SCAN: Self-Calibrated AutoregressioN for High-Quality Visual Generation

Zhanzhou Feng[1]*, **Qingpei Guo**[2], **Jingdong Chen**[2], **Feng Gao**[3], **Ming Yang**[2], **Shiliang Zhang**[1]
[1]State Key Laboratory of Multimedia Information Processing, School of Computer Science, Peking University, Beijing, China
[2]Ant Group
[3]School of Arts, Peking University, Beijing, China
fengzz@stu.pku.edu.cn

Abstract

Human artists can continuously refine their coarse sketches during artistic creation. This is quiet different from existing autoregressive generation, where a token is determined once sampled. Aiming to flexibly refine the generated contents, this paper presents a Self-Calibrated AutoregressioN (SCAN) model capable of self-evaluating and refining generation quality without regenerating the entire image. We unify image token generation and quality evaluation into a single autoregressive model, formulating both tasks as categorical prediction problems. During inference, the model first generates a coarse initial image, then iteratively refines the lowest-quality patches until satisfactory image quality is achieved. Experimental results demonstrate that SCAN effectively handles diverse real-world generation errors and achieves a promising balance between image quality and speed. For example, SCAN-XL achieves an FID of 2.10 and an IS of 326.1, surpassing the LlamaGen-XL by 1.29 (+38%) in FID and 99.0 (+43.6%) in IS, with a 5.6× speedup (19.76s → 3.56s). Compared to recent works, SCAN improves FID and speed by +18.3% and +23% over VAR-d20, and by +7% and +46% over RandAR-XL.

Code — https://github.com/ZhanzhouFeng/SCAN

Introduction

Autoregressive (AR) models have significantly advanced the development of Large Language Models (LLMs) (Touvron et al. 2023; Jiang et al. 2025; Song, Liu, and Shou 2025), visual perction (Xiao et al. 2025; Feng and Zhang 2023a) and Vision-Language Models (VLMs) (Zhu et al. 2023; Zhou et al. 2025; Sun et al. 2023, 2025). The core idea is to predict the sequence of tokens step-by-step, *i.e.*, once a token is generated, it is fixed and serves as contextual conditioning for subsequent predictions. This process decomposes long-sequence generation into a series of next-token predictions, thereby reducing prediction complexity. In the field of visual generation, AR models are emerging as a promising approach due to their impressive performance in multi-modal instruction following and scalability (Wang et al. 2024; Kondratyuk et al. 2023; Wu et al. 2024b; Song et al. 2024). However, a key limitation of existing AR visual generation is its one-way prediction nature—a token is determined once it is sampled, even if it presents unsatisfactory visual quality or conflicts with later tokens.

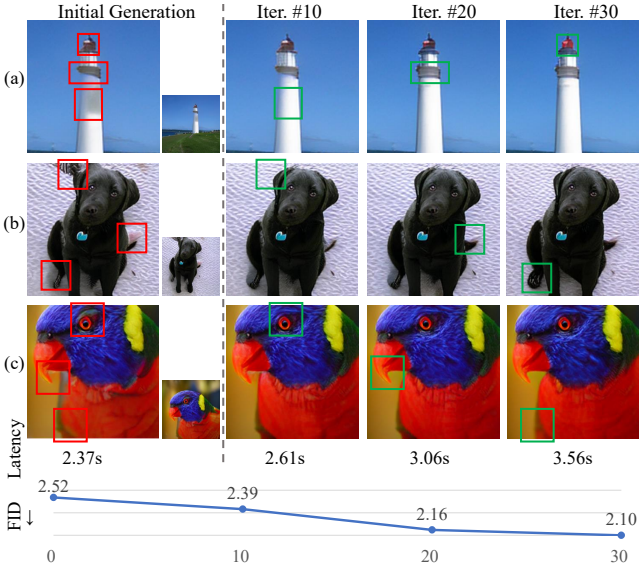

Figure 1: Three examples of the self-calibrated autoregression, showing the initial generation and zoomed-in areas of various refinement iterations. Flaws and refined areas are highlighted with red and green boxes, respectively. Latency and FID scores with 0, 10, 20, and 30 iterations are reported below, where lower FID indicates better performance.

Human artists often begin a painting with bold strokes on a blank canvas, then keep refining the designs or any flaws of the draft afterwards. With the composition in place, they can better identify problems and fix artifacts, leading to a polished work. Diffusion models (Peebles and Xie 2023; Feng et al. 2025a; Xie et al. 2025; Chen et al. 2025; Wan et al. 2024) and AR models (Wang et al. 2024; Wu et al. 2024b; Feng et al. 2025b) are the two main visual generation methods. Diffusion models (Rombach et al. 2022; Song, Liu, and Shou 2024; Hua et al. 2025; Chen, Chen, and Song 2025) perform multi-step denoising on all patches at each step, allowing them to fix and refine previous predictions. Yet, AR models (Wang et al. 2024; Kondratyuk et al. 2023; Wu et al.

*Work done during internship at Ant Group.

2024b) generate sequences of tokens in one-way, where each token is determined once sampled from a predicted distribution. The randomness in categorical probability sampling may lead to inconsistent and low-quality patches. As the length of the sequence grows, the cumulative probability of low-quality patches may degrade the overall generation quality. Current AR methods lack explicit mechanisms to detect or resolve such inter-token conflicts during generation. Recently, VAR (Tian et al. 2024) introduces a coarse-to-fine next-scale prediction, where fine-scale tokens supplement high-resolution details, but cannot fix flaws in the coarse-scale structure.

We aim to improve AR generation models by equipping them with the capability of self-calibrated flaw correction. This can be achieved by 1) evaluating the quality of each generated patch, and 2) recursively refining the low-quality ones. However, there is no external patch-level quality annotation available. Traditional GANs (Isola et al. 2017; Sauer, Schwarz, and Geiger 2022; Kang et al. 2023) are unsuitable as they assess entire images rather than specific patches, and their gradients cannot be back-propagated through vector quantization (Van Den Oord, Vinyals et al. 2017; Esser, Rombach, and Ommer 2021). Moreover, most existing AR models generate image patches sequentially in a raster order, with each patch strictly conditioned on preceding outputs. This design makes it hard to regenerate specific patches without reprocessing the entire image.

To tackle above challenges, we introduce a self-calibrated autoregression model (SCAN) capable of evaluating and refining low-quality patches. SCAN unifies image generation and quality evaluation within a single AR model. Image generation is formulated as a categorical classification task in VAE codebooks (Van Den Oord, Vinyals et al. 2017). Quality evaluation is formulated as a binary classification to distinguish high- and low-quality tokens. The tokens of ground truth images serve as high-quality positive samples. Low-quality negative samples are constructed from three sources, *i.e.*, random noise content, location-permuted patches, and teacher forcing prediction. During training, we randomly replace ground truth patches with negative samples at a ratio and train the quality evaluation function. The predicted probability serves as the quality score.

The quality score allows SCAN to spot and refine low-quality cues. To replace the raster-order generation in previous decoder-only AR models, we propose two types of input tokens, *i.e.*, position instruction tokens p for targeted patch generation and image tokens x for quality evaluation. We use position embeddings as position instructions p, enabling regeneration of position-specified low-quality tokens without reprocessing the entire sequence. It also allows for a parallel generation via next set prediction to accelerate refinement. During inference, SCAN first generates an initial image $\boldsymbol{x}^{(0)}$ and its patch-level quality scores $\boldsymbol{s}^{(0)}$. It then iteratively selects lowest-scoring tokens for refinement. The number of iterations can be manually set or based on application requirements, allowing for a flexible balance between computational cost and output quality.

SCAN is built on LlamaGen architecture (Sun et al. 2024), including three scales: SCAN-L (343M), SCAN-XL (775M), and SCAN-XXL (1.4B). In Fig. 1, iterative refinement progressively corrects flaws in generated images. Experimental results demonstrate that the SCAN significantly improves image quality and efficiency. SCAN-XL achieves the FID of 2.10 and IS of 326.1, surpassing baseline LlamaGen-XL (Sun et al. 2024) by 1.29 (+38%) in FID and 99.0 (+43.6%) in IS, with a 5.6× speedup. Compared to recent AR visual generation works of similar scale (∼700M), it outperforms VAR-d20 (Tian et al. 2024) by 0.47 (+18.3%) in FID and 23.5 (+7.8%) in IS with a 23% speedup, and RandAR-XL (Pang et al. 2024) by 0.15 (+7%) in FID and 8.3 (+3%) in IS with a 46% speedup.

SCAN combines coarse initial generation with iterative refinement to boost the output quality. To the best of our knowledge, this is an initial effort to empower autoregressive generation with self-evaluation and correction capabilities. We hope SCAN helps alleviate the limitations of existing AR models and inspires further exploration on self-calibrated generation.

Related Work

Masked Image Modeling (MIM). MIM methods predict image tokens from a masked sequence using bidirectional attention in a BERT-like (Devlin 2018) way, including MaskGIT (Chang et al. 2022), Muse (Chang et al. 2023), MagViT (Yu et al. 2023a) and MAR (Li et al. 2024). The design of masked placeholders with position information enables parallel token prediction and flexible generation order, reducing the number of steps required to predict a complete image and thus accelerating the inference (Chang et al. 2022). However, MIM is typically built on encoder-decoder architectures (Feng and Zhang 2024, 2023b) which lacks support for KV-Cache (Shazeer 2019; Li et al. 2025) and is not directly compatible with LLMs. Different from previous MIM, this work is built on a decoder-only AR and utilizes positional embeddings as instructions to enable specific patch generation and modification.

Decoder-only AR Language Generation. LLMs predict the next token iteratively until the entire sentence is generated. Recent works (Liu et al. 2024; Wu et al. 2024a) suggest that allowing LLMs to engage in internal thinking before producing a response can enhance their performance on reasoning, particularly when they are encouraged to explicitly articulate intermediate reasoning steps (Liu et al. 2024). Some works (Madaan et al. 2024) propose leveraging natural language feedback, both human- (Bai et al. 2022) and machine- (Fu et al. 2023) labeled, as intermediate thinking process to refine outputs. Sources of feedback include human evaluations (Bai et al. 2022), compilers (Yasunaga and Liang 2020), or Wikipedia edits (Schick et al. 2022). However, these approaches require external feedback data and cannot modify individual tokens without reprocessing the entire sequence. Different from these works on language generation, our work introduces the capability to independently evaluate and refine targeted tokens without requiring additional labeled data.

Decoder-only AR Visual Generation. Inspired by the scalability of AR language models, pioneering works explore AR visual generation, including VQGAN (Esser, Rombach,

and Ommer 2021), DALL-E (Ramesh et al. 2021), Parti (Yu et al. 2022), LlamaGen (Sun et al. 2024) in image generation and VideoPoet (Kondratyuk et al. 2023), CogVideo (Hong et al. 2022), MAGVIT-v2 (Yu et al. 2023b) and VILA-U (Wu et al. 2024b) in video generation. AR generation works typically adopt a raster-scan order, predicting tokens in a strictly fixed sequence. Recently, some works have sought to improve this fixed order. RAR (Yu et al. 2024) randomly permutes the original raster order with a probability r that starts at 1 and linearly decays to 0 during training. RandAR (Pang et al. 2024) enables both training and inference random-order generation through the interleaving of positional instruction tokens and image patches as the input. VAR (Tian et al. 2024) replaces the raster order with a coarse-to-fine next-scale prediction, where a higher-resolution image is predicted to supplement the previous coarse-scale prediction.

Different from previous AR works, this work aims to enable visual AR model with the ability to iteratively modify the generated images without introducing extra models and data. Compared to RandAR, our method does not require to refeed the interleaved image patches, thus avoiding redundant computations and speeding up inference. Compared to VAR, where fine-scale tokens cannot correct errors from previous coarse scales, this work can self-revise previously generated low-quality patches.

Method

Framework

Given a content condition $\mathbf{C}$, the decoder-only AR models predict discrete tokens $\boldsymbol{x} = \{x_n\}_{n=1:N}$ sequentially, where each token x_n is generated based on the previously generated ones. The training objective is to maximize the likelihood of the joint distribution on the ground truth sequence, which can be written as:

$$\max_{\theta} \quad \mathrm{p}_\theta(\boldsymbol{x} \mid \mathbf{C}) = \prod_{n=1}^{N} \mathrm{p}_\theta\left(x_n \mid x_{<n}, \mathbf{C}\right), \tag{1}$$

where p_θ denotes the categorical distribution predictor parameterized by θ. Existing decoder-only AR models generate token sequences uni-directionally, where each token is determined once sampled.

This work aims to endow the AR visual generation model with self-calibration capability, enabling token refinement without introducing substantial computation overheads. Our method first generates an initial image $\boldsymbol{x}^{(0)}$, where the superscript "0" denotes the initial generation stage. For each token $x_n^{(0)} \in \boldsymbol{x}^{(0)}$, we evaluate a quality score $s_n^{(0)}$, where a higher score indicates better quality. Based on $s_n^{(0)}$, low quality tokens are hence spotted and refined. Tokens with the lowest quality scores are then iteratively refined over T iterations, until a satisfactory image is achieved.

In order to implement this self-calibrated generation, we introduce two modules into AR: 1) position-specified tokens generation $\mathrm{Gen}(\cdot)$, that generates a token at the specified position instruction token p_n, and 2) token quality evaluation $\mathrm{Eva}(\cdot)$, that predicts the quality score for a token. Specifically, at the t-th refinement, c tokens with the lowest quality scores in $s_n^{(t)}$ are selected for regeneration, where c is a hyperparameter ablated in Sec. . Let the set $\mathcal{I}$ denote these selected tokens. It can be generated as

$$\mathcal{I}^{(t)} = \min(c, \boldsymbol{s}^{(t)}), \tag{2}$$

where $\min(c, \boldsymbol{s})$ returns indices of the c smallest scores in the set $\boldsymbol{s}$. Selected tokens are regenerated under the current context with specified indices in $\mathcal{I}$,

$$x'_n = \mathrm{Gen}(p_n \mid \boldsymbol{x}^{(t)}, \mathbf{C}), \forall n \in \mathcal{I}^{(t)}. \tag{3}$$

After regeneration, the quality scores for these newly generated tokens are evaluated with $\mathrm{Eva}(\cdot)$,

$$s'_n = \mathrm{Eva}(x'_n \mid \boldsymbol{x}^{(t)}, \mathbf{C}), \forall n \in \mathcal{I}^{(t)}. \tag{4}$$

$\mathrm{Gen}(\cdot)$ and $\mathrm{Eva}(\cdot)$ are iteratively executed to refine tokens and evaluate their quality.

The current image $\boldsymbol{x}^{(t)}$ is updated by replacing selected tokens with the newly generated ones, if their quality scores are improved. It can be denoted as,

$$x_n^{(t+1)} = \begin{cases} x'_n, & \text{if } n \in \mathcal{I}^{(t)} \text{ and } s'_n > s_n^{(t)} \\ x_n^{(t)}, & \text{otherwise.} \end{cases} \tag{5}$$

The quality score sequence is updated accordingly:

$$s_n^{(t+1)} = \begin{cases} s'_n, & \text{if } n \in \mathcal{I}^{(t)} \text{ and } s'_n > s_n^{(t)} \\ s_n^{(t)}, & \text{otherwise.} \end{cases} \tag{6}$$

Refinement continues until the quality scores no longer improve or reaches a predefined iteration number T. Our self-calibrated autoregression does not regenerate the entire image. It also can be accelerated by a parallel next-set prediction, *i.e.*, regenerating c low-quality tokens as a set. Therefore, it outperforms existing AR methods in both efficiency and generation quality.

The next parts present the implementation of $\mathrm{Gen}(\cdot)$ and $\mathrm{Eva}(\cdot)$, as well as our training procedure. An illustration of our inference pipeline is shown in Fig. 2 (b), where refinement is performed for 2 iterations with $c = 2$.

Token and Score Prediction

This section unifies position-specified image token generation $\mathrm{Gen}(\cdot)$ and quality evaluation $\mathrm{Eva}(\cdot)$ into a single decoder-only AR model $\mathrm{G}(\cdot)$, and introduces a next set prediction to enable parallel prediction.

Position-specified token generation. To enable the regeneration of a specific image token without reprocessing the entire image, the model should be informed of the spatial location of the tokens to generate next. We utilize position embeddings as position instruction tokens p_n to indicate spatial locations, which are fed into the model G to predict visual content at the n-th position. Function $\mathrm{Gen}(\cdot)$ can be represented as:

$$x_n = \mathrm{G}(p_n \mid \boldsymbol{x}, \mathbf{C}). \tag{7}$$

The position instruction tokens p_n is similar to the masked tokens combined with positional information in

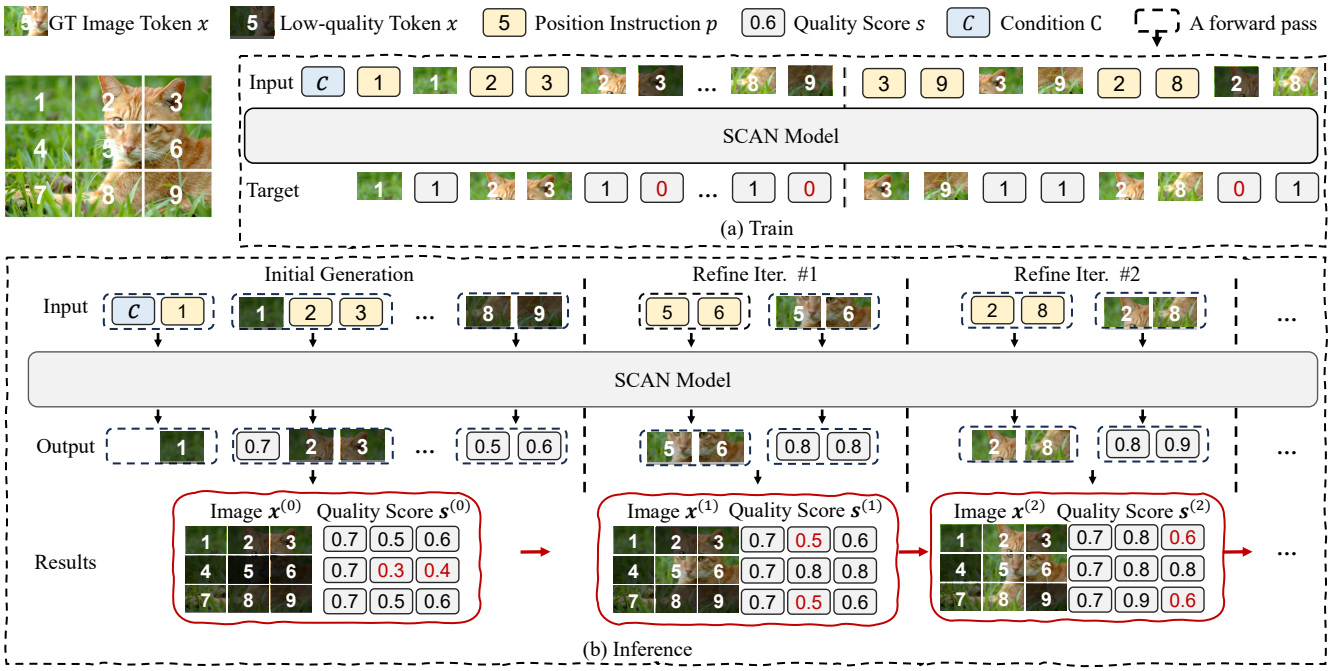

Figure 2: Overview of our method. The model predicts image tokens x_n given position instruction p_n, and predicts quality scores s_n given x_n. (a) The training input includes initial image and quality predictions, appended by sequence simulating refinement. (b) During inference, the model generates initial image $\boldsymbol{x}^{(0)}$ and quality scores $\boldsymbol{s}^{(0)}$, then iteratively refines.

MIM (Chang et al. 2022, 2023) Specifically, p_n for n-th image token at position (h, w) is:

$$p_n = \text{RoPE}(\boldsymbol{e}, h, w), \tag{8}$$

where a learnable embedding $\boldsymbol{e}$ is rotated according to 2D coordinates (h, w) following 2D-RoPE (Su et al. 2024; Pang et al. 2024).

Token quality evaluation. Each x_n sampled from the predicted categorical distribution is refed into the model to update the generated context. We leverage this forward pass to predict the quality score s_n, *i.e.*, x_n is input into the model G to predict s_n. Hence, $\text{Eva}(\cdot)$ is implemented as:

$$s_n = \text{G}(x_n \mid \boldsymbol{x}, \mathbf{C}). \tag{9}$$

Evaluating the quality of generated contents could be easier than generating them, similar to identifying flaws in an existing painting is simpler than painting from scratch.

Next set prediction. Utilizing position instruction tokens p_n as spatial indicators enables parallel generation, significantly reducing the number of iterations and accelerating inference. Image tokens $\boldsymbol{x} = \{x_n\}_{n=1:N}$ are partitioned into sets $\{x_1\}, \{x_2, x_3\}, ..., \{x_{N-2}, x_{N-1}, x_N\}$, where each set contains a predefined number of tokens and the number of sets is K. Similarly, quality scores are divided into corresponding sets $\{s\}$ for parallel generation. By default, we follow MAR (Li et al. 2024), increasing the number of tokens in each set with a sine schedule, and setting $K = 64$ for generation. Besides, the quality evaluation of previously generated image tokens and the generation of next iteration can be processed in parallel, forming the input sequence:

$$[\underbrace{\mathbf{C}, p_1}_{\text{Iter 1}}, \underbrace{x_1, p_2, p_3}_{\text{Iter 2}}, \underbrace{x_2, x_3, p_4, p_5}_{\text{Iter 3}}, \ldots, \underbrace{x_{N-2}, x_{N-1}, x_N}_{\text{Iter } K}]. \tag{10}$$

The corresponding output sequence can be represented as:

$$[\underbrace{x_1}_{\text{Iter 1}}, \underbrace{s_1, x_2, x_3}_{\text{Iter 2}}, \underbrace{s_2, s_3, x_4, x_5}_{\text{Iter 3}}, \ldots, \underbrace{s_{N-2}, s_{N-1}, s_N}_{\text{Iter } K}], \tag{11}$$

where the output of the condition $\mathbf{C}$ is omitted. It can be seen that each p_n generates x_n which is then evaluated to a quality score s_n. Bidirectional attention is used within set, and causal attention is used between sets. The causal attention between sets can be accelerated using KV-cache (Shazeer 2019) to speed up inference. Given an input set $[x_{n-1}, p_n, p_{n+1}]$ as an example, the model prediction can be represented as:

$$[s_{n-1}, x_n, x_{n+1}] = \text{G}(x_{n-1}, p_n, p_{n+1} \mid x_{<n-1}, \mathbf{C}). \tag{12}$$

Position-specified token generation and evaluation are achieved by a single decoder-only model without additional modules. The next set prediction accelerates this process, ensuring that even with refinements, our method maintains a clear speed advantage over recent works as shown in Tab. 1.

Model Training

The model is trained to perform both generation and evaluation within a single model G. We unify these tasks as a unified categorical prediction problem, with the training loss:

$$\mathcal{L} = \mathcal{L}_{gen} + \mathcal{L}_{eva} \tag{13}$$

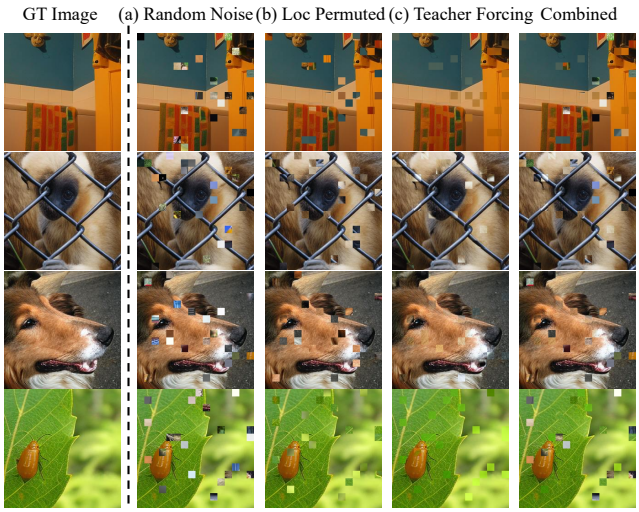

Figure 3: Three types of negative samples and combination.

where $\mathcal{L}_{gen}$ and $\mathcal{L}_{eva}$ represent the loss functions for generation and evaluation, respectively.

Generation training. The model G is trained to maximize the log-likelihood of the ground truth token sequence $\boldsymbol{x}$:

$$\mathcal{L}_{gen} = E[-\log \mathrm{p}(\boldsymbol{x}|\mathbf{C})], \tag{14}$$

where log-likelihood is computed as the cross-entropy loss between the true one-hot token and the predicted token.

Evaluation training. We formulate the quality score evaluation as a binary categorical prediction task, *i.e.*, distinguishing between positive (high-quality) and negative (low-quality) samples. Ground truth image tokens are defined as positive samples $\mathbf{x}^+$ with a quality score s of 1, while low-quality tokens $\mathbf{x}^-$ are assigned a score s of 0. Negative samples $\mathbf{x}^-$ are constructed from three types:

(a) Random noise content. Some input image tokens are randomly replaced with visual content from the VAE tokenizer discrete codes (Van Den Oord, Vinyals et al. 2017). Although the replaced content is valid in terms of the codebook, it does not align with the context of the image. This simulates errors caused by random sampling from the categorical distribution.

(b) Location permuted patches. Image tokens are spatially permuted, where each token is relocated to a different spatial position. These patches retain valid visual content but are spatially misaligned. This design enhances spatial sensitivity of the model and the ability to maintain the continuity of adjacent visual content.

(c) Teacher forcing prediction. The third type of negative samples comes from teacher forcing prediction errors, enabling the model to learn to identify and correct its own prediction mistakes. To make the predictions align with the ground truth visual content, we use teacher forcing prediction (Lamb et al. 2016) conditioned on the ground truth sequence. Specifically, we perform a forward pass, backward pass, and optimization conditioned on ground truth, following the standard AR training step (Sun et al. 2024). Currently, incorrectly predicted tokens are collected as negative samples for the next training step. It is worth noting that errors from teacher forcing predictions differ from those in real inference. In inference, errors arise from both model prediction and random sampling, leading to error accumulation. As a result, predictions from teacher forcing and real-world inference are drawn from different distributions, *i.e.*, the *data distribution* for teacher forcing and *model distribution* for real inference. This discrepancy, known as exposure bias (Ranzato et al. 2015), means we cannot rely solely on teacher forcing predictions to construct negative samples.

As visualized in Fig. 3, these types of negative samples represent different low-quality tokens, *i.e.*, random noise resembles generation errors, location permuted patches resembles spatial misalignment, and teacher forcing prediction resembles ambiguous or uncertain patches. Our negative samples $\mathbf{x}^-$ are constructed by randomly combining three types to ensure robust training and enhance the model to handle diverse real-world scenarios. Ablation studies in Tab. 3 validate this combination yields superior performance.

Training input $\boldsymbol{x}$ is constructed with a negative sample ratio r. Specifically, $\boldsymbol{m} \in \{0,1\}^N$ is a binary mask of the same shape as $\boldsymbol{x}$, composed of randomly assigned 0 or 1. r proportion values of $\boldsymbol{m}$ are set to 0 as negative, *i.e.*, $\sum_n^N m_n = N \times (1-r)$. The positive samples $\boldsymbol{x}^+$ and the negative samples $\boldsymbol{x}^-$ are combined using the mask $\boldsymbol{m}$ as:

$$\boldsymbol{x} = \boldsymbol{x}^+ \times \boldsymbol{m} + \boldsymbol{x}^- \times (1-\boldsymbol{m}). \tag{15}$$

Correspondingly, the quality scores $\boldsymbol{s}$ for tokens that match the ground truth are set to 1; otherwise, they are set to 0:

$$s_i = \begin{cases} 1, \text{ if } x_n = x_n^+ \\ 0, \text{ otherwise.} \end{cases} \tag{16}$$

The model is trained to predict these quality scores based on the context, which can be expressed as:

$$\mathcal{L}_{eva} = E[\sum_{n=1}^{N} -\log \mathrm{p}(s_n|x_{\leq n}, \mathbf{C})]. \tag{17}$$

During training, negative tokens requiring refinement are appended to the end of training sequence $\boldsymbol{x}$ to simulate iterative refinement. As shown in Fig. 2 (a), we append $[p_3, p_9, x_3, x_9, ...]$ to simulate the refinement of tokens $\{x_3, x_9\}$ and evaluation of newly generated tokens. These appended tokens are also set as negative samples with a probability r, simulating cases where refinement may still result in low-quality outputs.

Experiments

Implementation Details

For initial image generation, we use 64 steps to produce a 256×256 image with next set-of-tokens prediction, progressively increasing the number of tokens in each set from 1 to 4 using a sine schedule as in (Li et al. 2024; Chang et al. 2022; Li et al. 2023). By default, the refinement iteration T is set to 30, with $C = 4$ tokens refined at each iteration. The model is trained on the ImageNet-1K (Deng et al. 2009). During training, the negative sample ratio r is randomly sampled between 0 and 0.4 at each iteration. Detailed architectures and training hyper-parameters are provided in appendix.

Type	Model	#Para	#Step	#Refine	FID↓	IS↑	Pre↑	Rec↑	Latency (s)
Diffusion	ADM	554M	256	0	10.94	101	0.69	0.63	-
Diffusion	DiT-L/2	458M	250	0	5.02	167.2	0.75	0.57	285.82*
Diffusion	DiT-XL/2	675M	250	0	2.27	278.2	0.83	0.57	414.90*
AR	RAR-XL	955M	256	0	1.50	306.9	0.8	0.62	-
AR	RAR-XXL	1.4B	256	0	1.48	326	0.8	0.63	-
AR	RandAR-L	343M	88	0	2.55	288.8	0.81	0.58	-
AR	RandAR-XL	775M	88	0	2.25	317.8	0.80	0.60	6.6
AR	MAR	943M	64	0	1.55	303.7	0.81	0.62	53.30*
AR	LlamaGen-L	343M	256	0	3.80	248.3	0.83	0.51	13.72
AR	LlamaGen-XL	775M	256	0	3.39	227.1	0.81	0.54	19.76
AR	LlamaGen-XXL	1.4B	256	0	3.09	253.6	0.83	0.53	26.38
AR	VAR-d20	600M	10	0	2.57	302.6	0.83	0.56	4.61*
AR	VAR-d24	1.0B	10	0	2.09	312.9	0.82	0.59	5.53*
AR	SCAN-L†	343M	64	0	3.05	274.3	0.82	0.57	2.09
AR	SCAN-L	343M	64	30	2.51(**-17%**)	307.3(**+12%**)	0.81	0.60	2.47
AR	SCAN-XL†	775M	64	0	2.52	314.5	0.84	0.58	2.37
AR	SCAN-XL	775M	64	30	2.10(**-16%**)	326.1(**+4%**)	0.83	0.59	3.56
AR	SCAN-XXL†	1.4B	64	0	2.24	327.2	0.81	0.58	4.09
AR	SCAN-XXL	1.4B	64	30	1.95(**-13%**)	338.7(**+4%**)	0.83	0.61	5.62

Table 1: Model comparisons on class-conditional ImageNet 256×256. Symbols "↓" or"↑" indicate whether lower or higher values are better. "#Step": the number of model runs needed to generate an image. Wall-clock inference time is reported on one A100 GPU with batch size of 1. "*" indicates measured using official codes on our own. "†" denotes SCAN without refinement.

Refine Iters T	FID↓	IS↑	Times
0	2.52	314.51	2.37
10	2.39	323.54	2.61
20	2.16	324.26	3.06
30	2.10	326.13	3.56
40	2.04	326.55	4.07
50	2.03	329.68	4.53
60	2.03	329.71	5.11

Table 2: Impact of refinement iterations T.

Negative Sample	Initial Generation		With Refinement	
	FID↓	IS↑	FID↓	IS↑
Random Noise	3.27	260.80	2.81	283.2
Loc. Permuted	2.87	283.76	2.52	296.6
Teacher Forcing	2.49	307.46	2.31	320.8
Combined	2.52	314.51	2.10	326.1

Table 3: Impact of negative sample construction strategies.

Main Results

We evaluate SCAN models on the ImageNet 256×256 conditional generation benchmark, comparing them against SOTA generative models: diffusion models, such as ADM (Dhariwal and Nichol 2021) and DiT (Peebles and Xie 2023) and AR models, including RAR (Yu et al. 2024), RandAR (Pang et al. 2024), MAR (Li et al. 2024), LlamaGen (Sun et al. 2024), and VAR (Tian et al. 2024). The results are summarized in Tab. 1. We report model performance without and with refinement. After 30 refinement iterations, the three SCAN models improve FID scores by 0.54 (-17%), 0.42 (-16.7%), and 0.29 (-12.9%), respectively. With refinement, SCAN maintains a strong balance between speed and performance compared to recent methods. For example, SCAN-XL with refinement achieves a latency of 3.56s, significantly faster than 19.76s of LlamaGen-XL, 6.6s of RandAR-XL, and 4.61s of VAR-d20, while achieving a comparable FID score (2.10 for ours vs. 3.39, 2.25, and 2.09 for these methods, respectively). This efficiency is attributed to next set-of-tokens prediction and the generation-then-refine strategy. The wall-clock times in Tab. 1 empirically show that SCAN-XL accelerates inference by 22.8% over VAR-d20, 46.1% over RandAR-XL, 93.3% over MAR, and 76.9% over RAR-XL. Experimental results consistently show improvements across different model scales, demonstrating **high-quality generation** and **efficiency** of the SCAN.

Ablation Studies

Number of refinement iterations. Our refinement iteratively selects and refine tokens with the lowest quality scores for T iterations. Tab. 2 compares performance and speed across different numbers of refinement iterations T. As shown, refinement consistently improves FID and IS scores, though with saturating improvements as T increases. For example, the FID improvements from 10 to 60 iterations are 0.13, 0.23, 0.06, 0.06, and 0.01, respectively. It is worth noting that more refinement iterations do not always lead to better results. Since only tokens with improved quality scores are updated to prevent over-refinement, later iterations may fail to enhance token quality, rendering further refinements ineffective. Fig. 2 visualizes this effect, showing that generated images become static after a certain point. We set the number of refinement iterations to 30 as a good balance of performance and speed.

Negative sample construction. We construct negative samples using three strategies to simulate different types of low-quality tokens. Tab. 3 compares strategies include ran-

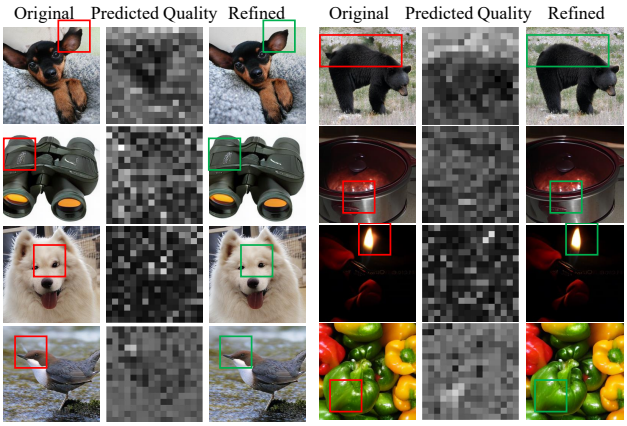

Figure 4: Predicted quality scores. We present the original generated images, the predicted quality scores, and the corresponding refined images. In predicted quality, brighter patches indicate lower quality and require refinement. Red and green boxes highlight the flawed and refined regions.

dom noise, location-permuted tokens, teacher forcing prediction, and their combination, with and without 30 refinement iterations. It can be seen that using only random noise or location-permuted tokens degrades the initial generation performance, as severe corruption of the input disrupts normal generation training. This results in relatively poor FID scores of 3.27 (+0.75) and 2.87 (+0.35), and IS scores of 260.80 (-53.71) and 283.76 (-30.75), respectively. Using only teacher forcing prediction preserves the quality of initial generation but falls short in learning refinement capabilities. This is because the errors generated by teacher forcing are relatively homogeneous, unable fully simulate the diverse types of errors encountered in real-world inference. Combining all three strategies significantly improves both initial generation and refinement performance. This ensures the model can effectively learn to correct real-world errors, leading to robust and high-quality generation.

Visualization

Predicted quality scores. Fig. 4 visualizes the predicted quality scores for generated images, where brighter areas indicate lower quality. The model identifies low-quality tokens, such as blurry, inconsistent, or structurally flawed regions. For example, corresponding to these three types of problems, in the first row (Chihuahua), the blurry ear region is detected; in the second row (telescope and pot), discontinuous lines are corrected; and in the third row (flame), anomalous flame is identified. Additionally, the model fixes color problem, such as the yellowish tint of the green pepper in the fourth row, restoring it to correct green color. By addressing low-quality areas in both objects and backgrounds, the model noticeably improves overall visual quality.

Refined with more iterations. Fig. 5 visualizes the effects of refining images for up to 50 iterations. Early refinement iterations improve low-quality regions, such as the roof of the yurt in the first example and the grassland behind the dog. Using a greedy strategy, only tokens with improved

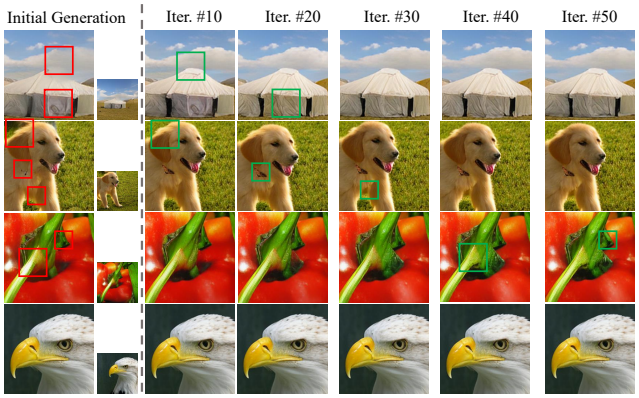

Figure 5: Visualization of additional refinement iterations, with central regions zoomed in. Red and green boxes highlight flaws and refinements. Early iterations effectively correct flaws, and further refinements do not cause over-refinement. The last two cases show two extremes: one refined up to 50 iters, and another achieving ideal quality initially, performing no refinement.

quality scores after regeneration are updated at each step, ensuring the image quality converges and preventing over-refinement. As iterations increase, gains gradually saturate, and further refinements no longer improve the image. In rare cases, like the red pepper stem example, some uncertainty remains, and the image hasn't fully converged even after 50 iterations. The last row shows another rare case where the initial eagle image is already high quality, and no refinement is performed. A meaningful future investigation would be to explore different strategies for selecting regions to refine, such as incorporating randomness or connectivity, to further enhance refinement efficiency and effectiveness.

Conclusion

We introduce Self-Calibrated AutoregressioN (SCAN), a novel autoregressive model capable of predicting token quality and adaptively refining generated content. SCAN integrates two key abilities: (1) generating image tokens at specific locations and (2) evaluating the quality scores. We formulate quality prediction as a binary classification task, and construct low-quality negative samples from three sources: random noise content, location-permuted patches, and teacher forcing prediction, without the need for additional modules or labeled data. During inference, the model first generates a coarse initial image and its quality scores, then iteratively refines low-quality patches, until satisfactory quality is achieved. This approach improves the baseline by 38% in FID and achieves a 5.6× speedup, while outperforming recent works in both performance and speed. We hope SCAN inspires further exploration of token quality evaluation and refinement in AR models.

Acknowledgments

This work is supported in part by Grant No. 2023-JCJQ-LA-001-088, in part by Grant No. 2025ZD1601300, in part by Natural Science Foundation of China under Grant No.

U20B2052, 61936011, in part by the Okawa Foundation Research Award, in part by Ant Group Research Intern Program, in part by the Ant Group Research Fund, and in part by the Kunpeng&Ascend Center of Excellence, Peking University.

References

Bai, Y.; Jones, A.; Ndousse, K.; Askell, A.; Chen, A.; DasSarma, N.; Drain, D.; Fort, S.; Ganguli, D.; Henighan, T.; et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Chang, H.; Zhang, H.; Barber, J.; Maschinot, A.; Lezama, J.; Jiang, L.; Yang, M.-H.; Murphy, K.; Freeman, W. T.; Rubinstein, M.; et al. 2023. Muse: Text-to-image generation via masked generative transformers. *arXiv preprint arXiv:2301.00704*.

Chang, H.; Zhang, H.; Jiang, L.; Liu, C.; and Freeman, W. T. 2022. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11315–11325.

Chen, J.; Hu, J.; Wang, G.; Jiang, Z.; Zhou, T.; Chen, Z.; and Lv, C. 2025. TaoAvatar: Real-Time Lifelike Full-Body Talking Avatars for Augmented Reality via 3D Gaussian Splatting. In *CVPR*.

Chen, X.; Chen, Z.; and Song, Y. 2025. Transanimate: Taming layer diffusion to generate rgba video. *arXiv preprint arXiv:2503.17934*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. Ieee.

Devlin, J. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34: 8780–8794.

Esser, P.; Rombach, R.; and Ommer, B. 2021. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12873–12883.

Feng, F.; Xie, Y.; Yang, X.; Wang, J.; and Geng, X. 2025a. Redefining¡ creative¿ in dictionary: Towards an enhanced semantic understanding of creative generation. In *CVPR*.

Feng, Z.; Guo, Q.; Xiao, X.; Xu, R.; Yang, M.; and Zhang, S. 2025b. Unified Video Generation via Next-Set Prediction in Continuous Domain. In *ICCV*, 19427–19438.

Feng, Z.; and Zhang, S. 2023a. Efficient vision transformer via token merger. *IEEE Transactions on Image Processing*, 32: 4156–4169.

Feng, Z.; and Zhang, S. 2023b. Evolved part masking for self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10386–10395.

Feng, Z.; and Zhang, S. 2024. Evolved Hierarchical Masking for Self-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Fu, J.; Ng, S.-K.; Jiang, Z.; and Liu, P. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Hong, W.; Ding, M.; Zheng, W.; Liu, X.; and Tang, J. 2022. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*.

Hua, L.; Liu, F.; Su, J.; Miao, X.; Ouyang, Z.; Wang, Z.; Hu, R.; Wen, Z.; Zhai, B.; Long, Y.; et al. 2025. Attention in diffusion model: A survey. *arXiv preprint arXiv:2504.03738*.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1125–1134.

Jiang, Z.; Xu, J.; Zhang, S.; Shen, T.; Li, J.; Kuang, K.; Cai, H.; and Wu, F. 2025. FedCFA: Alleviating Simpson's Paradox in Model Aggregation with Counterfactual Federated Learning. In *AAAI*, 17662–17670.

Kang, M.; Zhu, J.-Y.; Zhang, R.; Park, J.; Shechtman, E.; Paris, S.; and Park, T. 2023. Scaling up gans for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10124–10134.

Kondratyuk, D.; Yu, L.; Gu, X.; Lezama, J.; Huang, J.; Schindler, G.; Hornung, R.; Birodkar, V.; Yan, J.; Chiu, M.-C.; et al. 2023. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*.

Lamb, A. M.; ALIAS PARTH GOYAL, A. G.; Zhang, Y.; Zhang, S.; Courville, A. C.; and Bengio, Y. 2016. Professor forcing: A new algorithm for training recurrent networks. *Advances in neural information processing systems*, 29.

Li, K.; Jiang, Z.; Shen, Z.; Wang, Z.; Lv, C.; Zhang, S.; Wu, F.; and Wu, F. 2025. MadaKV: Adaptive Modality-Perception KV Cache Eviction for Efficient Multimodal Long-Context Inference. In *ACL*.

Li, T.; Chang, H.; Mishra, S.; Zhang, H.; Katabi, D.; and Krishnan, D. 2023. Mage: Masked generative encoder to unify representation learning and image synthesis. In *CVPR*, 2142–2152.

Li, T.; Tian, Y.; Li, H.; Deng, M.; and He, K. 2024. Autoregressive Image Generation without Vector Quantization. *arXiv preprint arXiv:2406.11838*.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2024. Self-refine: Iterative refinement with self-feedback. *NIPS*, 36.

Pang, Z.; Zhang, T.; Luan, F.; Man, Y.; Tan, H.; Zhang, K.; Freeman, W. T.; and Wang, Y.-X. 2024. RandAR: Decoder-only Autoregressive Visual Generation in Random Orders. *arXiv preprint arXiv:2412.01827*.

Peebles, W.; and Xie, S. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4195–4205.

Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; and Sutskever, I. 2021. Zero-shot text-to-image generation. In *ICML*, 8821–8831. Pmlr.

Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *CVPR*, 10684–10695.

Sauer, A.; Schwarz, K.; and Geiger, A. 2022. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, 1–10.

Schick, T.; Dwivedi-Yu, J.; Jiang, Z.; Petroni, F.; Lewis, P.; Izacard, G.; You, Q.; Nalmpantis, C.; Grave, E.; and Riedel, S. 2022. Peer: A collaborative language model. *arXiv preprint arXiv:2208.11663*.

Shazeer, N. 2019. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*.

Song, Y.; Huang, S.; Yao, C.; Ye, X.; Ci, H.; Liu, J.; Zhang, Y.; and Shou, M. Z. 2024. Processpainter: Learn painting process from sequence data. *arXiv preprint arXiv:2406.06062*.

Song, Y.; Liu, C.; and Shou, M. Z. 2025. MakeAnything: Harnessing Diffusion Transformers for Multi-Domain Procedural Sequence Generation. *arXiv preprint arXiv:2502.01572*.

Song, Y.; Liu, X.; and Shou, M. Z. 2024. Diffsim: Taming diffusion models for evaluating visual similarity. *arXiv preprint arXiv:2412.14580*.

Su, J.; Ahmed, M.; Lu, Y.; Pan, S.; Bo, W.; and Liu, Y. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568: 127063.

Sun, P.; Jiang, Y.; Chen, S.; Zhang, S.; Peng, B.; Luo, P.; and Yuan, Z. 2024. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*.

Sun, Z.; Guo, K.; Hu, Y.; Tian, D.; Gao, Q.; Wang, J.; Gao, J.; Sun, Y.; and Yin, B. 2025. Large-Small Model Synergy with Multimodal Fine-Grained Heuristics for Knowledge-Based Visual Question Answering. In *The 33rd ACM MM*, 935–944.

Sun, Z.; Hu, Y.; Gao, Q.; Jiang, H.; Gao, J.; Sun, Y.; and Yin, B. 2023. Breaking the barrier between pre-training and fine-tuning: A hybrid prompting model for knowledge-based vqa. In *ACM MM*.

Tian, K.; Jiang, Y.; Yuan, Z.; Peng, B.; and Wang, L. 2024. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *arXiv preprint arXiv:2404.02905*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Van Den Oord, A.; Vinyals, O.; et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Wan, C.; Luo, X.; Cai, Z.; Song, Y.; Zhao, Y.; Bai, Y.; He, Y.; and Gong, Y. 2024. Grid: Visual layout generation. *arXiv preprint arXiv:2412.10718*.

Wang, X.; Zhang, X.; Luo, Z.; Sun, Q.; Cui, Y.; Wang, J.; Zhang, F.; Wang, Y.; Li, Z.; Yu, Q.; et al. 2024. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*.

Wu, T.; Lan, J.; Yuan, W.; Jiao, J.; Weston, J.; and Sukhbaatar, S. 2024a. Thinking llms: General instruction following with thought generation. *arXiv preprint arXiv:2410.10630*.

Wu, Y.; Zhang, Z.; Chen, J.; Tang, H.; Li, D.; Fang, Y.; Zhu, L.; Xie, E.; Yin, H.; Yi, L.; et al. 2024b. Vila-u: a unified foundation model integrating visual understanding and generation. *arXiv preprint arXiv:2409.04429*.

Xiao, J.; Chen, Y.; Feng, X.; Wang, R.; and Wu, Z. 2025. RecNet: Optimization for Dense Object Detection in Retail Scenarios Based on View Rectification. In *ICASSP 2025*, 1–5.

Xie, Y.; Feng, F.; Shi, R.; Wang, J.; Rui, Y.; and Geng, X. 2025. Kind: Knowledge integration and diversion for training decomposable models. In *ICML*.

Yasunaga, M.; and Liang, P. 2020. Graph-based, self-supervised program repair from diagnostic feedback. In *ICML*, 10799–10808. PMLR.

Yu, J.; Xu, Y.; Koh, J. Y.; Luong, T.; Baid, G.; Wang, Z.; Vasudevan, V.; Ku, A.; Yang, Y.; Ayan, B. K.; et al. 2022. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3): 5.

Yu, L.; Cheng, Y.; Sohn, K.; Lezama, J.; Zhang, H.; Chang, H.; Hauptmann, A. G.; Yang, M.-H.; Hao, Y.; Essa, I.; et al. 2023a. Magvit: Masked generative video transformer. In *CVPR*, 10459–10469.

Yu, L.; Lezama, J.; Gundavarapu, N. B.; Versari, L.; Sohn, K.; Minnen, D.; Cheng, Y.; Birodkar, V.; Gupta, A.; Gu, X.; et al. 2023b. Language Model Beats Diffusion–Tokenizer is Key to Visual Generation. *arXiv preprint arXiv:2310.05737*.

Yu, Q.; He, J.; Deng, X.; Shen, X.; and Chen, L.-C. 2024. Randomized autoregressive visual generation. *arXiv preprint arXiv:2411.00776*.

Zhou, Y.; Jin, S.; Hua, L.; Lv, W.; Duan, H.; and Han, J. 2025. ConsDreamer: Advancing Multi-View Consistency for Zero-Shot Text-to-3D Generation. *arXiv preprint arXiv:2504.02316*.

Zhu, D.; Chen, J.; Shen, X.; Li, X.; and Elhoseiny, M. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.