79	0.71	0.97	84.29
Aesthetic + Random	6.38	0.69	0.99	86.04
	+ ZO-2	6.33	0.69	85.90
	+ Paths-2	6.31	0.70	85.86
CLIPScore + Random	5.68	0.82	1.22	86.15
	+ ZO-2	5.72	0.81	85.48
	+ Paths-2	5.71	0.81	85.45
ImageReward + Random	5.81	0.74	1.58	87.09
	+ ZO-2	5.79	0.73	86.22
	+ Paths-2	5.76	0.74	86.33
Ensemble + Random	6.06	0.77	1.41	88.18
	+ ZO-2	5.99	0.77	87.25
	+ Paths-2	6.02	0.76	86.84

Table 2 | *Performance of search algorithms with different verifiers on DrawBench.* The results are obtained from FLUX.1-dev evaluated on DrawBench. The first row denotes the performance without search where we fix denoising budget to be 30 NFEs, and for the rest we fix search budget to be 2880 NFEs.

of a verifier depends on how well its criteria align with the specific requirements of the task, with certain verifiers being better suited for particular tasks than others. This inspires the design of more task-specific verifiers, which we leave as future works.

Algorithms. In Table 2 we demonstrate the performance of the three presented search algorithms on DrawBench. For Zero-Order Search, we set a fixed number for neighbors, $N = 2$. For Search over Paths, we set the number of initial noises, $N = 2$, as well. More detailed settings are included in Appendix A.

We see that all three methods can effectively improve the sampling quality, with random search outperforming the other two methods in some aspects. Again we credit this behavior to the locality nature of Zero-Order Search and Search over Paths (Figure 2). Since all verifiers and metrics we present are evaluated on a per-sample basis, random search will drastically accelerate the convergence to the bias of verifiers, whereas the other two algorithms need to perform refinement on the suboptimal candidates.

4.2. Search is Compatible with Finetuning

Both search and finetuning methods [11, 18] aim to align the final samples with explicit reward models or human preferences. While the former shifts the sample modes toward the bias of specific verifiers, the latter directly modifies the model’s distribution to align with the rewards. This raises a natural question: can we still shift the sample modes according to verifiers after the model distribution has been modified?

Model	Aesthetic	CLIP	PickScore
SDXL	5.56	0.73	22.39
+ DPO	5.59	0.74	22.54
+ DPO & Search	5.66	0.76	23.54

Table 3 | *Performance of Search with DPO-finetuned SDXL.* We use random search with Verifier Ensemble on DrawBench to obtain the result. We set the denoising budget to 40 NFEs, and search budget to 1280 NFEs.

Among all finetuning methods explored, Diffusion-DPO [83], as a more efficient and simpler alternative to RLHF [54] methods, has been widely adopted in aligning large-scale text-to-image models. To answer the question, we take the DPO fine-tuned Stable Diffusion XL model in [83] and conduct search on the DrawBench dataset. Since the model is finetuned on the dataset Pick-a-Pic [39], we replace ImageReward with the PickScore evaluator. The results are included in Table 3.

We see that search method can generalize to different models and can improve the performance of an already aligned model. We note this will be a useful tool to mitigate the cases where finetuned models disagree with reward models [28], and to improve their generalizability to other metrics [11].

5. Axes of Inference Compute Investment

Due to the iterative sampling nature of diffusion models, there are multiple dimensions in which we can scale NFEs with search. We present them below and investigate their impact on performances.

Number of search iterations. Intuitively, increasing the number of search iterations allows the selected noises to approach the optimal set with respect to verifiers and can thus substantially improve performance. We have observed such behavior in all of our previous experiments.

Compute per search iteration. Within each search iteration, we could adjust the number of denoising steps the model takes. For simplicity, we denote this $NFEs/iter$. Whereas the model performance plateaus quickly when only increasing denoising steps (Figure 1), we observe that during the search process, adjusting $NFEs/iter$ can reveal distinct compute-optimal regions, as demonstrated in Figure 10. Smaller $NFEs/iter$ during search enables efficient convergence, though with a lower final performance. Conversely, larger $NFEs/iter$ result in slower convergence but yield improved performance. Additionally, a diminishing return effect is demonstrated: when $NFEs/iter \geq 50$, further increases in $NFEs/iter$ yield minimal gains despite the additional computational investment. Inspired by this observation, we set the $NFEs/iter$ for each search iteration to 50 for previous experiments on ImageNet for efficient compute allocation. For experiments in text-to-image setting, since FLUX-1.dev is able to generate high-quality samples with relatively small number of denoising steps, we fix the $NFEs/iter$ to 30, aligning with the final generation.

Compute of final generation. Despite the freedom in adjusting the denoising steps for the final generation, we always use the optimal setting for the best final sample quality. In ImageNet, we fix 250 NFEs for the denoising budget, and in text-to-image setting a 30-step sampler is used, as scaling up further will quickly come to a performance plateau.

5.1. Effectiveness of Investing Compute

We explore the effectiveness of scaling inference-time compute for smaller diffusion models and highlight its efficiency relative to the performance of their larger counterparts without search. For ImageNet tasks, we utilize SiT-B and SiT-L, and for text-to-image tasks, we use the smaller transformer-based model PixArt- Σ [8] besides FLUX-1.dev. We report various metrics evaluated on these models under their optimal setups: Zero-Order Search with DINO logits for FID on ImageNet, Random Search with DINO logits for IS on ImageNet, and Random Search with the Verifier Ensemble for text-to-image evaluation on DrawBench. Since models of different sizes incur significantly different costs per forward pass, we use estimated GFLOPs to measure their computational cost instead of NFEs.

From Figure 11, we observe that scaling inference-time compute for small models on ImageNet can

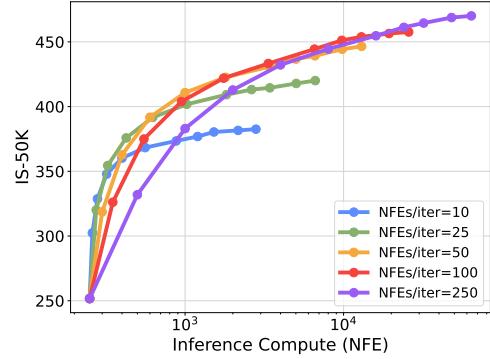


Figure 10 | *Performance of scaling compute for single search iteration.* We use the SiT-XL model, fix the denoising budget to 250 NFE, and demonstrate the performance differences with respect to the NFEs devoted to a single search iteration.

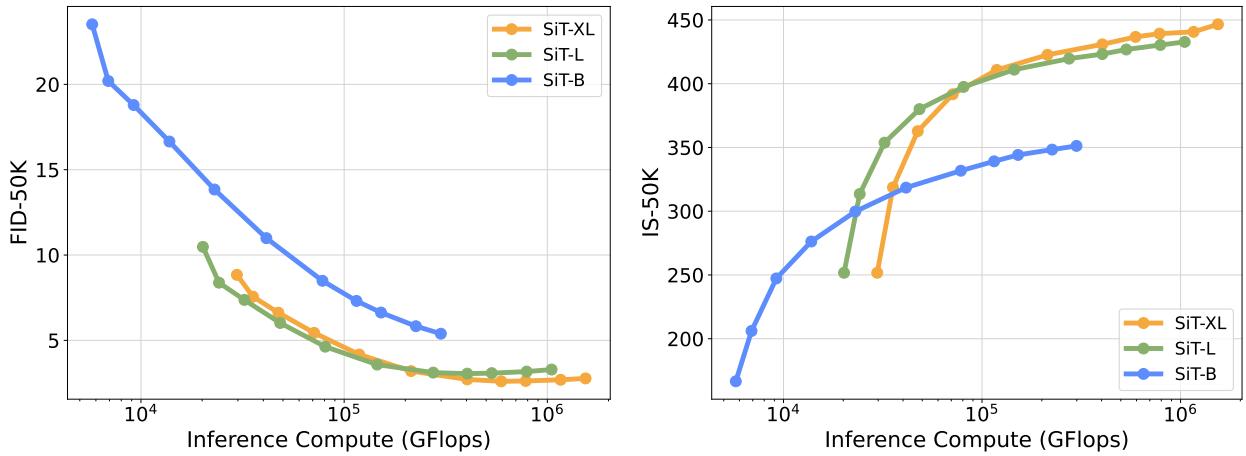


Figure 11 | *Performance of our search methods across different model sizes (SiT-{B,L,XL}) on ImageNet.* We use the best set up for FID and IS separately. Left: ZO-4 with DINO-LinearHead.; Right: Random Search with DINO-LinearHead.

be highly effective. With a fixed inference compute budget, performing search on small models can outperform larger models without search. For instance, SiT-L demonstrates an advantage over SiT-XL in regions with limited inference compute. However, comparing SiT-B with the other two models reveals that this effectiveness depends on the relatively strong baseline performance of the small models. When a small model’s baseline performance lags significantly, the benefits of scaling are limited, resulting in suboptimal outcomes.

These observations extend to the text-conditioned setting, as demonstrated in Table 4. With just one-tenth of the compute, PixArt- Σ outperforms FLUX-1.dev without search, and with roughly double the compute, PixArt- Σ surpasses FLUX-1.dev without search by a significant margin. These results have important practical implications: the substantial compute resources invested in training can be offset by a fraction of that compute during generation, enabling access to higher-quality samples more efficiently.

6. Related Work

Scaling test-time compute. Scaling test-time compute is proven to be highly effective on pre-trained LLMs. This presents a completely different axis in LLM’s scaling behaviors and inspires many investigations. Recent studies in test-time scaling of LLMs mainly focus on three aspects: (1) better search/planning algorithms [20, 74, 87, 91]; (2) better verifiers [12, 44, 45, 84]; and (3) scaling law of test-time compute [7, 69, 90]. These works highlight the importance of test-time compute and methods for effectively allocating these compute under a certain budget, orienting the community towards building agents with the ability to reason and self-correct. Inspired by these works, we study the scaling behavior of diffusion models at inference-time, introduce a general search framework over injected noises during sampling, and demonstrate its effectiveness across different benchmarks, aiming to motivate more

Model	Compute Ratio	Aesthetic	CLIP	ImageReward	LLM Grader
FLUX	1	5.79	0.71	0.97	84.29
PixArt- Σ	~0.06	5.94	0.68	0.70	84.67
	~0.09	6.03	0.71	0.97	85.62
	~2.59	6.20	0.73	1.15	86.95

Table 4 | *Comparison between PixArt- Σ when search with Verifier Ensemble and FLUX without search.* We use the total compute consumed by FLUX to generate one sample as the standard unit and scale the compute used by PixArt- Σ accordingly. These total compute estimates are based on our best approximation and may not be entirely precise.

explorations of inference-time scaling in the diffusion model community.

Fine-tuning diffusion models. To align diffusion models with human preferences, multiple fine-tuning methods have been proposed. Fan and Lee [17] interpret the denoising process as a multi-step decision-making task and use policy gradient algorithms to fine-tune diffusion samplers. Black et al. [6], Fan et al. [18] formulate the fine-tuning task as an RL problem, and using policy gradient to maximize the feedback-trained reward. Clark et al. [11], Xu et al. [92] further simplifies this task by directly back-propagating the reward function gradient through the full sampling procedure. Wallace et al. [83] reformulate *Direct Preference Optimization* [59] to derive a differentiable preference objective that accounts for a diffusion model notion of likelihood, and Yang et al. [94] discard the explicit reward model and directly fine-tune the model on human preference data. Lastly, Domingo-Enrich et al. [14] casts fine-tuning problem as stochastic optimal control to better align with the tilted distribution of base and reward models. These studies represent substantial advancements in enforcing alignment in diffusion models, ensuring they better adhere to human preferences, ethical considerations, and controlled behaviors.

Sample selection and optimization in diffusion models. The large variation in diffusion’s sampling qualities leads to the natural question of how to find good samples during test-time. To address this, several works focus on sample selection guided by some pre-defined metrics using the Random Search algorithm. Karthik et al. [34] and Liu et al. [48] use pre-trained VQA and human preference models to guide the selection, and Liu et al. [48] further update the proposal distribution during selection to better align with the ground truth distribution. Similarly, Na et al. [51] performs rejection sampling on the updated proposal distribution during intermediate diffusion denoising step. On the other hand, Tang et al. [77] and Samuel et al. [66] use a small set of ground truth images as reference and use the similarity between reference and generated images as a guide for selection. Yet, these works primarily focus on addressing challenges using very specific verifier and algorithm, while largely overlooking a comprehensive investigation into the biases inherent in different verifiers, the interplay of multiple verifiers and search methods on different tasks, and the relationship between inference-time compute budget and scaling performance. Some other works [5, 16, 35, 52, 82] utilize the gradient of a pre-trained reward model to directly optimize for a better sample. We note, again, that these works focus on relatively small-scaled tasks (in-painting, editing, super-resolution), and the costs of these methods are prohibitive due to the need to back-propagate through the diffusion sampling process.

Recently, several studies[2, 99] have proposed approximating the distribution of “good” noises using neural networks. These approaches first identify preferable noises x'_T by transforming random noises $x_T \sim \mathcal{N}(0, I)$ through guided DDIM inversion. Subsequently, they train a lightweight predictor on the set of (x_T, x'_T) pairs for sampling preferable noises at inference-time. Although these methods shift computational costs from test time to a one-time training phase, they require additional dataset curation and parameter tuning, and can have unsatisfying performance in some application scenarios.

7. Conclusion

In this work, we present a framework for inference-time scaling in diffusion models, demonstrating that scaling compute through search could significantly improve performances across various model sizes and generation tasks, and different inference-time compute budget can lead to varied scaling behavior. Identifying verifiers and algorithms as two crucial design axes in our search framework, we show that optimal configurations vary by task, with no universal solution. Additionally, our investigation into the alignment between different verifiers and generation tasks uncovers their inherent biases, highlighting the need for more carefully designed verifiers to align with specific vision generation tasks.

Acknowledgements

We thank Ziyu Wan, Jack Lu, Boyang Zheng, Oliver Wang, Jason Baldridge and Sayak Paul for their insightful discussions.

References

- [1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [2] D. Ahn, J. Kang, S. Lee, J. Min, M. Kim, W. Jang, H. Cho, S. Paul, S. Kim, E. Cha, K. H. Jin, and S. Kim. A noise is worth diffusion guidance. *arXiv preprint arXiv:2412.03895*, 2024.
- [3] M. S. Albergo and E. Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=li7qeBbCR1t>.
- [4] B. D. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.
- [5] H. Ben-Hamu, O. Puny, I. Gat, B. Karrer, U. Singer, and Y. Lipman. D-flow: Differentiating through flows for controlled generation. *arXiv preprint arXiv:2402.14017*, 2024.
- [6] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.
- [7] B. Brown, J. Juravsky, R. Ehrlich, R. Clark, Q. V. Le, C. Ré, and A. Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.
- [8] J. Chen, C. Ge, E. Xie, Y. Wu, L. Yao, X. Ren, Z. Wang, P. Luo, H. Lu, and Z. Li. Pixart-\sigma: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. *arXiv preprint arXiv:2403.04692*, 2024.
- [9] T. Chen, B. Xu, C. Zhang, and C. Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.
- [10] Z. Chen, Y. Du, Z. Wen, Y. Zhou, C. Cui, Z. Weng, H. Tu, C. Wang, Z. Tong, Q. Huang, et al. Mj-bench: Is your multimodal reward model really a good judge for text-to-image generation? *arXiv preprint arXiv:2407.04842*, 2024.
- [11] K. Clark, P. Vicol, K. Swersky, and D. J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=1vmSEVL19f>.
- [12] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

-
- [13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
 - [14] C. Domingo-Enrich, M. Drozdzal, B. Karrer, and R. T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. *arXiv preprint arXiv:2409.08861*, 2024.
 - [15] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.
 - [16] L. Eyring, S. Karthik, K. Roth, A. Dosovitskiy, and Z. Akata. Reno: Enhancing one-step text-to-image models through reward-based noise optimization. *arXiv preprint arXiv:2406.04312*, 2024.
 - [17] Y. Fan and K. Lee. Optimizing DDPM sampling with shortcut fine-tuning. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 9623–9639. PMLR, 2023.
 - [18] Y. Fan, O. Watkins, Y. Du, H. Liu, M. Ryu, C. Boutilier, P. Abbeel, M. Ghavamzadeh, K. Lee, and K. Lee. Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.
 - [19] A. D. Flaxman, A. T. Kalai, and H. B. McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. *arXiv preprint cs/0408007*, 2004.
 - [20] K. Gandhi, D. Lee, G. Grand, M. Liu, W. Cheng, A. Sharma, and N. D. Goodman. Stream of search (sos): Learning to search in language. *arXiv preprint arXiv:2404.03683*, 2024.
 - [21] L. Gao, J. Schulman, and J. Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR, 2023.
 - [22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.
 - [23] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.
 - [24] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.
 - [25] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
 - [26] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
 - [27] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.
 - [28] X. Hu, T. He, and D. Wipf. New desiderata for direct preference optimization. *arXiv preprint arXiv:2407.09072*, 2024.

-
- [29] Y. Hu, B. Liu, J. Kasai, Y. Wang, M. Ostendorf, R. Krishna, and N. A. Smith. Tifa: Accurate and interpretable text-to-image faithfulness evaluation with question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20406–20417, 2023.
- [30] K. Huang, K. Sun, E. Xie, Z. Li, and X. Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36:78723–78747, 2023.
- [31] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [32] T. Karras, M. Aittala, T. Aila, and S. Laine. Elucidating the design space of diffusion-based generative models. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=k7FuTOWM0c7>.
- [33] T. Karras, M. Aittala, J. Lehtinen, J. Hellsten, T. Aila, and S. Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24174–24184, 2024.
- [34] S. Karthik, K. Roth, M. Mancini, and Z. Akata. If at first you don't succeed, try, try again: Faithful diffusion-based text-to-image generation by selection. *arXiv preprint arXiv:2305.13308*, 2023.
- [35] K. Karunratanakul, K. Preechakul, E. Aksan, T. Beeler, S. Suwajanakorn, and S. Tang. Optimizing diffusion noise can serve as universal motion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1334–1345, 2024.
- [36] F. Khader, G. Mueller-Franzes, S. T. Arasteh, T. Han, C. Haarburger, M. Schulze-Hagen, P. Schad, S. Engelhardt, B. Baessler, S. Foersch, et al. Medical diffusion: denoising diffusion probabilistic models for 3d medical image generation. *arXiv preprint arXiv:2211.03364*, 2022.
- [37] K. Kim, J. Jeong, M. An, M. Ghavamzadeh, K. Dvijotham, J. Shin, and K. Lee. Confidence-aware reward optimization for fine-tuning text-to-image models. *arXiv preprint arXiv:2404.01863*, 2024.
- [38] D. P. Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [39] Y. Kirstain, A. Polyak, U. Singer, S. Matiana, J. Penna, and O. Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:36652–36663, 2023.
- [40] T. Kynkänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32, 2019.
- [41] B. F. Labs. Flux.1 [dev]. <https://blackforestlabs.ai/>.
- [42] T. Lee, M. Yasunaga, C. Meng, Y. Mai, J. S. Park, A. Gupta, Y. Zhang, D. Narayanan, H. Teufel, M. Bellagente, et al. Holistic evaluation of text-to-image models. *Advances in Neural Information Processing Systems*, 36, 2024.
- [43] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022.

-
- [44] W. Li and Y. Li. Process reward model with q-value rankings. *arXiv preprint arXiv:2410.11287*, 2024.
- [45] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let’s verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.
- [46] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=PqvMRDCJT9t>.
- [47] X. Liu, C. Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=XVjTT1nw5z>.
- [48] Y. Liu, Y. Zhang, T. Jaakkola, and S. Chang. Correcting diffusion generation through resampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8713–8723, 2024.
- [49] Y. Lu, X. Yang, X. Li, X. E. Wang, and W. Y. Wang. Llmscore: Unveiling the power of large language models in text-to-image synthesis evaluation. *Advances in Neural Information Processing Systems*, 36, 2024.
- [50] N. Ma, M. Goldstein, M. S. Albergo, N. M. Boffi, E. Vanden-Eijnden, and S. Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. *arXiv preprint arXiv:2401.08740*, 2024.
- [51] B. Na, Y. Kim, M. Park, D. Shin, W. Kang, and I.-C. Moon. Diffusion rejection sampling. *arXiv preprint arXiv:2405.17880*, 2024.
- [52] Z. Novack, J. McAuley, T. Berg-Kirkpatrick, and N. J. Bryan. Ditto: Diffusion inference-time t-optimization for music generation. *arXiv preprint arXiv:2401.12179*, 2024.
- [53] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- [54] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [55] A. Pan, K. Bhatia, and J. Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544*, 2022.
- [56] A. Polyak, A. Zohar, A. Brown, A. Tjandra, A. Sinha, A. Lee, A. Vyas, B. Shi, C.-Y. Ma, C.-Y. Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.
- [57] Z. Qi, L. Bai, H. Xiong, et al. Not all noises are created equally: Diffusion noise selection and optimization. *arXiv preprint arXiv:2407.14041*, 2024.
- [58] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

-
- [59] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.
 - [60] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021.
 - [61] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.
 - [62] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
 - [63] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.
 - [64] T. Salimans and J. Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=TIIdIXIpzhoI>.
 - [65] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.
 - [66] D. Samuel, R. Ben-Ari, S. Raviv, N. Darshan, and G. Chechik. Generating images of rare concepts using pre-trained diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4695–4703, 2024.
 - [67] F. Schneider. Archisound: Audio generation with diffusion. *arXiv preprint arXiv:2301.13267*, 2023.
 - [68] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. 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.
 - [69] C. Snell, J. Lee, K. Xu, and A. Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.
 - [70] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.
 - [71] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=St1giarCHLP>.
 - [72] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=PxTIG12RRHS>.

-
- [73] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.
- [74] D. Su, S. Sukhbaatar, M. Rabbat, Y. Tian, and Q. Zheng. Dualformer: Controllable fast and slow thinking by learning with randomized reasoning traces. *arXiv preprint arXiv:2410.09918*, 2024.
- [75] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
- [76] Z. Tan, X. Yang, L. Qin, M. Yang, C. Zhang, and H. Li. Evalalign: Supervised fine-tuning multimodal llms with human-aligned data for evaluating text-to-image models. *arXiv preprint arXiv:2406.16562*, 2024.
- [77] L. Tang, N. Ruiz, Q. Chu, Y. Li, A. Holynski, D. E. Jacobs, B. Hariharan, Y. Pritch, N. Wadhwa, K. Aberman, et al. Realfill: Reference-driven generation for authentic image completion. *ACM Transactions on Graphics (TOG)*, 43(4):1–12, 2024.
- [78] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [79] K. Tian, Y. Jiang, Z. Yuan, B. Peng, and L. Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *Advances in Neural Information Processing Systems*, 2024.
- [80] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [81] P. von Platen, S. Patil, A. Lozhkov, P. Cuenca, N. Lambert, K. Rasul, M. Davaadorj, D. Nair, S. Paul, W. Berman, Y. Xu, S. Liu, and T. Wolf. Diffusers: State-of-the-art diffusion models. <https://github.com/huggingface/diffusers>, 2022.
- [82] B. Wallace, A. Gokul, S. Ermon, and N. Naik. End-to-end diffusion latent optimization improves classifier guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7280–7290, 2023.
- [83] B. Wallace, M. Dang, R. Rafailov, L. Zhou, A. Lou, S. Purushwalkam, S. Ermon, C. Xiong, S. Joty, and N. Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.
- [84] P. Wang, L. Li, Z. Shao, R. Xu, D. Dai, Y. Li, D. Chen, Y. Wu, and Z. Sui. Math-shepherd: A label-free step-by-step verifier for llms in mathematical reasoning. *arXiv preprint arXiv:2312.08935*, 2023.
- [85] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.
- [86] J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

-
- [87] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
 - [88] O. Wiles, C. Zhang, I. Albuquerque, I. Kajić, S. Wang, E. Bugliarello, Y. Onoe, C. Knutsen, C. Rashtchian, J. Pont-Tuset, et al. Revisiting text-to-image evaluation with gecko: On metrics, prompts, and human ratings. *arXiv preprint arXiv:2404.16820*, 2024.
 - [89] X. Wu, S. Huang, and F. Wei. Multimodal large language model is a human-aligned annotator for text-to-image generation. *arXiv preprint arXiv:2404.15100*, 2024.
 - [90] Y. Wu, Z. Sun, S. Li, S. Welleck, and Y. Yang. An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024.
 - [91] Y. Xie, A. Goyal, W. Zheng, M.-Y. Kan, T. P. Lillicrap, K. Kawaguchi, and M. Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.
 - [92] J. Xu, X. Liu, Y. Wu, Y. Tong, Q. Li, M. Ding, J. Tang, and Y. Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2024.
 - [93] Y. Xu, M. Deng, X. Cheng, Y. Tian, Z. Liu, and T. Jaakkola. Restart sampling for improving generative processes. *Advances in Neural Information Processing Systems*, 36:76806–76838, 2023.
 - [94] K. Yang, J. Tao, J. Lyu, C. Ge, J. Chen, W. Shen, X. Zhu, and X. Li. Using human feedback to fine-tune diffusion models without any reward model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8941–8951, 2024.
 - [95] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.
 - [96] L. Yu, J. Lezama, N. B. Gundavarapu, L. Versari, K. Sohn, D. Minnen, Y. Cheng, V. Birodkar, A. Gupta, X. Gu, A. G. Hauptmann, B. Gong, M.-H. Yang, I. Essa, D. A. Ross, and L. Jiang. Language model beats diffusion – tokenizer is key to visual generation. In *The Twelfth International Conference on Learning Representations*, 2024.
 - [97] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.
 - [98] X. Zhou, V. Koltun, and P. Krähenbühl. Simple multi-dataset detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7571–7580, 2022.
 - [99] Z. Zhou, S. Shao, L. Bai, Z. Xu, B. Han, and Z. Xie. Golden noise for diffusion models: A learning framework. *arXiv preprint arXiv:2411.09502*, 2024.

Appendices

A. Experiment Settings

We present our experimental settings below.

A.1. Training Settings

Most models used in our work are pre-trained: in ImageNet, we use the pre-trained SiT-XL model; under Text-to-Image setting, we use the publicly released weights of FLUX.1-dev and PixArt- Σ from the diffusers library [81]. In Section 5, the reported SiT-B and SiT-L are self-trained following the identical architectures and training configurations from [50]. The final numbers included in Figure 11 are from models trained at 800K iterations.

A.2. Sampling Settings

We summarize the sampling settings in our work below.

Configs	Class-conditioned		Text-conditioned	
	SiT-XL	FLUX.1-dev	PixArt- Σ	
ODE solver	2 nd order Heun	Euler	DDIM	
NFEs/iter	50 [†]	30	30	
final denoising steps	250	30	30	
guidance scale	1.0 [‡]	3.5	4.5	
resolution	256	1024	1024	

Table 5 | *Default sampling settings for Class-conditioned and Text-conditioned generation.* [†] In Figure 10 we report numbers with different NFEs/iter; [‡] In Figure 4 we report results with different guidance scales.

A.3. Search Settings

Random Search. During search, we randomly sample a set of i.i.d Gaussian noises $|S|$ as the candidates for each conditioning, generate samples from them with the ODE solver, and select the one with the highest score output by the verifiers as the noise used for final generation. We take the size of S as the primary scaling axis and explore $|S| = 2^k$ for $k \in \{1, 2, 3, 4, 5, 6, 7, 8\}$ in our experiments.

Zero-Order Search. There are three tunable parameters in Zero-Order Search: search iterations K , number of neighbors N , and step size λ . As K is the most scalable, we fix it as the primary scaling dimension when studying the behavior of Zero-Order Search. In Figure 6 we additionally investigate the performance of Zero-Order Search when tuning N , as it provides a secondary axis in scaling compute. In Figure 12, we demonstrate the effect of tuning step size λ . We fix $N = 2$ and explore the performance of modifying the values of λ with respect to different values of K .

Expectedly, when λ is small, Zero-Order Search has slightly worse performance and lower compute efficiency; when λ is large, Zero-Order Search suffers from overfitting - the selected set of noises fits too close to the high scoring area of the verifier, leading to loss of diversity. As a result, while it has the best Inception Score among the three, its FID starts to increase once the compute is scaled over 10^3 NFEs. We provide further analysis of this observation in Section B.

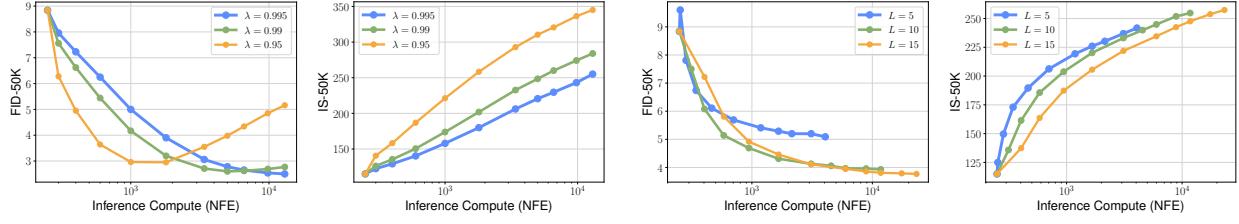


Figure 12 | *Performance of tuning additional parameters for algorithms.* Left: Zero-Order Search with step sizes λ ; Right: Search Over Paths with lengths L . We use SiT-XL and fix the verifier to be the classification logits from DINO.

Search Over Paths. We summarize the hyperparameters for Search Over Paths below

Hyperparameter	Description
initial paths	N The number of paths to start the search with.
paths width	M The number of noises to sample within each path.
search start	σ The time to start search.
backward stepsize	Δb The length of time interval to run ODE solver.
forward stepsize	Δf The length of time interval to run noising process.
paths length	L The NFEs devoted in each backward step.

Table 6 | *Hyperparameters for Search Over Paths.*

For $N > 1$, we start search with N i.i.d samples Gaussian noises and obtain N noisy samples $\{x_\sigma^i\}$. For each x_σ , we then formulate its sampling paths and search over them. Once the search terminates, N different \hat{x}_0 are left, and we run them through the Best-of-N selection to obtain the best one.

In our experiments, we set M and N to be our primary and secondary scaling axis, respectively, as shown in Figure 6. We further explore the behavior of tuning the paths length in Figure 12, where we see that scaling up the paths length can be beneficial to FID but have marginal effect on Inception Score. This supports our claim that the search settings need to be specifically tuned to different application scenarios.

We fix other hyperparameters: $\sigma = 0.11$, $\Delta b = 0.81$, $\Delta f = 0.78$, inspired by the setting in [93].

A.4. Verifier Settings

ImageNet. We consider a total of four verifiers for search on ImageNet. We list the settings below:

- **FID:** we denote the ground truth feature statistics for ImageNet training set μ_{ref} and Σ_{ref} . During search, we select the first 1024 samples randomly and use them to initialize a running mean and covariance $\hat{\mu}$ and $\hat{\Sigma}$. In the following search iterations, for each candidate batch b_i a staged mean $\hat{\mu}_i$ and covariance $\hat{\Sigma}_i$ are calculated with the batch information and the previous running mean and variance. A corresponding FID_i will be obtained between $\hat{\mu}_i$, $\hat{\Sigma}_i$ and μ_{ref} , Σ_{ref} , which is then used as the verifier score. Eventually, $b = \arg \min_i \text{FID}_i$ will be selected, and μ_{ref} and Σ_{ref} are then updated accordingly. Such iteration is repeated until we reach 50000 samples.
- **IS:** The class confidence probability output by the InceptionV3 model is taken as the verifier score.
- **CLIP:** For logits, we use the zero-shot classifier weight W generated by prompt engineering specified in Radford et al. [58] and take the cosine-similarity between the corresponding class entry in W and the image feature to be the verifier score. For the self-supervised version, we directly extract the image feature.

-
- **DINO**: For logits, we use the pre-trained linear classification head provided in Oquab et al. [53] as the verifier. Specifically, we concatenate the `cls` tokens from the last four layers along with the average pooling of the feature from the last layer to formulate the input for the linear head. For the self-supervised version, we directly take the `cls` token from the last layer.

Text-to-Image. We consider a total of four verifiers for search in Text-to-Image setting:

- **Aesthetic**: we take the aesthetic predictor pre-trained on subset of LAION-5B. It consists of a single MLP without any non-linearity and takes the image feature from a pre-trained CLIP-L model as input. The output is on a scale of 0 – 10 rating the images’ aesthetic quality.
- **CLIPScore**: we take the pre-trained CLIP-L model and measure the cosine similarity between visual and text features. Following [23], each text prompt is additionally prefixed with ‘A photo depicts’, and the final score is rescaled by $2.5 * \max(\cos_sim, 0)$.
- **ImageReward**: we take the pre-trained model for approximating human preference from [92] and use the identical evaluation setting.
- **Verifier Ensemble**: We separately run candidates through the above three verifiers, rank the scores output by each, and use the unweighted average rankings as the final score for the Verifier Ensemble.

A.5. Evaluation Setting

ImageNet. Following standard practice, we calculate FID and Inception Score using 50000 synthesized samples. We use randomly generated conditions and a global batch size of 256 for all evaluations. We extracted the ImageNet statistics and calculated FID and IS following Karras et al. [33].

DrawBench. We search for one noise per prompt for generating the sample. For evaluators other than the *LLM Grader*, we simply input the synthesized samples into the pre-trained evaluator models and report the averaged scores over the 200 prompts.

For *LLM Grader*, we prompt the Gemini-1.5 flash model to assess synthesized images from five different perspectives: Accuracy to Prompt, Originality, Visual Quality, Internal Consistency, and Emotional Resonance. Each perspective is rated from 0 to 100, and the averaged overall score is used as the final metric. We include the break-down scores in Table 7, and in Figure 16 we present the detailed prompt. We observe that search can be beneficial to each scoring category of the *LLM Grader*.

T2I-CompBench. For each prompt we search for two noises and generate two samples. During evaluation, the samples are splitted into six categories: `color`, `shape`, `texture`, `spatial`, `numeracy`, and `complex`. Following Huang et al. [30], we use the BLIP-VQA model [43] for evaluation in `color`, `shape`, and `texture`, the UniDet model [98] for `spatial` and `numeracy`, and a weighted averaged scores from BLIP VQA, UniDet, and CLIP for evaluating the `complex` category.

B. Verifier Hacking Leads to Degeneracy in Evaluation Metrics

Many prior works [6, 21, 37] noticed the overoptimization issue when finetuning diffusion models using pre-trained reward models, that excessively optimizing against a reward model will lead to degeneracy in other evaluation metrics. We have similar observations when we excessively search against a verifier and quickly overfit to its bias.

When search on ImageNet against the DINO or CLIP classification logits, we notice the sudden increasing in FID score once pass a certain search iteration numbers despite the constantly improving Inception

Model	Accuracy↑	Originality↑	Visual↑	Consistency↑	Emotional↑	Overall↑
FLUX.1-dev	89.35	67.58	93.00	97.04	73.99	84.29
	+ 4 search iters	91.33	68.49	93.42	96.99	75.31
	+ 16 search iters	91.95	71.52	93.76	97.24	76.30
	+ 64 search iters	93.83	75.38	93.57	97.04	79.34
PixArt-Σ	84.60	73.29	91.91	95.80	76.34	84.67
	+ 4 search iters	87.88	74.03	91.92	96.29	77.32
	+ 16 search iters	88.15	75.39	91.72	96.04	79.17
	+ 64 search iters	89.30	77.79	92.46	96.68	80.43

Table 7 | *Break-down scores of LLM Grader for FLUX.1-dev and PixArt-Σ*. The evaluation is done on DrawBench with random search and verifier ensemble.

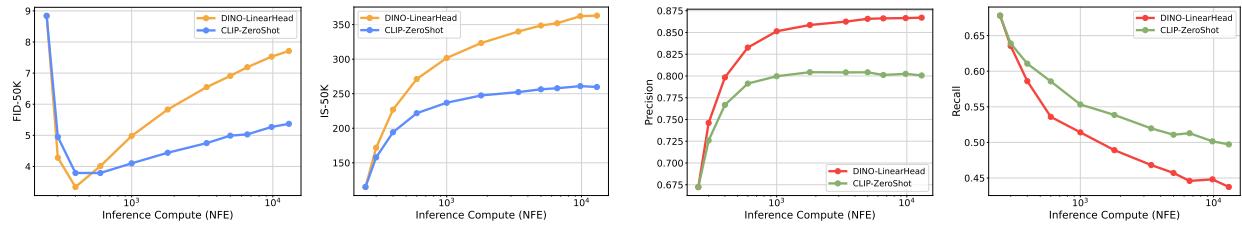


Figure 13 | *Performance of Random Search on ImageNet against DINO and CLIP classification logits*. We use random search on the SiT-XL model and report FID, IS, Precision, and Recall.

Score, as shown in Figure 13. To investigate this issue, we calculate the Precision and Recall [40] and plot them in Figure 13. We see that while Precision increases with search iterations, demonstrating the consistent improvement in sample quality, Recall decreases with search iterations, implying the loss of diversity of the sample set.

We credit this to the DINO and CLIP classification verifiers. When searching against these verifiers, we only operate on a per-noise basis - select the one noise whose corresponding sample has the highest classification score. Therefore, when the search iterations increase, our final set of the selected noises will get closer to the high scoring regions of the classification verifiers. This have two consequences: 1) the selected noises overfit to the verifiers and degenerate other metrics; 2) the selected noises cluster around the high scoring regions and disregard the overall variance of the final set. We deem the latter to be more impactful on the evaluated FID, since FID is known to take great account of the diversity of the generated samples. The Zero-Order Search and Search over Paths we proposed in Section 3.2 alleviate this issue to some extent by searching in the local neighborhood of the Gaussian noise \mathbf{n} sampled at the beginning or at the intermediate sampling steps. However, if we expand the neighborhood range for Zero-Order Search as shown in Figure 12, it will suffer from the diversity issue as well.

A more fundamental solution would be to use the verifiers operating on a population basis and taking into account of the global structure of the set of selected noises. From the trivial example in Figure 3, we see that such verifiers could be effective. We leave further exploration to future works.

Consequence (1) is better demonstrated in Figure 8. We see that over-search against Aesthetic Score will lead to degeneracy in CLIPScore, and vice versa.

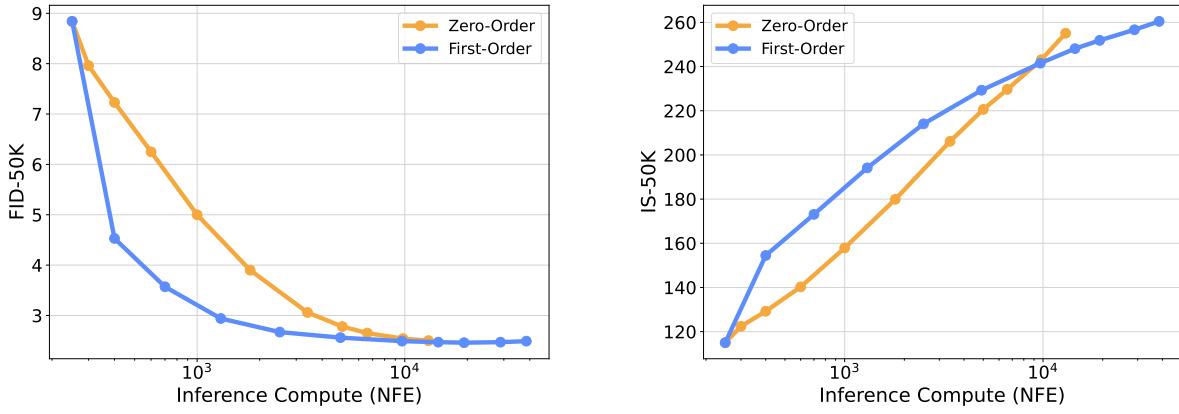


Figure 14 | *Comparison between Zero-Order and First-Order Search.* We use the SiT-XL model and fix the verifier to be the DINO-LinearHead. The Inference Compute is aligned via the rough estimation of $\text{cost}(\text{backward}) \sim 3 \times \text{cost}(\text{forward})$.

C. Zero-Order and First-Order Search

Since many verifiers are differentiable, we also investigate First-Order Search on ImageNet guided by the gradient of verifiers. Specifically:

1. we initialize the noise prior with a randomly sampled Gaussian vector \mathbf{n}
2. run \mathbf{n} through the diffusion ODE solver to obtain the sample and its corresponding score output by verifier \mathcal{V}
3. backpropagate through the verifier and the ODE solver to calculate $\nabla_{\mathbf{n}} \mathcal{V}(\mathbf{n})$
4. update \mathbf{n} via gradient descent: $\mathbf{n}' = \mathbf{n} - \eta \nabla_{\mathbf{n}} \mathcal{V}(\mathbf{n})$, and repeat step 2-4.

Due to the iterative nature of diffusion sampling process, step 2 will incur prohibitive memory cost if naively backpropagating through the ODE solver. To alleviate this issue, we perform gradient checkpointing [9] on each ODE integration step following [5, 52, 82]. This discards the intermediate activation values and re-calculate them using one extra model forward call during backpropagation, thus greatly reducing space complexity at the cost of slightly increased execution time.

We also note that performing naive gradient descent in step 4 might push the updated \mathbf{n}' outside the Gaussian manifold, resulting in training and sampling inconsistency. To resolve this, we simply rescale \mathbf{n}' so that its norm is consistent with the norm of i.i.d Gaussian vectors⁶.

In Figure 14 we include the comparison between Zero-Order Search and First-Order Search. We fix the learning rate to be $\eta = 0.01$ for First-Order Search to roughly match the step size of Zero-Order Search with $\lambda = 0.995$. At a best estimation we attribute the overhead of gradient checkpointing as twice the number of model forward calls, making each iteration 3× costly than without backpropagation.

With inference compute roughly aligned, although First-Order Search shows faster convergence speed over Zero-Order, we see that it does not demonstrate a significant margin when continuously scaling up compute, despite its higher memory cost⁷ and worse scalability on large models. However, by its gradient-guided nature, First-Order Search can be advantageous in tasks with more fine-grained objectives, such as image editing, inpainting, and solving inverse problems [5, 35, 52, 82].

⁶In \mathbb{R}^d , the norm of isotropic Gaussian vectors is distributed according to the chi-squared distribution with d degrees of freedom.

⁷Gradient checkpointing still requires $O(\sqrt{n})$ space complexity [9], with n being the number of layers inside the model.

Verifiers	Aesthetic	CLIPScore	ImageReward
-	5.79	0.71	0.97
CLIP-SSL + 4 search iters	5.76	0.71	0.99
+ 16 search iters	5.72	0.71	1.04
DINO-SSL + 4 search iters	5.79	0.71	0.99
+ 16 search iters	5.78	0.70	1.03
SigLIP-SSL + 4 search iters	5.79	0.70	1.02
+ 16 search iters	5.75	0.70	1.02

Table 8 | *Performance of self-supervised verifiers on DrawBench.* All numbers are from FLUX.1-dev with random search. The first row is the reference performance without search.

D. Self-Supervised Verifiers have Marginal Effect in Text-to-Image Setting

Following Section 3.1, we investigate the performance of self-supervised verifiers in text-to-image setting. Apart from DINO and CLIP, we additionally incorporate SigLIP [97] as an extension to CLIP. Different from ImageNet where the self-supervised verifiers are good surrogate for classification verifiers, on DrawBench they do not demonstrate the expected performance, as shown in Table 8.

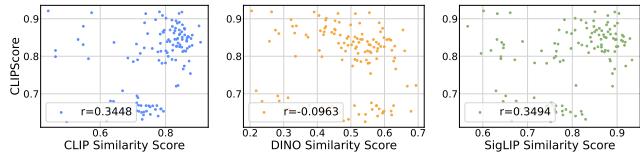


Figure 15 | From Left to Right: Correlation of CLIP, DINO, and SigLIP feature similarity score with CLIPScore. All points are generated from FLUX.1-dev.

From Figure 15, we observe much weaker metric correlations comparing with self-supervised verifiers in ImageNet. We credit this observation to the following:

1) evaluation metrics in text-to-image settings usually focus on more nuanced perspectives: visual-text alignment, composition correctness, human preferences, etc. Even the aesthetic predictor has its bias - it prefers stylized images over others [11]. On the other hand, self-supervised verifiers essentially select the samples with smallest trajectory curvature in the feature space, which implies a stabler sampling process and thus potentially higher sample quality. Yet, by the subtle and holistic nature of evaluation in text-to-image settings [42], such "higher sample quality" may not align with the specific perspectives each metric focuses on. For example, under the same text prompt, an image with high visual quality but misaligned content might not be preferred over an image with slightly degraded visual quality but richer and more aligned visual content.

2) The rich conditionings and extensive fine-tuning in text-to-image models on large scale datasets might lead to different sampling dynamics comparing to the small class-conditioned models trained on ImageNet. This may lead to failure of the self-supervised verifier themselves, as the low trajectory curvature measured in feature space might no longer be indicative of the sample quality.

This also calls for the design of task specific verifiers. From the self-supervised verifiers across class-conditioned and text-conditioned tasks, we see that the effectiveness of verifiers can be highly task-dependent. Therefore, to conduct search that's better aligned with desired objectives, we deem it necessary to have verifiers designed specifically for each task; during search, it's also very important to avoid hacking to the specific bias of each verifier. We have proposed some simple methods in our work, and we leave further explorations of this problem to future works.

"You are a multimodal large-language model tasked with evaluating images generated by a text-to-image model. Your goal is to assess each generated image based on specific aspects and provide a detailed critique, along with a scoring system. The final output should be formatted as a JSON object containing individual scores for each aspect and an overall score. Below is a comprehensive guide to follow in your evaluation process:

1. Key Evaluation Aspects and Scoring Criteria:

For each aspect, provide a score from 0 to 10, where 0 represents poor performance and 10 represents excellent performance. For each score, include a short explanation or justification (1-2 sentences) explaining why that score was given. The aspects to evaluate are as follows:

a) Accuracy to Prompt

Assess how well the image matches the description given in the prompt. Consider whether all requested elements are present and if the scene, objects, and setting align accurately with the text. Score: 0 (no alignment) to 10 (perfect match to prompt).

b) Creativity and Originality

Evaluate the uniqueness and creativity of the generated image. Does the model present an imaginative or aesthetically engaging interpretation of the prompt? Is there any evidence of creativity beyond a literal interpretation? Score: 0 (lacks creativity) to 10 (highly creative and original).

c) Visual Quality and Realism

Assess the overall visual quality, including resolution, detail, and realism. Look for coherence in lighting, shading, and perspective. Even if the image is stylized or abstract, judge whether the visual elements are well-rendered and visually appealing. Score: 0 (poor quality) to 10 (high-quality and realistic).

d) Consistency and Cohesion

Check for internal consistency within the image. Are all elements cohesive and aligned with the prompt? For instance, does the perspective make sense, and do objects fit naturally within the scene without visual anomalies? Score: 0 (inconsistent) to 10 (fully cohesive and consistent).

e) Emotional or Thematic Resonance

Evaluate how well the image evokes the intended emotional or thematic tone of the prompt. For example, if the prompt is meant to be serene, does the image convey calmness? If it's adventurous, does it evoke excitement? Score: 0 (no resonance) to 10 (strong resonance with the prompt's theme).

2. Overall Score

After scoring each aspect individually, provide an overall score, representing the model's general performance on this image. This should be a weighted average based on the importance of each aspect to the prompt or an average of all aspects."

Figure 16 | *The detailed prompt for evaluation with the LLM Grader.*

E. More Visualizations on Scaling Behavior

E.1. SiT-XL

The images presented in this section are sampled from the pre-trained SiT-XL in 256 resolution, using 2nd order Heun sampler and guidance scale of 4.0. Each row of images is structured as follows:

- **Left three:** sampled with increasing steps: 10, 20, 250.
- **Right three:** sampled with Zero-Order Search and the DINO classification verifier. We set $N = 2$ and $\lambda = 0.95$ for Zero-Order Search, and the equivalent NFEs invested are 450, 1850, 6650.



Figure 17 | “loggerhead turtle” (33)



Figure 18 | “Sulphur-crested cockatoo” (89)



Figure 19 | “Siberian husky” (250)



Figure 20 | “Arctic wolf” (270)



Figure 21 | “baseball” (429)



Figure 22 | “hammer” (587)



Figure 23 | “volcano” (980)

E.2. FLUX.1-dev

The images presented in this section are sampled from the pre-trained FLUX.1-dev in 1024 resolution, using Euler sampler and guidance scale of 3.5. Each row of images is structured as follows:

- **Left three:** sampled with increasing steps: 4, 16, 30.
- **Right three:** sampled with Zero-Order Search and the Verifier Ensemble. We set $N = 2$ and $\lambda = 0.95$ for Zero-Order Search, and the equivalent NFEs invested are 120, 960, 2880.



Figure 24 | “New York Skyline with ‘Diffusion’ written with fireworks on the sky.”

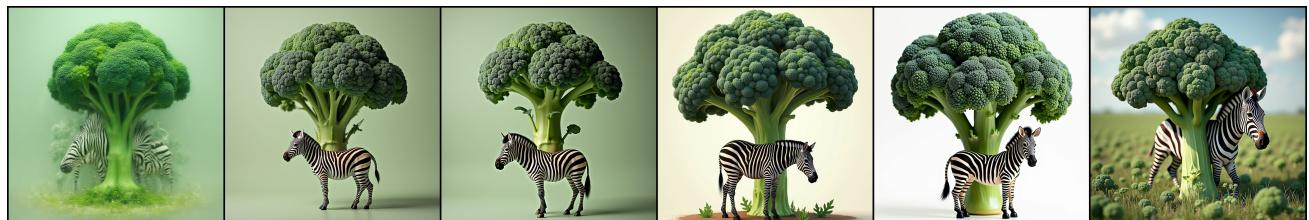


Figure 25 | “A zebra underneath a broccoli.”



Figure 26 | “A car on the left of a bus.”



Figure 27 | “Lego Arnold Schwarzenegger.”



Figure 28 | “An ancient Egyptian painting depicting an argument over whose turn it is to take out the trash.”



Figure 29 | “A storefront with ‘Deep Learning’ written on it.”



Figure 30 | “An IT-guy trying to fix hardware of a PC tower is being tangled by the PC cables like Laokoon. Marble, copy after Hellenistic original from ca. 200 BC. Found in the Baths of Trajan, 1506.”

E.3. PixArt- Σ

The images presented in this section are sampled from the pre-trained PixArt- Σ in 1024 resolution, using DDIM sampler and guidance scale of 4.5. Each row of images is structured as follows:

- **Left three:** sampled with increasing steps: 4, 8, 28.
- **Right three:** sampled with Zero-Order Search and the Verifier Ensemble. We set $N = 2$ and $\lambda = 0.95$ for Zero-Order Search, and the equivalent NFEs invested are 112, 896, 2688.



Figure 31 | “An oil painting portrait of the regal Burger King posing with a Whopper.”

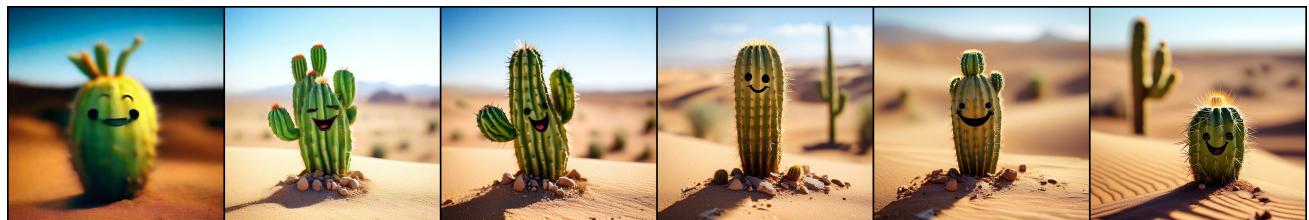


Figure 32 | “A small cactus with a happy face in the Sahara desert.”



Figure 33 | “Greek statue of a man tripping over a cat.”