AdaNAT: Exploring Adaptive Policy for Token-Based Image Generation

Zanlin Ni[1⋆], Yulin Wang[1⋆], Renping Zhou[1], Rui Lu[1], Jiayi Guo[1], Jinyi Hu[1], Zhiyuan Liu[1], Yuan Yao[2†], and Gao Huang[1†]

[1] Tsinghua University
[2] National University of Singapore

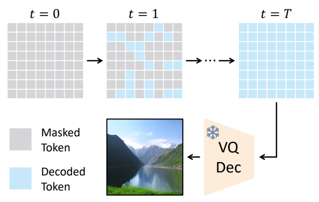
Abstract. Recent studies have demonstrated the effectiveness of token-based methods for visual content generation. As a representative work, non-autoregressive Transformers (NATs) are able to synthesize images with decent quality in a small number of steps. However, NATs usually necessitate configuring a complicated generation policy comprising multiple manually-designed scheduling rules. These heuristic-driven rules are prone to sub-optimality and come with the requirements of expert knowledge and labor-intensive efforts. Moreover, their one-size-fits-all nature cannot flexibly adapt to the diverse characteristics of each individual sample. To address these issues, we propose AdaNAT, a *learnable* approach that automatically configures a suitable policy *tailored for* every sample to be generated. In specific, we formulate the determination of generation policies as a Markov decision process. Under this framework, a lightweight policy network for generation can be learned via reinforcement learning. Importantly, we demonstrate that simple reward designs such as FID or pre-trained reward models, may not reliably guarantee the desired quality or diversity of generated samples. Therefore, we propose an adversarial reward design to guide the training of policy networks effectively. Comprehensive experiments on four benchmark datasets, *i.e.*, ImageNet-256^2&512^2, MS-COCO, and CC3M, validate the effectiveness of AdaNAT. Code and pre-trained models will be released at `https://github.com/LeapLabTHU/AdaNAT`.

Keywords: non-autoregressive Transformers · reinforcement learning · adaptive image generation

1 Introduction

Recent years have witnessed unprecedented growth in the field of AI-generated content (AIGC). In computer vision, diffusion models [9, 45, 48] have emerged as an effective approach. On the contrary, within the context of natural language processing, content is typically synthesized via the generation of discrete tokens using Transformers [4, 18, 43, 57]. Inspired by such discrepancy, there has been a growing interest in investigating this token-based generation paradigm for visual

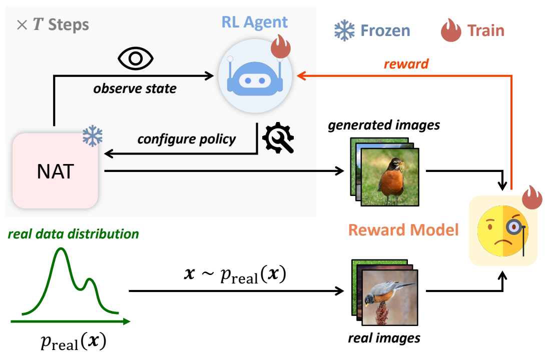
⋆ Equal contributions. †Corresponding Authors.



(a)

	generation policy	
	learnable?	*adaptive?*
existing works [5, 6, 29, 30, 64]	✗	✗
AdaNAT	✓	✓

(b)



(c)

Fig. 1: (a) **The generation process of non-autoregressive Transformers (NATs)** starts from a entirely masked canvas and parallely decodes multiple tokens at each step. The fully decoded tokens are then mapped to the pixel space with a pre-trained VQ-decoder [12]. (b) **Existing works *vs.* AdaNAT.** (c) **Overview of AdaNAT with adversarial reward modeling.** The **RL agent** is our policy network, which configures the suitable policy based on the observed generation status. The **reward model**, providing the probability of a sample being real as the reward, is simultaneously refined to better discriminate between real and fake samples. It is important to note that AdaNAT only learns the proper generation policy upon a pre-trained NAT model. The NAT model itself is kept **frozen** throughout our pipeline.

synthesis [5, 6, 29, 30, 62, 65]. Different from diffusion models, these approaches utilize a discrete data format akin to language models. This makes them straightforward to harness well-established language model optimizations such as refined scaling strategies [4, 24, 42, 59] and advances in model infrastructure [7, 11, 53]. Moreover, explorations in this field may facilitate the development of more advanced, scalable multimodal models with a shared token space [16, 17, 37, 54, 66] as well as general-purpose vision foundation models that unify visual understanding and generation capabilities [30, 55].

In the direction of token-based visual generation, the recently proposed non-autoregressive Transformers (NATs) have exhibited noteworthy potential in terms of both generation quality and computational efficiency [5, 6, 29, 41]. Compared to traditional autoregressive Transformers [10, 12, 39, 62], NATs natively support producing images of decent quality with only 4 to 8 steps. This ability of few-step sampling mainly stems from their "parallel decoding" mechanism, which allows multiple tokens to be decoded simultaneously in each step, as shown in Figure 1a. Nevertheless, this mechanism introduces intricate design considerations, such as determining the number and subset of tokens to decode at each step and setting the temperatures for sampling. Effectively managing these aspects necessitates a complex generation policy comprising numerous hyperparameters, making it difficult for practitioners to utilize these models properly. Existing works alleviate this problem by *manually* developing multiple scheduling functions for policy configuration. However, this approach demands expertise and extensive effort, yet still falls short of capturing the optimal dynamics of the generation process (see Table 5). Furthermore, the *globally shared* policy may lack the flexibility to accommodate the diverse characteristics of different samples.

To address these aforementioned issues, this paper presents AdaNAT. The major insight behind AdaNAT is to consider a *learnable* policy network that automatically configures the policy *adaptively* conditioned on each sample. Consequently, the generation policy tailored for each sample can be obtained with minimal human effort. To implement our idea, a non-trivial challenge lies in designing an effective algorithm to train the policy network. More precisely, due to the non-differentiable nature of the discrete token-based generation process, it is infeasible to straightforwardly adopt standard end-to-end optimization techniques [46]. Hence, we propose to formulate the determination of the optimal generation policy as a Markov decision process (MDP). As a result, our policy network can be naturally defined as an agent, which observes the status of generation, adaptively configures the policy to maximize generation quality, and can be trained through reinforcement learning (*e.g.*, policy gradient).

Importantly, in our problem, designing appropriate reward signals is crucial for training the policy network effectively. To investigate this issue, we first consider two off-the-shelf design choices: 1) standard evaluation metrics like Fréchet Inception Distance (FID) [22] 2) pre-trained image reward models [61]. We experimentally demonstrate that with these designs, although the expected value of rewards can be successfully maximized, our resulting generative model usually fails to produce sufficiently high-quality and diversified images (see Figure 4). In other words, the policy network tends to "overfit" these rewards. Inspired by this phenomenon, we hypothesize that it might be a rescue to consider a reward that is *updated concurrently with the policy network during the learning process* and *adjusted dynamically under the goal of resisting overfitting.* Therefore, we propose an adversarial reward model, which is formulated as a discriminator similar to that used in generative adversarial networks (GANs) [19]. When the policy network learns to maximize the reward, we refine the reward model simultaneously to better distinguish between real and generated samples. In this way, the policy network is effectively prevented from overfitting a static objective, and we observe a balanced diversity and fidelity of the generated images. An overview of our proposed AdaNAT is illustrated in Figure 1c.

Empirically, the effectiveness of AdaNAT is extensively validated on four benchmark datasets, *i.e.*, ImageNet 256×256 [47], ImageNet 512×512 [47], MS-COCO [31], and CC3M [52]. AdaNAT is able to adaptively adjust the generation policy (see Figure 3) and improve the performance of NATs considerably (*e.g.*, 40% relative improvement, see Table 5).

2 Related Work

Reinforcement learning in image generation. The integration of reinforcement learning (RL) for image generation began with early works like [1]. Recently, ImageReward [61] collected a large-scale human preference dataset to train a reward model. This pre-trained reward model has spurred research on using RL to fine-tune diffusion-based image generation models [3,15,67]. Unlike these methods, which optimize the generative model itself, our work uses an RL agent to enhance a frozen generative model (NAT) without tuning it.

Generative adversarial networks and RL. There are several works that use ideas from reinforcement learning to train GANs [50,58,60,63] for text generation, information retrieval, point cloud completion, *etc*. Recently, SFT-PG [14] combines RL with a GAN objective for diffusion-based image generation.Our work differs from these works in many important aspects. First, the main idea is orthogonal. We are interested in exploring a better generation policy *given* a pre-trained generator backbone (which is kept unchanged throughout our method), while previous studies investigate the alternative "RL+GAN" approach to directly train or fine-tune the generator backbone itself. Second, the research problem is different. We focus on token-based generation, while previous works aim to improve GAN or diffusion-based models. Third, compared to SFT-PG [14], we conduct comparative analyses on different reward designs and large-scale experiments while they mainly explored the GAN-based objective and prove their concept on relatively simple datasets (*e.g.*, CIFAR [27] and CelebA [33]).

Non-autoregressive Transformers (NATs) find their roots in machine translation, notably for their quick inference abilities [18,20]. These models are recently employed for image synthesis, producing decent-quality images efficiently, as highlighted by several studies [5,6,29,30,41,64]. As a pioneering work, MaskGIT [6] first demonstrates NAT's effectiveness on ImageNet. It has been further extended for text-to-image generation and scaled up to 3B parameters in Muse [5] and yields remarkable performance. MAGE [30] proposes leveraging NATs to unify representation learning with image synthesis. More recently, AutoNAT [38] optimizes the policies in NATs with FID [22] as the objective. However, as we will show in this paper, the FID-based objective may lead to degenerate solutions. We present more comparisons with AutoNAT in Appendix D.

3 Preliminaries of Non-autoregressive Transformers

In this section, we briefly introduce the non-autoregressive Transformers (NATs) [5, 6, 30] for image generation, laying the basis for our proposed AdaNAT.

3.1 Overview

Non-autoregressive Transformers (NATs) typically utilize a pre-trained VQ autoencoder [12,44,56] to convert images into discrete visual tokens and vice versa. The VQ autoencoder comprises an encoder $\mathcal{E}^{\mathrm{VQ}}$, a quantizer $\mathcal{Q}$ with a learnable codebook e, and a decoder $\mathcal{D}^{\mathrm{VQ}}$. The encoder and quantizer transform an image into a sequence of visual tokens, while the decoder reconstructs the image from these tokens. NATs generate visual tokens in the latent VQ space by training with the masked token modeling (MLM) objective from BERT [8]. This allows the model to predict tokens based on surrounding unmasked tokens, enabling multi-step token generation conditioned on previously predicted tokens.

3.2 Generation via Parallel Decoding

During inference, NATs generate the latent visual tokens iteratively. Specifically, we denote the visual tokens obtained by the VQ encoder as $\boldsymbol{v} = [v_i]_{i=1:N}$, where N is the sequence length. Each visual token v_i corresponds to a specific index of the VQ encoder's codebook. The model starts from an all-`[MASK]` token sequence $\boldsymbol{v}^{(0)}$. At t^{th} step, the model predicts $\boldsymbol{v}^{(t+1)}$ from $\boldsymbol{v}^{(t)}$ by first **parallely decoding** all tokens and then **re-masking** less reliable predictions, as described below[3].

Parallel decoding. Given visual tokens $\boldsymbol{v}^{(t)}$, the model first parallely decodes all of the `[MASK]` tokens to form an initial guess $\hat{\boldsymbol{v}}^{(t+1)}$:

$$\hat{v}_i^{(t+1)} \begin{cases} \sim \hat{p}_{\tau_1^{(t)}}(v_i|\boldsymbol{v}^{(t)}), & \text{if } v_i^{(t)} = \texttt{[MASK]}; \\ = v_i^{(t)}, & \text{otherwise.} \end{cases}$$

Here, $\hat{p}_{\tau_1^{(t)}}(v_i|\boldsymbol{v}^{(t)})$ represents the model's predicted probability distribution at position i, scaled by a temperature $\tau_1^{(t)}$. Meanwhile, confidence scores $\boldsymbol{c}^{(t)}$ are defined for all tokens:

$$c_i^{(t)} = \begin{cases} \log \hat{p}(v_i = \hat{v}_i^{(t+1)}|\boldsymbol{v}^{(t)}), & \text{if } v_i^{(t)} = \texttt{[MASK]}; \\ +\infty, & \text{otherwise.} \end{cases}$$

where $\hat{p}(v_i = \hat{v}_i^{(t+1)}|\boldsymbol{v}^{(t)})$ is the predicted probability for the selected token $\hat{v}_i^{(t+1)}$ at position i.

Re-masking. From the initial guess $\hat{\boldsymbol{v}}^{(t+1)}$, the model then obtains $\boldsymbol{v}^{(t+1)}$ by re-masking the $\lceil m^{(t)} \cdot N \rceil$ least confident predictions:

$$v_i^{(t+1)} = \begin{cases} \hat{v}_i^{(t+1)}, & \text{if } i \in \mathcal{I}; \\ \texttt{[MASK]}, & \text{if } i \notin \mathcal{I}. \end{cases}$$

Here, $m^{(t)} \in [0, 1]$ regulates the proportion of tokens to be re-masked at each step. The set $\mathcal{I}$ comprises indices of the $N - \lceil m^{(t)} \cdot N \rceil$ most confident predictions and are sampled without replacement from $\text{Softmax}(\boldsymbol{c}^{(t)}/\tau_2^{(t)})$[4], where $\tau_2(\cdot)$ is the temperature parameter for re-masking.

The model iterates the process for T steps to decode all `[MASK]` tokens, yielding the final sequence $\boldsymbol{v}^{(T)}$. The sequence is then fed into the VQ decoder to obtain the image $\boldsymbol{x}$:

$$\boldsymbol{x} = \mathcal{D}^{\text{VQ}}(\boldsymbol{v}^{(T)}).$$

Classifier-free guidance. Notably, NATs can also employ classifier-free guidance (CFG) [5, 23] to improve generation quality. This is achieved by extrapolating the logits during the parallel-decoding phase of each timestep t with a guidance scale $w^{(t)}$. We refer readers to [5] for more details.

[3] For the ease of notation, we omit condition $\boldsymbol{c}$.

[4] In practice, this sampling procedure is implemented via Gumbel-Top-k trick [26].

4 AdaNAT

4.1 Motivation

$$\text{Existing works}: \quad m^{(t)}, \tau_1^{(t)}, \tau_2^{(t)}, w^{(t)} = \boldsymbol{\eta}(t) \qquad \textit{pre-defined, static} \tag{1}$$

$$\textbf{AdaNAT}: \quad m^{(t)}, \tau_1^{(t)}, \tau_2^{(t)}, w^{(t)} = \boldsymbol{\eta}_{\boldsymbol{\phi}}(t, \boldsymbol{v}^{(t)}) \qquad \textit{learnable, adaptive} \tag{2}$$

Motivation I: automatic policy acquisition. The flexible parallel decoding scheme endows NATs with an inherent ability for efficient image generation. However, it necessitates an intricate generation policy comprising a variety of hyperparameters for meticulous control. As discussed in Section 3, a T-step generation process introduces $4 \times T$ hyperparameters: $\{m^{(t)}, \tau_1^{(t)}, \tau_2^{(t)}, w^{(t)}\}_{t=0}^{T-1}$. The abundance of these hyperparameters makes it difficult for practitioners to take full advantage of NATs in realistic scenarios. Existing works mainly alleviate this problem by leveraging multiple *pre-defined* scheduling functions (denoted together as $\boldsymbol{\eta}$) to manually configure these hyperparameters (Eq. 1, detailed in Appendix F). Nevertheless, such a manual-design regime relies on a fair amount of expert knowledge and labor efforts, yet still leads to a notably sub-optimal policy (see Table 5). To address this issue, we propose to consider a learnable policy net $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ that is trained to produce the appropriate policy automatically. As a result, a significantly superior generation policy can be acquired with minimal human efforts once a proper learning algorithm for $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ is utilized (which will be discussed later).

Motivation II: adaptive policy adjustment. Furthermore, it is noteworthy that Eq. 1 is a *static* formulation, *i.e.*, the generation of all samples shares the same groups of scheduling functions. In contrast, we argue that every sample has its own characteristics, and ideally the generation process should be adaptively adjusted according to each individual sample. To attain this goal, we propose to dynamically determine the policy for t^{th} generation step conditioned on the current generation status, *i.e.*, the current visual tokens $\boldsymbol{v}^{(t)}$. In other words, $\boldsymbol{v}^{(t)}$ provides necessary information on how the generated sample 'looks like' at t^{th} step, based on which a tailored policy can be derived to better enhance the generation quality. More evidence to validate the rationality of our idea can be found in Table 5 and Figure 3.

Integrating the discussions above, we propose to establish a policy network $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ that directly *learns* to produce the appropriate policy in a *adaptive* manner, as shown in Eq. 2. In the following, we will introduce how to train $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ effectively.

4.2 Policy Network Optimization

Formally, given a pre-trained NAT model parameterized by $\boldsymbol{\theta}$ and a policy network $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ to be trained, our objective is to maximize the expected quality of the generated images:

$$\underset{\boldsymbol{\phi}}{\text{maximize}} \quad J(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{\eta}_{\boldsymbol{\phi}})}[r(\boldsymbol{x})], \tag{3}$$

where $r(\cdot)$ is a function quantifying image quality, as detailed in Section 4.3, and $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{\eta}_{\boldsymbol{\phi}})$ denotes the distribution of the images generated by the NAT model $\boldsymbol{\theta}$, under the generation policy specified by the policy network $\boldsymbol{\eta}_{\boldsymbol{\phi}}$.

It is challenging to solve problem 3 directly. This is caused by the non-differentiability of the generation process, attributed to the discrete nature of the visual tokens in NATs. To address this issue, we propose to reformulate the determination of the generation policy as a Markov decision process (MDP). Under this framework, the policy network $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ can be naturally defined as an agent that observes the generation status and takes actions to configure the policy, which can then be optimized via reinforcement learning.

Markov Decision Process (MDP) formulation. We model the NATs' generation process as a T-horizon Markov Decision Process: $MDP(\mathcal{S}, \mathcal{A}, P, R)$, where:

– $\mathcal{S}$ denotes the state space. Each state $\boldsymbol{s}_t \in \mathcal{S}$ is defined as:

$$\boldsymbol{s}_t \triangleq (t, \boldsymbol{v}^{(t)}), \tag{4}$$

where $\boldsymbol{v}^{(t)}$ is the visual tokens at time t.

– $\mathcal{A}$ represents the action space, with each action $\boldsymbol{a}_t \in \mathcal{A}$ corresponding to the generation policy at time t:

$$\boldsymbol{a}_t \triangleq \left(m^{(t)}, \tau_1^{(t)}, \tau_2^{(t)}, w^{(t)}\right) \tag{5}$$

– $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition probability function. Given the current state $\boldsymbol{s}_t$ and action $\boldsymbol{a}_t$, P models the probability of transitioning to the next state $\boldsymbol{s}_{t+1}$:

$$P(\boldsymbol{s}_{t+1} \mid \boldsymbol{s}_t, \boldsymbol{a}_t) \triangleq \left(\delta_{t+1}, \mathcal{T}(\boldsymbol{v}^{(t+1)} | \boldsymbol{v}^{(t)}; \boldsymbol{a}_t, \boldsymbol{\theta})\right), \tag{6}$$

where $\delta_{t+1}(\cdot)$ is the Dirac delta function that is nonzero only at $t + 1$. $\mathcal{T}(\boldsymbol{v}^{(t+1)} | \boldsymbol{v}^{(t)}; \boldsymbol{a}_t, \boldsymbol{\theta}))$ denotes the transition distribution from visual tokens $\boldsymbol{v}^{(t)}$ to $\boldsymbol{v}^{(t+1)}$. This term encapsulates the visual token transition process as detailed in Section 3.2.

– $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, designed to reflect the quality of the visual output. The reward function is defined as:

$$R(\boldsymbol{s}_t, \boldsymbol{a}_t) \triangleq r\left(\mathcal{D}^{\text{VQ}}(\boldsymbol{v}^{(t)})\right) \cdot \mathbb{I}_{\{t=T\}}, \tag{7}$$

where $r(\cdot)$ measures the quality of the generated image and $\mathcal{D}^{\text{VQ}}$ is the VQ-decoder that converts visual tokens to pixels. This formulation ensures that the reward is assessed only at the final timestep T, as we are only concerned with the quality of the final output. We describe the reward function design in more detail in Section 4.3.

Learning the adaptive policy. Given the MDP formulation, we are able to formalize the policy network as an agent $\boldsymbol{\pi}_{\boldsymbol{\phi}}$ that takes actions to configure the generation policy based on the current state $\boldsymbol{s}_t$:

$$m^{(t)}, \tau_1^{(t)}, \tau_2^{(t)}, w^{(t)} \sim \boldsymbol{\pi}_{\boldsymbol{\phi}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t) \tag{8}$$

As $\boldsymbol{s}_t$ includes both the decoding step t and the current visual tokens $\boldsymbol{v}^{(t)}$, the policy network is able to perform adaptive configuration selection. Notably, the agent is defined as a stochastic version of the original policy network $\boldsymbol{\eta}_{\boldsymbol{\phi}}$ to balance exploration and exploitation in reinforcement learning:

$$\boldsymbol{\pi}_{\boldsymbol{\phi}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t) \triangleq \mathcal{N}\big(\boldsymbol{\eta}_{\boldsymbol{\phi}}(\boldsymbol{s}_t), \sigma \boldsymbol{I}\big), \tag{9}$$

where $\mathcal{N}(\cdot)$ denotes a multivariate normal distribution, and σ is a hyperparameter controlling the exploration level. At test time, we simply take the mean of the distribution as the action $\boldsymbol{a}_t$, which reduces to the deterministic policy $\boldsymbol{\eta}_{\boldsymbol{\phi}}(\boldsymbol{s}_t)$.

We train $\boldsymbol{\pi}_{\boldsymbol{\phi}}$ to maximize the expected reward over the generation process:

$$J(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\phi}}}\left[\sum_{t=0}^{T} R(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] = \mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\phi}}}\big[R(\boldsymbol{s}_T, \boldsymbol{a}_T)\big], \tag{10}$$

To effectively estimate policy gradient, we generally follow the clipped surrogate objective in Proximal Policy Optimization (PPO) algorithm [51]:

$$\begin{aligned} L^{PPO}(\boldsymbol{\phi}) = \mathbb{E}_t\Bigg[& \min\Big(\rho_t(\boldsymbol{\phi})\hat{A}_t, \mathrm{clip}\left(\rho_t(\boldsymbol{\phi}), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\Big) \\ & - c\big(V_{\boldsymbol{\phi}}(\boldsymbol{s}_t) - R(\boldsymbol{s}_T, \boldsymbol{a}_T)\big)^2\Bigg], \end{aligned} \tag{11}$$

where $\rho_t(\boldsymbol{\phi}) = \frac{\boldsymbol{\pi}_{\boldsymbol{\phi}}(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\boldsymbol{\pi}_{\boldsymbol{\phi}_{\text{old}}}(\boldsymbol{a}_t|\boldsymbol{s}_t)}$ denotes the probability ratio of the new policy to the old policy for taking action $\boldsymbol{a}_t$ in state $\boldsymbol{s}_t$, $V(\boldsymbol{s}_t)$ is a learned state-value function, $\hat{A}_t$ is the advantage estimate at timestep t, and ϵ, c are hyperparameters. The advantage estimate $\hat{A}_t$ is calculated as:

$$\hat{A}_t = -V(\boldsymbol{s}_t) + R(\boldsymbol{s}_T, \boldsymbol{a}_T), \tag{12}$$

We provide more details about the PPO algorithm in the Appendix A.

4.3 Reward Design

One of the key aspects in training our reinforcement learning agent is the design of the reward function. In this section, we start with straightforward reward designs and discuss their challenges. Then, we propose an adversarial reward design that effectively addresses these challenges.

A: Pre-defined evaluation metric. The most straightforward reward design is to employ commonly used evaluation metrics in the image generation task such as Fréchet Inception Distance (FID). However, we observe two challenges in practice. *First*, statistical metrics face challenges in providing sample-wise reward signals. Evaluation metrics in image generation tasks are usually statistical, *i.e.*, computed over a large number of generated images. For example, the common practice in ImageNet 256×256 benchmark is to evaluate the FID and IS score over 50K generated images [2, 40], which makes it challenging to attribute the reward to specific actions. In practice, we find this lack of informative feedback leads to failure in training the adaptive policy network, as detailed in Appendix B. *Second*, better metric scores do not necessarily translate into better visual quality. As shown in Figure 4a, even when the evaluation metric FID is successfully optimized (*e.g.*, using a non-adaptive variant of AdaNAT), the generated images may still suffer from poor visual quality. This underscores the limitations of directly adopting evaluation metrics as optimization objectives and motivates us to find better alternatives for the reward function.

B: Pre-trained reward model. Another option is to adopt a pre-trained, off-the-shelf reward model [61] specialized at assessing the visual quality of images. This approach addresses the two challenges in option A as the reward signal is now 1) provided for individual images and 2) more consistent with the image quality. However, we observe that the generated images in this case tend to converge on a similar style with relatively low diversity, as shown in Figure 4b. Furthermore, establishing a proper image reward model often involves collecting large-scale annotated human preference data and training additional deep networks, a procedure that is generally costly. Therefore, the access to a pre-trained image reward model may not be assumed in all the scenarios.

Ours: Adversarial reward modeling. One shared issue with the aforementioned two designs is that, while the expected value of rewards can be effectively maximized, the resultant images exhibit unintended inferior quality or limited diversity. In other words, the policy network tends to "overfit" these rewards. Motivated by this observation, we hypothesize that it might be a rescue to consider a reward that is updated concurrently with the policy network during the learning process and adjusted dynamically with the goal of resisting overfitting. To this end, we propose adversarial reward modeling, where we learn an adversarial reward model $r_{\boldsymbol{\psi}}$ together with the policy network. Specifically, we formulate $r_{\boldsymbol{\psi}}$ as a discriminator akin to that in GANs [19] and establish a minimax game between the policy network and $r_{\boldsymbol{\psi}}$:

$$\underset{\boldsymbol{\phi}}{\text{maximize}} \quad J(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{\pi}_{\boldsymbol{\phi}})} \left[r_{\boldsymbol{\psi}}(\boldsymbol{x}) \right], \quad \text{(Maximize Eq. 11 in practice)} \tag{13}$$

$$\underset{\boldsymbol{\psi}}{\text{minimize}} \quad L(\boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{\pi}_{\boldsymbol{\phi}})} \left[\log r_{\boldsymbol{\psi}}(\boldsymbol{x}) \right] + \mathbb{E}_{\boldsymbol{x} \sim p_{\text{real}}(\boldsymbol{x})} \left[\log(1 - r_{\boldsymbol{\psi}}(\boldsymbol{x})) \right], \tag{14}$$

where p_{real} denotes the distribution of real images. As the policy tries to maximize the reward, the reward model is refined simultaneously to better distinguish between real and generated samples. Consequently, we effectively curb the policy network from overfitting a static objective, and more balanced diversity and fidelity of the generated images are also observed (see Figure 4c). Moreover, the adversarial reward model offers immediate, sample-wise reward signals, contrasting with the statistical metrics used in option A; and it obviates the need for expensive human preference data required by the pre-training of reward models in option B. We summarize the training procedure for AdaNAT in Algorithm 1.

Algorithm 1 Training Procedure for AdaNAT

```
1: Input: Policy network $\boldsymbol{\pi}_{\boldsymbol{\phi}}$ and adversarial reward model $r_{\boldsymbol{\psi}}$
2: for $i = 0, 1, 2, \ldots$ do
3:     # Policy network optimization
4:     Sample trajectories $\tau \sim \boldsymbol{\pi}_{\boldsymbol{\phi}_{\text{old}}}$
5:     Update $\boldsymbol{\pi}_{\boldsymbol{\phi}}$ with the gradient $\nabla_{\boldsymbol{\phi}} L^{PPO}(\boldsymbol{\phi})$    ▷ Refer to Eq. 11
6:     $\boldsymbol{\phi}_{\text{old}} \leftarrow \boldsymbol{\phi}$
7:     # Reward model optimization
8:     Sample images $\boldsymbol{x}_{\text{real}} \sim p_{\text{real}}$, $\boldsymbol{x}_{\text{fake}} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{\pi}_{\boldsymbol{\phi}})$
9:     Update $r_{\boldsymbol{\psi}}$ with the gradient $\nabla_{\boldsymbol{\psi}} L(\boldsymbol{\psi})$    ▷ Refer to Eq. 14
10: end for
```

5 Experiments

Setups. Consistent with prior work [5,6,30], we utilize a pretrained VQGAN [12] with a 1024-codebook for image-token conversion. Our NAT models adopt the U-ViT [2] architecture, a Transformer type tailored for image generation, in two sizes: AdaNAT-S (13 layers, 512 dimensions) and AdaNAT-L (25 layers, 768 dimensions). We use a patch size of 2 for ImageNet 512×512 to manage the higher token count. We perform the optimization loop of AdaNAT in Algorithm 1 for 1000 iterations using Adam optimizer [25]. In practice, our adaptive policy network reuses the off-the-shelf NAT model's output feature $f_{\boldsymbol{\theta}}(\boldsymbol{v}^{(t)})$ as the generation status inputs, which we find to be more efficient. Notably, the pre-trained NAT model is kept fixed and there is no need to back-propagate gradient through it throughout the optimization process. For class-conditional generation on ImageNet [47], we report FID-50K following [9,40]. For text-to-image generation on MS-COCO [31] and CC3M [52], we report FID-30K following [2,5]. Due to space limitation, the specifics of NATs pre-training and more implementation details of our method are deferred in Appendix A.

5.1 Main Results

Class-conditional generation on ImageNet. In Tables 1 and 2, we compare our approach with other generative models on ImageNet 256×256 and 512×512, respectively. Despite having fewer parameters and lower inference costs, our AdaNAT-S achieves a competitive FID of 4.54 on ImageNet 256×256. With more computational budget, *e.g.*, 0.3 TFLOPs, AdaNAT-S improves its FID to 3.71, outperforming most baselines. The AdaNAT-L model furthers this improvement, reaching an FID of 2.86 with 8 steps. On ImageNet 512×512, our top-performing model secures an FID of 3.66, surpassing leading models while requiring substantially less computational effort. We also compare the practical latency between AdaNAT and several competitive baselines in Appendix C.
Text-to-image generation on MS-COCO and CC3M. The efficacy of AdaNAT in text-to-image generation is demonstrated on both MS-COCO [31]

Table 1: Class-conditional image generation on ImageNet 256×256 . TFLOPs quantify the computational cost for generating a single image. For DPM-Solver [34] augmented diffusion models (marked with †), we follow [34] to tune configurations and report the lowest FID. ‡: methods without classifier-based or classifier-free guidance [9, 23]. Diff: diffusion, AR: autoregressive, NAT: non-autoregressive Transformers.

Method	Type	Params	Steps	TFLOPs↓	FID-50K↓	IS↑
VQVAE-2‡ [44] *(NeurIPS'19)*	AR	13.5B	5120	-	31.1	∼ 45
VQGAN‡ [12] *(CVPR'21)*	AR	1.4B	256	-	15.78	78.3
ADM-G [9] *(NeurIPS'21)*	Diff.	554M	250	334.0	4.59	186.7
ADM-G, ADM-U [9] *(NeurIPS'21)*	Diff.	608M	250	239.5	3.94	215.8
LDM [45] *(CVPR'22)*	Diff.	400M	250	52.3	3.60	247.7
VQ-Diffusion‡ [21] *(CVPR'22)*	Diff.	554M	100	12.4	11.89	-
Draft-and-revise [28] *(NeurIPS'22)*	NAT	1.4B	72	-	3.41	224.6
ADM-G† [9] *(NeurIPS'21)*	Diff.	554M	4	5.3	22.35	-
			8	10.7	8.81	174.2
LDM† [45] *(CVPR'22)*	Diff.	400M	4	1.2	11.74	-
			8	2.0	4.56	262.9
U-ViT-H† [2] *(CVPR'23)*	Diff.	501M	4	1.4	8.45	-
			8	2.4	3.37	235.9
DiT-XL† [40] *(ICCV'23)*	Diff.	675M	4	1.3	9.71	-
			8	2.2	5.18	213.0
USF [32] *(ICLR'24)*	Diff.	554M	8	10.7	9.72	-
MaskGIT‡ [6] *(CVPR'22)*	NAT	227M	8	0.6	6.18	182.1
MaskGIT [6] *(CVPR'22)*	NAT	227M	12	1.4	4.92	-
MaskGIT-RS [6] *(CVPR'22)*	NAT	227M	8	8.4	4.02	-
Token-Critic‡ [29] *(ECCV'22)*	NAT	422M	36	1.9	4.69	174.5
Token-Critic-RS [29] *(ECCV'22)*	NAT	422M	36	8.7	3.75	-
MAGE‡ [30] *(CVPR'23)*	NAT	230M	20	1.0	6.93	-
MaskGIT-FSQ [36] *(ICLR'24)*	NAT	225M	12	0.8	4.53	-
AdaNAT-S	NAT	58M	4	**0.2**	4.54	-
			8	0.3	3.71	224.5
AdaNAT-L	NAT	206M	4	0.5	3.63	-
			8	0.9	**2.86**	265.4

and CC3M [52]. On MS-COCO, AdaNAT-S achieves a FID score of 5.75 with only 0.3 TFLOPs, surpassing competing baselines and even outperforming recent diffusion models [2,35] with lower computational resources. On CC3M, AdaNAT outperforms the advanced non-autoregressive Transformer Muse [5], achieving a FID of 6.83 versus 7.67, and maintains superior performance even when the computational budget of the Muse model is doubled.

5.2 Analysis of AdaNAT

This section presents more analyses of AdaNAT, including its effectiveness, the learned adaptive policy, and comparisons of different reward designs.

Effectiveness of AdaNAT. In Table 5, we analyze the effectiveness of AdaNAT. Incorporating learnability into the generation policy alone leads to a 2.25 decrease in FID, marking a 30% improvement over the baseline. This highlights the potential sub-optimality of the manual designs in prior works [5, 6, 29, 30]. When the adaptive mechanism is introduced, the FID score is further reduced

Table 2: Class-conditional image generation on ImageNet 512×512. †: DPM-Solver [34] augmented diffusion models. ‡: methods without classifier-based or classifier-free guidance [9, 23]. See Table 1 for more details.

Method	Type	Params	Steps	TFLOPs↓	FID-50K↓	IS↑
VQGAN‡ [12] *(CVPR'21)*	AR	227M	1024	-	26.52	66.8
ADM-G [9] *(NeurIPS'21)*	Diff.	559M	250	579.0	7.72	172.7
ADM-G, ADM-U [9] *(NeurIPS'21)*	Diff.	731M	250	719.0	3.85	221.7
ADM-G† [9] *(NeurIPS'21)*	Diff.	559M	8	18.5	16.16	109.2
U-ViT-H† [2] *(CVPR'23)*	Diff.	501M	8	3.4	4.60	286.8
DiT-XL† [40] *(ICCV'23)*	Diff.	675M	8	9.6	5.44	275.0
MaskGIT‡ [6] *(CVPR'22)*	NAT	227M	12	3.3	7.32	156.0
MaskGIT-RS [6] *(CVPR'22)*	NAT	227M	12	13.1	4.46	-
Token-Critic‡ [29] *(ECCV'22)*	NAT	422M	36	7.6	6.80	182.1
Token-Critic-RS [29] *(ECCV'22)*	NAT	422M	36	34.8	4.03	-
AdaNAT-L	NAT	232M	8	**1.2**	**3.66**	297.3

Table 3: Text-to-image generation on MS-COCO; †: DPM-Solver [34] augmented diffusion models.

method	params	steps	TFLOPs↓	FID-30K↓
VQ-Diffusion [21]	370M	100	-	13.86
Frido [13]	512M	200	-	8.97
U-Net† [2]	53M	50	-	7.32
U-ViT† [2]	58M	4	0.4	11.88
		8	0.6	6.37
AdaNAT-S	57M	8	**0.3**	**5.75**

Table 4: Text-to-image generation on CC3M; all models are trained and evaluated on CC3M.

method	params	steps	TFLOPs↓	FID-30K↓
VQGAN [12]	600M	256	-	28.86
LDM-4 [45]	645M	50	-	17.01
Draft-and-revise [28]	654M	72	-	9.65
Muse [5]	500M	8	2.8	7.67
		16	5.4	7.01
AdaNAT-Muse	512M	8	2.8	**6.83**

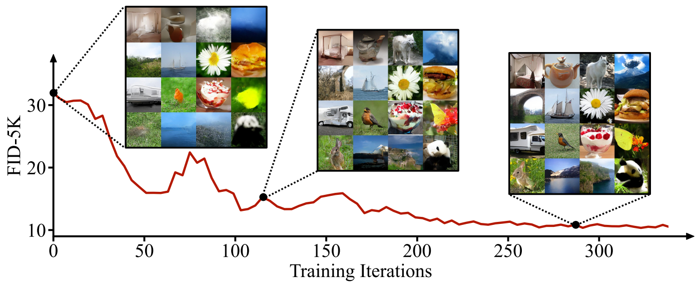


Fig. 2: Optimization process of AdaNAT. We plot the training curve of AdaNAT-L ($T = 4$) on ImageNet 256×256 and visualize samples from different stages. We train the policy network to output suitable configuration, while keeping the pre-trained NAT model *fixed* and only use it for inference. FID-5K is used for efficient evaluation.

to 4.54, achieving an additional 16% improvement over the learnable setup. The above results underscore the effectiveness of a learnable, adaptive policy in the generation process. We offer additional qualitative results in Figure 3 to further illustrate this point.

Table 5: Effectiveness of AdaNAT. We use AdaNAT-S ($T = 4$) on ImageNet-256.

Generation Policy		FID-50K↓
Learnable?	*Adaptive?*	
✗	✗	7.65
✓	✗	5.40 (-2.25)
✓	✓	**4.54** (-3.11)

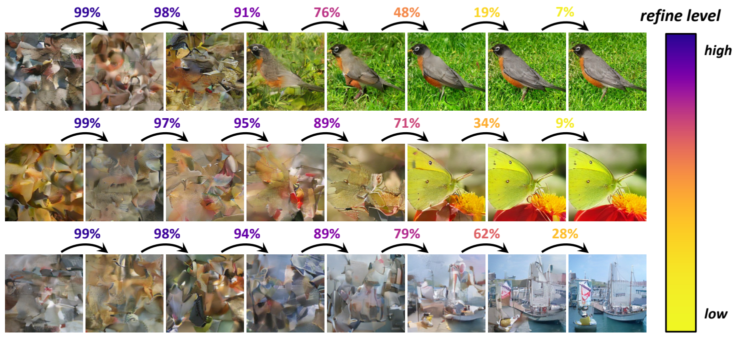
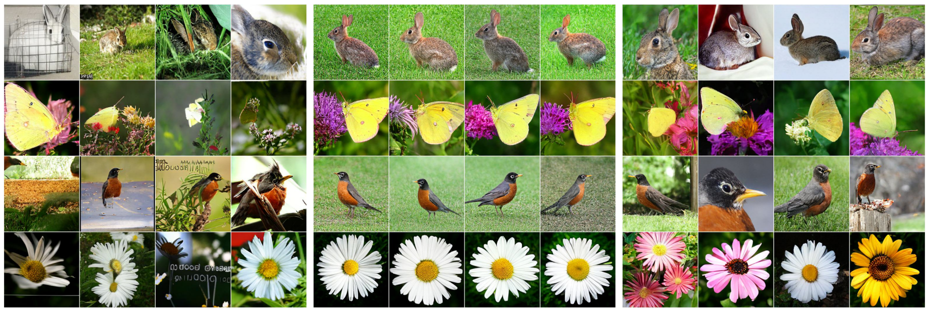


Fig. 3: Visualizing the adaptive policy. The re-masking ratio $m^{(t)}$ (**refine level**), which controls the proportion of least-confident tokens to be refined at each step, is visualized as an example (see Section 3.2 for $m^{(t)}$'s definition). The policy network adaptively reduces $m^{(t)}$ for only minor refinements when the sample already reaches a decent quality; otherwise, it keeps adopting relatively higher $m^{(t)}$ for more adjustments.

In Figure 2, we present the optimization curve of AdaNAT, together with images sampled at different stages of the training process. Initially, the policy network's policy lead to very blurry images, with high FID scores. As training progresses, the policy network gradually refines itself, resulting in lower FID scores and a perceptible improvement in image quality, despite occasional distortions. Finally, the policy network converges with a robust generation policy that yields low FID scores and produces images of decent quality consistently. This offers a more comprehensive understanding of AdaNAT's efficacy in navigating towards more feasible generation policy for non-autoregressive Transformers.

Visualizing the learned adaptive policy. In Figure 3, we visualize the learned adaptive policy, taking the re-masking ratio $m^{(t)}$ as a representative example. The re-masking ratio controls the proportion of least-confident tokens to be refined at each step, so we also call it "**refine level**". For the sample with a relatively simple structure (first row), the refine level is rapidly reduced to a lower level as the sample reaches a decent quality, which restricts the NAT model to only make minor refinements. When the structure of the sample becomes harder (second row), more steps are required to reach a satisfactory quality, and the policy network keeps adopting relatively higher refine levels for more adjustments. For a sample with a complex structure (third row), the policy network consis-



(a) AdaNAT-FID (FID=2.56) (b) AdaNAT-PRM (FID=33.1) **(c) AdaNAT** (FID=2.86)

Fig. 4: Ablation on different reward designs in AdaNAT. (a) AdaNAT-FID: directly using FID [49] as the reward. (b) AdaNAT-PRM: using a pre-trained reward model [61]. (c) AdaNAT: our main approach with adversarial reward model modeling.

tently applies higher refine levels to make more adjustments till the end. These visualizations provide deeper insights into the adaptive mechanism of AdaNAT.

Comparing different reward designs. In Figure 4, we compare the generation results with different reward designs. Figure 4a shows results[5] using Fréchet Inception Distance (FID) [49]. Despite this numerical superiority, we observe inconsistency between the low FID score and image quality, with many images displaying distortions or blurriness. Figure 4b depicts results from a pretrained reward model [61], which produced higher quality images but reduced diversity in poses, scales, and orientations. In contrast, Figure 4c shows our adversarial reward model yielding high-quality images with a broader spectrum of variations, achieving a balanced trade-off between fidelity and diversity.

6 Conclusion

In this paper, we presented AdaNAT, a framework to enhance the generation policy of non-autoregressive Transformers. Our approach uses a policy network, trained via reinforcement learning within a Markov decision process, to adaptively select generation policies based on the sampling context. A key aspect of AdaNAT is the adversarial reward design, which overcomes limitations of straightforward reward signals, ensuring balanced quality and diversity in generated images. Validation on multiple benchmark datasets highlights AdaNAT's advantages in token-based image synthesis.

[5] Initially, we tried learning the adaptive policy, but FID's inability to provide individual sample-wise reward signals led to convergence issues, prompting us to use a non-adaptive variant (see Appendix B for more details).

Acknowledgements

This work is supported in part by the National Key R&D Program of China under Grant 2021ZD0140407, the National Natural Science Foundation of China under Grants 62321005 and 42327901.

References

1. Bachman, P., Precup, D.: Data generation as sequential decision making. In: NeurIPS (2015)
2. Bao, F., Li, C., Cao, Y., Zhu, J.: All are worth words: a vit backbone for score-based diffusion models. In: CVPR (2023)
3. Black, K., Janner, M., Du, Y., Kostrikov, I., Levine, S.: Training diffusion models with reinforcement learning. In: ICLR (2024)
4. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. In: NeurIPS (2020)
5. Chang, H., Zhang, H., Barber, J., Maschinot, A., Lezama, J., Jiang, L., Yang, M.H., Murphy, K., Freeman, W.T., Rubinstein, M., et al.: Muse: Text-to-image generation via masked generative transformers. In: ICML (2023)
6. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: CVPR (2022)
7. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., et al.: Palm: Scaling language modeling with pathways. JMLR (2023)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. google ai language. In: NAACL (2019)
9. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. In: NeurIPS (2021)
10. Ding, M., Yang, Z., Hong, W., Zheng, W., Zhou, C., Yin, D., Lin, J., Zou, X., Shao, Z., Yang, H., et al.: Cogview: Mastering text-to-image generation via transformers. In: NeurIPS (2021)
11. Du, N., Huang, Y., Dai, A.M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A.W., Firat, O., et al.: Glam: Efficient scaling of language models with mixture-of-experts. In: ICML (2022)
12. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: CVPR (2021)
13. Fan, W.C., Chen, Y.C., Chen, D., Cheng, Y., Yuan, L., Wang, Y.C.F.: Frido: Feature pyramid diffusion for complex scene image synthesis. In: AAAI (2023)
14. Fan, Y., Lee, K.: Optimizing ddpm sampling with shortcut fine-tuning. arXiv preprint arXiv:2301.13362 (2023)
15. Fan, Y., Watkins, O., Du, Y., Liu, H., Ryu, M., Boutilier, C., Abbeel, P., Ghavamzadeh, M., Lee, K., Lee, K.: Reinforcement learning for fine-tuning text-to-image diffusion models. In: NeurIPS (2023)
16. Ge, Y., Ge, Y., Zeng, Z., Wang, X., Shan, Y.: Planting a seed of vision in large language model. In: ICLR (2024)
17. Ge, Y., Zhao, S., Zeng, Z., Ge, Y., Li, C., Wang, X., Shan, Y.: Making llama see and draw with seed tokenizer. In: ICLR (2024)

18. Ghazvininejad, M., Levy, O., Liu, Y., Zettlemoyer, L.: Mask-predict: Parallel decoding of conditional masked language models. In: EMNLP (2019)
19. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NeurIPS (2014)
20. Gu, J., Kong, X.: Fully non-autoregressive neural machine translation: Tricks of the trade. In: ACL (2021)
21. Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., Guo, B.: Vector quantized diffusion model for text-to-image synthesis. In: CVPR (2022)
22. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS (2017)
23. Ho, J., Salimans, T.: Classifier-free diffusion guidance. In: NeurIPS Workshop (2021)
24. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D.d.L., Hendricks, L.A., Welbl, J., Clark, A., et al.: Training compute-optimal large language models. arXiv preprint arXiv:2203.15556 (2022)
25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
26. Kool, W., Van Hoof, H., Welling, M.: Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In: ICML (2019)
27. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
28. Lee, D., Kim, C., Kim, S., Cho, M., HAN, W.S.: Draft-and-revise: Effective image generation with contextual rq-transformer. In: NeurIPS (2022)
29. Lezama, J., Chang, H., Jiang, L., Essa, I.: Improved masked image generation with token-critic. In: ECCV (2022)
30. Li, T., Chang, H., Mishra, S., Zhang, H., Katabi, D., Krishnan, D.: Mage: Masked generative encoder to unify representation learning and image synthesis. In: CVPR (2023)
31. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
32. Liu, E., Ning, X., Yang, H., Wang, Y.: A unified sampling framework for solver searching of diffusion probabilistic models. In: ICLR (2024)
33. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: ICCV (2015)
34. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In: NeurIPS (2022)
35. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. arXiv preprint arXiv:2211.01095 (2022)
36. Mentzer, F., Minnen, D., Agustsson, E., Tschannen, M.: Finite scalar quantization: Vq-vae made simple. In: ICLR (2023)
37. Mizrahi, D., Bachmann, R., Kar, O., Yeo, T., Gao, M., Dehghan, A., Zamir, A.: 4m: Massively multimodal masked modeling. In: NeurIPS (2023)
38. Ni, Z., Wang, Y., Zhou, R., Guo, J., Hu, J., Liu, Z., Song, S., Yao, Y., Huang, G.: Revisiting non-autoregressive transformers for efficient image synthesis. In: CVPR (2024)
39. Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D.: Image transformer. In: ICML (2018)

40. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: ICCV (2023)
41. Qian, S., Chang, H., Li, Y., Zhang, Z., Jia, J., Zhang, H.: Strait: Non-autoregressive generation with stratified image transformer. arXiv preprint arXiv:2303.00750 (2023)
42. Rae, J.W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al.: Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint arXiv:2112.11446 (2021)
43. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR (2020)
44. Razavi, A., van den Oord, A., Vinyals, O.: Generating diverse high-fidelity images with vq-vae-2 (2019)
45. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)
46. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016)
47. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. In: IJCV (2015)
48. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. In: NeurIPS (2022)
49. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: NeurIPS (2016)
50. Sarmad, M., Lee, H.J., Kim, Y.M.: Rl-gan-net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion. In: CVPR (2019)
51. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
52. Sharma, P., Ding, N., Goodman, S., Soricut, R.: Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: ACL (2018)
53. Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., Catanzaro, B.: Megatron-lm: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053 (2019)
54. Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., et al.: Gemini: a family of highly capable multimodal models. arXiv preprint arXiv:2312.11805 (2023)
55. Tian, C., Tao, C., Dai, J., Li, H., Li, Z., Lu, L., Wang, X., Li, H., Huang, G., Zhu, X.: Addp: Learning general representations for image recognition and generation with alternating denoising diffusion process. In: ICLR (2024)
56. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. In: NeurIPS (2017)
57. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
58. Wang, J., Yu, L., Zhang, W., Gong, Y., Xu, Y., Wang, B., Zhang, P., Zhang, D.: Irgan: A minimax game for unifying generative and discriminative information retrieval models. In: SIGIR (2017)
59. Wang, T., Roberts, A., Hesslow, D., Le Scao, T., Chung, H.W., Beltagy, I., Launay, J., Raffel, C.: What language model architecture and pretraining objective works best for zero-shot generalization? In: ICML (2022)

60. Wang, X., Chen, W., Wang, Y.F., Wang, W.Y.: No metrics are perfect: Adversarial reward learning for visual storytelling. arXiv preprint arXiv:1804.09160 (2018)
61. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation. In: NeurIPS (2023)
62. Yu, J., Xu, Y., Koh, J.Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B.K., et al.: Scaling autoregressive models for content-rich text-to-image generation. TMLR (2022)
63. Yu, L., Zhang, W., Wang, J., Yu, Y.: Seqgan: Sequence generative adversarial nets with policy gradient. In: AAAI (2017)
64. Yu, L., Lezama, J., Gundavarapu, N.B., Versari, L., Sohn, K., Minnen, D., Cheng, Y., Gupta, A., Gu, X., Hauptmann, A.G., et al.: Language model beats diffusion–tokenizer is key to visual generation. In: ICLR (2024)
65. Yu, L., Shi, B., Pasunuru, R., Muller, B., Golovneva, O., Wang, T., Babu, A., Tang, B., Karrer, B., Sheynin, S., et al.: Scaling autoregressive multi-modal models: Pretraining and instruction tuning. arXiv preprint arXiv:2309.02591 (2023)
66. Zhan, J., Dai, J., Ye, J., Zhou, Y., Zhang, D., Liu, Z., Zhang, X., Yuan, R., Zhang, G., Li, L., et al.: Anygpt: Unified multimodal llm with discrete sequence modeling. arXiv preprint arXiv:2402.12226 (2024)
67. Zhang, Y., Tzeng, E., Du, Y., Kislyuk, D.: Large-scale reinforcement learning for diffusion models. arXiv preprint arXiv:2401.12244 (2024)