Pixel-space Autoregressive Image Synthesis via Spectrum Serialization and Flow-based Refinement

Guiwei Zhang[1] **Tianyu Zhang**[2] **Yalong Bai**[3] **Ying Ba**[4] **Zichang Tan**[5] **Yang Yang**[6*]

[1]Du Xiaoman Technology [2]Tomorrow Advancing Life (TAL) [3]Canva Research [4]Renmin University of China
[5]Sangfor Technologies Inc. [6]Institute of Automation, Chinese Academy of Sciences

Abstract

Autoregressive (AR) image generators commonly rely on vector-quantized (VQ) autoencoders to compress images into discrete token sequences. However, quantization can discard visual information, and reconstruction fidelity alone does not ensure that the resulting sequences are easy to model. We revisit the representation choice for AR image generation, seeking a simpler representation that supports both quantization and sequence modeling. Our analyses identify pixel blocks as a representation with low estimated intrinsic dimension and favorable quantization properties, while their spectrum-serialized tokens exhibit greater AR predictability in controlled comparisons. Motivated by these complementary observations, we propose **PixeLLM**, an AR generative framework that directly quantizes spectrum-transformed pixel blocks, eliminating the need for a separately trained image autoencoder. PixeLLM decomposes generation into two stages: an AR model generates a text-aligned draft capturing global scene structure, and a conditional flow-based pixel refinement Transformer restores fine-grained details and mitigates artifacts. This division of labor couples discrete structure prediction with continuous detail synthesis. PixeLLM achieves competitive performance on challenging text-conditioned and class-conditioned image generation benchmarks. Code repository: `https://github.com/zhangguiwei610/PixeLLM`.

1 Introduction

Autoregressive (AR) models offer a scalable approach to image synthesis by representing images as token sequences and predicting them with architectures developed for language modeling. Recent text-to-image systems pair a large language model (LLM) with a learned visual tokenizer, allowing language understanding and sequence modeling to support conditional image generation [46, 12, 16, 47]. The visual representation plays a central role in this pipeline: it must preserve information needed for reconstruction while exposing token dependencies that the AR model can predict effectively.

These requirements are related but not equivalent. Quantization introduces reconstruction error, yet improving reconstruction alone does not necessarily make a token sequence easier to model. GigaTok [55], for example, shows that scaling a visual tokenizer can improve reconstruction while degrading downstream AR generation, and motivates evaluating tokenizers through AR probing. This distinction suggests that visual representations should be assessed through both their reconstruction pathways and the conditional token distributions presented to the sequence model.

We revisit whether AR image synthesis can operate closer to pixels without a learned image autoencoder. Our analysis provides two complementary empirical findings (Section 2). Under a common normalized-space protocol, patchified RGB exhibits the lowest estimated intrinsic dimension and quantization distortion among the four evaluated representations (Figure 2), motivating

*Corresponding author: `yang.yang@nlpr.ia.ac.cn`.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

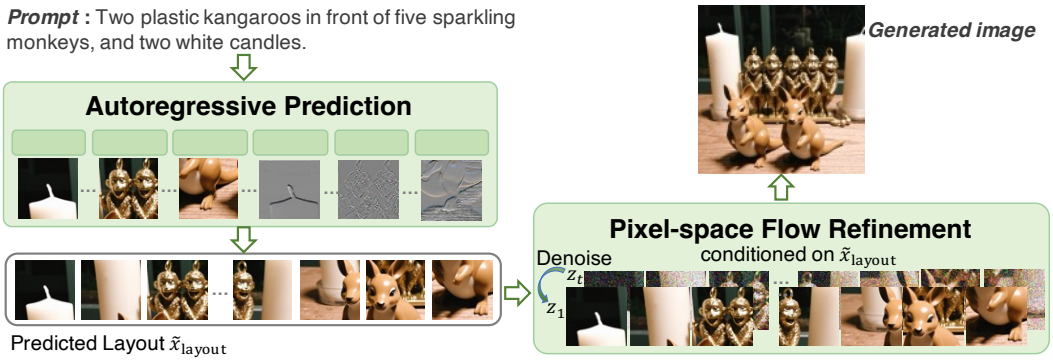


Figure 1: **Highlights of PixeLLM.** A text-conditioned autoregressive model first predicts spectrum-based pixel-block tokens that define a visual draft. A conditional flow-based pixel Transformer then refines the draft with fine-scale details while preserving its global structure. The pipeline does not require a separately pretrained image autoencoder.

direct pixel-block quantization. Controlled AR comparisons further show that spectrum-serialized pixel-block tokens are more predictable than learned IBQ latent tokens [42] (Figure 3). This finding provides direct evidence that quantization in pixel space can produce discrete sequences well suited to autoregressive modeling (Section 2.2).

Together, these quantization and predictability findings motivate **PixeLLM**, an autoencoder-free generator that directly quantizes spectrum-transformed RGB blocks and models their discrete dependencies in a coarse-to-fine order. A two-dimensional discrete wavelet transform (DWT) [19] separates low-frequency structure from horizontal, vertical, and diagonal details. Frequency-specific codebooks represent these components, and coarse-to-fine serialization presents them to the causal AR model in an order that exposes their dependencies. This factorization provides a structured alternative to a single raster-ordered codebook while retaining a fixed token budget; its representational properties are discussed in Appendix D.

PixeLLM assigns complementary roles to structure prediction and detail synthesis (Figure 1). The AR model predicts a text-aligned visual draft, which is converted back to an image through inverse DWT. A conditional flow-based pixel Transformer then adds fine-scale detail and reduces local artifacts directly in pixel space. The two stages use an autoregressive maximum-likelihood objective and a flow-matching objective, respectively, without requiring separate image-autoencoder pretraining.

Our experiments evaluate geometric structure, controlled AR predictability, downstream generation, and inference efficiency as distinct aspects of the design. PixeLLM achieves competitive results on GenEval [17], DPG-Bench [20], and GenEval2 [22] at 384×384 resolution, and on class-conditioned ImageNet generation [10] at 256×256 resolution. Together, these analyses support direct pixel-block tokenization, spectrum factorization, and explicit draft–refinement decomposition as an effective design for AR image synthesis.

2 Manifold complexity analysis

We distinguish the geometry of a representation from the predictability of the discrete sequences constructed from it. Intrinsic-dimension and clustering analyses characterize the former; a controlled AR experiment evaluates the latter after quantization and serialization.

2.1 Geometric analysis of representation spaces

A set of representation vectors in an ambient space $\mathbb{R}^N$ may lie near a manifold with intrinsic dimension $d \ll N$. This dimension describes the effective local degrees of freedom of the representation [38]. We estimate it using the k-nearest-neighbor maximum-likelihood estimator of Levina and Bickel [26]. Let $T_j(x)$ denote the Euclidean distance from a sample x to its j-th nearest neighbor.

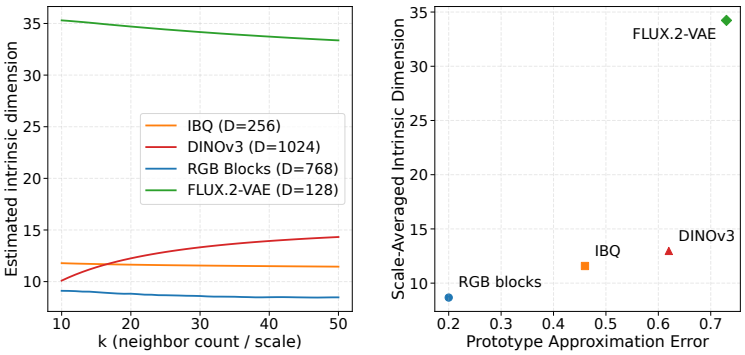


Figure 2: **Geometric analysis.** Intrinsic dimension estimated by kNN-MLE across neighborhood sizes k, and RMSE from quantization with 16,384 centroids. All four feature spaces are L2-normalized. Pixel blocks yield the lowest estimates under this protocol. These results describe geometry; Figure 3 tests AR predictability directly.

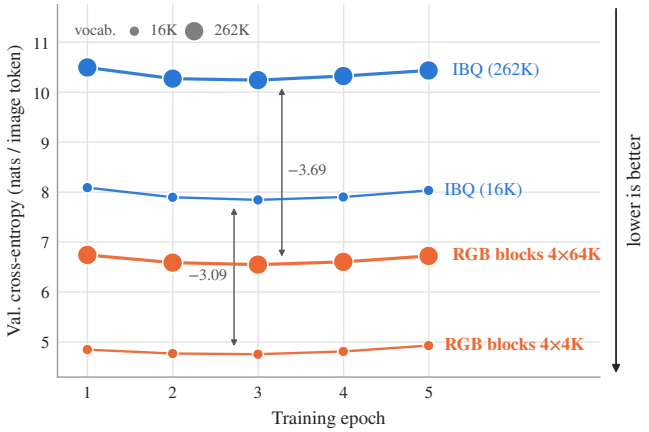


Figure 3: **AR validation loss.** Qwen3.5-2B [40] is trained on ImageNet for five epochs and tested on val-50k (Eq. (3)). All runs use 256×256 images, 256 visual tokens, and class conditioning. Small and large markers denote 2^{14} and 2^{18} classes, respectively. "RGB blocks" uses spectrum order and four frequency-specific codebooks (Section 3.1). Lower is better.

The local estimate is

$$\hat{m}_k(x) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{T_k(x)}{T_j(x)} \right]^{-1}. \tag{1}$$

Pooling the log-distance ratios over n samples gives

$$\bar{m}_k = \left[\frac{1}{n(k-1)} \sum_{i=1}^{n} \sum_{j=1}^{k-1} \log \frac{T_k(x_i)}{T_j(x_i)} \right]^{-1}. \tag{2}$$

Representations and protocol. We analyze the 50k ImageNet validation images [10] in four spaces: IBQ VQ-VAE representations [42], KL-regularized VAE latents [24], DINOv3 features [43], and 16×16 RGB blocks from 256×256 images. All feature vectors are L2-normalized to control for differences in magnitude. Across the tested neighborhood sizes k, patchified RGB consistently exhibits the lowest estimated intrinsic dimension (Figure 2), indicating fewer effective local degrees of freedom among the evaluated representations.

Feature-space quantization distortion. We complement the geometric analysis by fitting 16,384 k-means centroids separately in each normalized representation space using features from the same images. We report the root mean squared Euclidean distance from the feature vectors to their nearest centroids. L2 normalization removes differences in vector magnitude, placing this approximation error on a common unit-norm scale. Patchified RGB yields the lowest distortion at the same codebook budget (Figure 2), indicating that its normalized features are more accurately approximated by the codebook. Together, these geometric and quantization findings motivate direct pixel-block tokenization, whose AR sequence predictability we evaluate next.

2.2 Controlled evaluation of AR predictability

To test sequence predictability directly, we adopt the AR-probing principle of GigaTok [55]: evaluate each tokenization through the held-out next-token prediction loss of a common AR learner. This

measures the predictability of the discrete sequences actually presented to the model, complementing the geometric diagnostics above.

Evaluation metric. For representation r and total codebook size K, we train a separate class-conditional AR model $p_{\theta_{r,K}}$ and evaluate its teacher-forced mean visual-token cross-entropy:

$$\mathcal{L}_{\text{val}}^{(r,K)} = -\frac{1}{N_{\text{val}} L} \sum_{i=1}^{N_{\text{val}}} \sum_{t=1}^{L} \log p_{\theta_{r,K}}\left(z_{i,t}^{(r,K)} \mid z_{i,<t}^{(r,K)}, c_i\right). \tag{3}$$

Here, $z_{i,t}^{(r,K)}$ is the t-th quantized visual token of image i, c_i is its class label, $L = 256$, and $N_{\text{val}} = 50{,}000$. We use natural logarithms and report nats per image token. Only visual-token targets contribute to the loss; their predictions are conditioned on ground-truth prefixes and the class label. The AR models are trained exclusively on the ImageNet training split and evaluated on val-50k after each epoch.

Matched protocol at two codebook sizes. We compare learned IBQ tokens [42] with our directly quantized, spectrum-serialized pixel blocks (Section 3.1). Both use 256×256 images, 256 visual tokens per image, class conditioning, and the same Qwen3.5-2B backbone [40] trained for five epochs. We test $K \in \{16{,}384, 262{,}144\}$. IBQ uses a single K-entry codebook; our representation distributes the same total number of entries across four frequency-specific codebooks of $K/4$ entries. At each budget, both AR models predict over the full K-class visual vocabulary, without restricting spectrum-token predictions to the active frequency band. Thus, the comparison matches both sequence length and prediction-vocabulary size within each budget.

Results and interpretation. Direct pixel-space VQ of spectrum-transformed RGB blocks consistently yields lower AR validation cross-entropy than learned IBQ tokenization (Figure 3). This advantage holds at both codebook sizes and across all five training epochs; at epoch three, the reductions are 3.09 and 3.69 nats per visual token for $K = 16{,}384$ and $K = 262{,}144$, respectively. Under the matched Qwen3.5-2B protocol, these results provide direct, method-specific evidence that our spectrum-serialized pixel-block tokens form more predictable sequences than IBQ latent tokens and are more amenable to autoregressive modeling.

Implications for PixeLLM. Sections 2.1 and 2.2 provide complementary support for direct pixel-block tokenization. Patchified RGB exhibits the lowest normalized-space quantization distortion among the evaluated representations under the common clustering protocol, while spectrum-serialized pixel blocks yield more predictable token sequences than learned IBQ latents under matched AR training. Together, these findings motivate PixeLLM, an autoencoder-free generator that directly quantizes spectrum-transformed RGB blocks and models the resulting tokens in a coarse-to-fine order. The AR model predicts a text-aligned structural draft, and a conditional pixel-space flow refiner restores fine details.

3 PixeLLM

PixeLLM combines an AR spectrum-token generator with a conditional pixel-space refiner (Figure 4). The representation uses a fixed wavelet transform and learned codebooks, without a learned image encoder or autoencoder decoder. The AR stage predicts coarse image structure; the refiner synthesizes details conditioned on this prediction and the prompt.

3.1 Image-spectrum serialization

We use a single-level two-dimensional Haar discrete wavelet transform (DWT) to separate an RGB image x into a low-frequency approximation and three detail subbands:

$$\left(x^{\text{low}}, x^{H}, x^{V}, x^{D}\right) = \text{DWT}_1(x). \tag{4}$$

Each subband has half the input height and width. We serialize low-frequency tokens first, followed by spatially aligned horizontal, vertical, and diagonal detail tokens. This ordering makes the predicted coarse structure available when the causal AR model generates details.

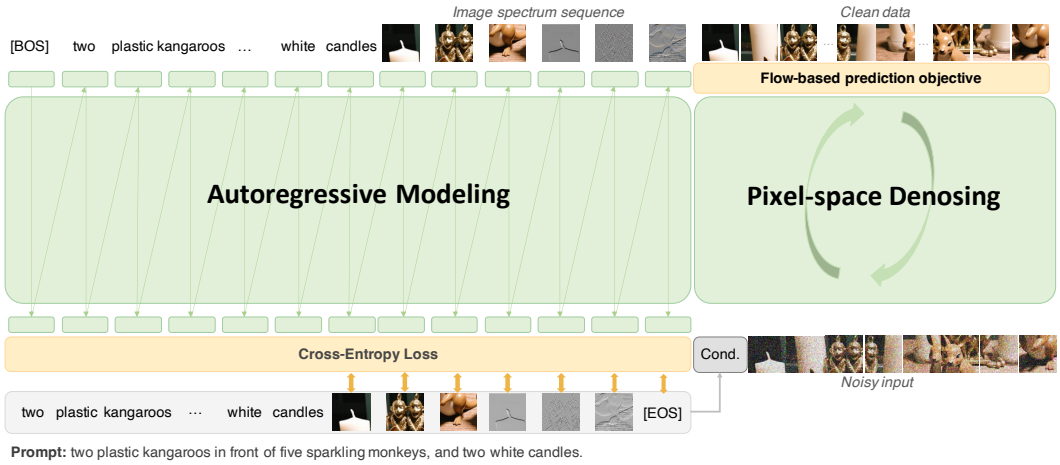


Figure 4: **Overview of PixelLLM.** The AR model predicts spectrum tokens conditioned on text and is trained by next-token maximum likelihood. A conditional pixel-space Transformer uses the predicted spectrum representation and text to synthesize details; it is trained with a flow-matching objective.

Patchification and quantization. We partition each subband into non-overlapping $P \times P$ blocks, with $P = 16$. Low-frequency patch vectors z_n^{low} are quantized with a dedicated codebook $\mathcal{E}^{\text{low}}$. For high-frequency content, we concatenate $[x^H; x^V; x^D]$ along channels and use group-wise vector quantization [21] with $G = 3$ groups and independent codebooks $\mathcal{E}_g^{\text{high}}$. Let q_n^{low} and $q_{m,g}^{\text{high}}$ denote the corresponding quantized vectors. The VQ objective is

$$\begin{aligned} \ell_{\text{VQ}}(z, q) &= \|\text{sg}[z] - q\|_2^2 + \beta\|z - \text{sg}[q]\|_2^2, \\ \mathcal{L}_{\text{VQ}} &= \sum_{n=1}^{N_{\text{low}}} \ell_{\text{VQ}}(z_n^{\text{low}}, q_n^{\text{low}}) + \sum_{m=1}^{N_{\text{high}}} \sum_{g=1}^{G} \ell_{\text{VQ}}(z_{m,g}^{\text{high}}, q_{m,g}^{\text{high}}), \end{aligned} \tag{5}$$

where sg denotes stop-gradient and β weights the commitment term. Codebooks are learned together with the initial refiner, as detailed in Appendix F.

We serialize the code indices as

$$\begin{aligned} s = \big(&\texttt{<L>}, l_1, \ldots, l_{N_{\text{low}}}, \texttt{<L>}, \\ &\texttt{<H>}, h_{1,1}, \ldots, h_{1,G}, h_{2,1}, \ldots, h_{N_{\text{high}},G}, \texttt{<H>}\big), \end{aligned} \tag{6}$$

where l_n and $h_{m,g}$ index the low- and high-frequency codebooks. Within each subband, spatial positions follow raster order. For an $H \times W$ image, the sequence contains $N_{\text{low}} + 3N_{\text{high}} = HW/P^2$ visual tokens, excluding delimiters: 256 at 256×256 and 576 at 384×384. The main configuration uses four codebooks of 65,536 entries each. The AR-probing study in Section 2.2 evaluates this 262,144-entry budget and a smaller budget of four 4,096-entry codebooks.

3.2 Flow-based pixel refinement

Flow-matching formulation. Given a clean image x and Gaussian noise ϵ, we use the linear path

$$z_t = tx + (1 - t)\epsilon, \qquad t \in [0, 1], \tag{7}$$

whose target velocity is $v = x - \epsilon$. Training samples t from a logit-normal distribution, with parameters given in Appendix F. The velocity-regression objective is

$$\mathcal{L}_{\text{flow}}(\theta) = \mathbb{E}_{t,x,\epsilon}\big[\|v_\theta(z_t, t) - v\|_2^2\big]. \tag{8}$$

Sampling integrates the learned velocity field from noise at $t = 0$ to an image at $t = 1$. The controlled latency comparison uses 50-step Euler integration (Section 4.5).

Conditioning and clean-image prediction. The predicted sequence $\hat{s}$ defines an RGB draft through codebook lookup and inverse DWT. For conditioning, we embed the spectrum tokens and combine them with prompt embeddings in a shallow Transformer:

$$c = \mathsf{CondEnc}\big([\mathsf{Embed}(\hat{s}); \mathsf{Text}(p)]\big). \tag{9}$$

These features condition a JiT-style pixel Transformer [28]; Appendix E.2 describes the architecture. The network predicts a clean-image estimate $x_\theta(z_t, t, c)$, which induces

$$\begin{aligned} \epsilon_\theta(z_t, t, c) &= \frac{z_t - t x_\theta(z_t, t, c)}{1 - t}, \\ v_\theta(z_t, t, c) &= x_\theta(z_t, t, c) - \epsilon_\theta(z_t, t, c) = \frac{x_\theta(z_t, t, c) - z_t}{1 - t}. \end{aligned} \tag{10}$$

For $t \in (0, 1)$, substituting this parameterization into the velocity loss gives

$$\mathcal{L}_{\text{refine}}(\theta) = \mathbb{E}_{t,x,\epsilon}\left[\frac{\|x_\theta(z_t, t, c) - x\|_2^2}{(1-t)^2}\right]. \tag{11}$$

At inference, classifier-free guidance combines the conditioned and condition-dropped predictions:

$$\tilde{x}_\theta(z_t, t, c) = \omega x_\theta(z_t, t, c) + (1 - \omega) x_\theta(z_t, t, c_\varnothing), \tag{12}$$

with scale $\omega > 1$. Algorithm 1 specifies the three training stages: joint codebook/refiner pretraining, AR training, and refiner adaptation to AR-predicted layouts.

4 Experiments

4.1 Experimental setup

We evaluate PixeLLM in both text-conditioned and class-conditioned image generation settings. For text-conditioned generation, we compare with representative baselines on GenEval [17], DPG-Bench [20], and GenEval2 [22] at 384×384 resolution. For class-conditioned generation, we evaluate on ImageNet val-50k [10] at 256×256 resolution. Implementation details are provided in the Appendix E.

4.2 Text-conditioned image generation

Quantitative results. Table 1 reports results on GenEval, DPG-Bench, and GenEval2. PixeLLM obtains overall scores of 0.82 and 84.41 on GenEval and DPG-Bench, respectively, and Soft-TIFA AM/GM scores of 71.74/30.90 on GenEval2, the highest values among the listed comparisons. Soft-TIFA AM summarizes atomic concept accuracy, whereas GM penalizes poorly satisfied concepts within a prompt [22]. The latter result is consistent with improved compositional prompt adherence. These benchmark comparisons span different pretrained systems and do not isolate architecture or training-data effects.

Qualitative results. Figure 5 compares PixeLLM with Janus-Pro-7B [8], Tar-7B [18], and Z-Image [2]. The selected examples illustrate attribute binding, spatial relations, and local detail. PixeLLM generates at 384×384; higher-resolution baseline outputs are resized for display as specified in the caption. These examples complement the benchmark scores rather than constitute a controlled perceptual study.

4.3 Class-conditioned image generation

Quantitative results. On ImageNet at 256×256, PixeLLM achieves a gFID of 1.64 after 200 epochs of refiner training (Table 2). This is lower than the listed pixel-space baselines and competitive with the latent-space methods. The comparison uses each method's reported configuration, with different training budgets. It demonstrates the viability of the complete pixel-space pipeline, rather than isolating the contribution of either stage.

Table 1: **Text-to-image benchmark results.** PixeLLM is evaluated at 384×384. Comparisons use the listed systems' benchmark scores and may differ in training data and sampling configuration. GenEval and DPG-Bench report overall scores; GenEval2 reports Soft-TIFA arithmetic and geometric means (AM/GM). Higher is better. Category-level results appear in Appendix G.1.

(a) GenEval.

Model	Overall↑
SD3-Medium [14]	0.74
MaskGen-XL [23]	0.57
DC-AR [53]	0.69
LlamaGen [45]	0.32
TokenFlow [39]	0.55
Emu3-Gen [51]	0.66
Show-o [54]	0.53
Janus-Pro [8]	0.80
OmniGen2 [52]	0.80
PixeLLM	**0.82**

(b) DPG-Bench.

Model	Overall↑
DALL-E 3 [1]	83.50
SD3-Medium [14]	84.08
Emu3-Gen [51]	80.60
UniWorld-V1 [31]	81.38
TokenFlow [39]	73.38
BLIP3-o 8B [5]	81.60
OmniGen2 [52]	83.57
Janus-Pro [8]	84.19
PixeLLM	**84.41**

(c) GenEval2.

Model	AM↑	GM↑
SDXL† [37]	50.1	9.1
SD3 [14]	66.3	21.3
SD3.5-L [14]	69.7	22.8
Flux [24]	67.1	21.1
BAGEL+CoT† [9]	70.9	23.1
Janus-Pro [8]	54.92	13.1
Tar-7B [18]	48.10	10.16
PixeLLM	**71.74**	**30.90**

Table 2: **Class-conditioned generation on ImageNet at** 256×256**.** Training epochs and gFID are reported for each listed configuration; training budgets differ. PixeLLM uses 10 AR epochs and 200 refiner epochs. RAE denotes a representation autoencoder.

Method	Epochs	Representation	Generator	gFID↓
FAR-L [56]	400	KL-VAE	AR	3.45
MAR-H [30]	800	KL-VAE	Masked AR	1.55
VAR [49]	350	VQ-VAE	AR	1.73
DiT-XL/2 [36]	1400	KL-VAE	Diffusion	2.27
MaskDiT [59]	1600	KL-VAE	Diffusion	2.28
SiT-XL/2 [33]	1400	KL-VAE	Flow	2.06
RAE, DiT-XL [58]	800	RAE	Diffusion	**1.41**
PixelFlow, XL/4 [7]	320	Pixels	Flow	1.98
PixNerd, XL/16 [50]	160	Pixels	Diffusion	2.15
JiT-H/16 [28]	600	Pixels	Flow	1.86
PixeLLM	200	Pixel blocks	AR + Flow	1.64

Method	gFID↓
Flow-based pixel refiner	
with PixeLLM-predicted draft	**2.25**
with Qwen3-VL embeddings	3.49

Table 3: **Effect of the AR draft.** ImageNet val-50k gFID at 256×256 after 10 AR epochs and 100 refiner epochs. The spectrum module is initialized from Stage 1, with its learned codebooks held fixed (Appendix F).

4.4 Ablation studies

Effect of the autoregressive draft stage. We replace the AR-predicted draft with class embeddings from Qwen3-VL-4B-Instruct [27] while retaining the pixel refiner. ImageNet gFID increases from 2.25 to 3.49 (Table 3), supporting the value of an explicit image-space structural condition in this setting. Figure 6 illustrates corresponding differences in layout and local consistency.

Effect of pixel refinement. Figure 7 compares the AR draft with the full pipeline. The draft captures coarse composition but retains quantization artifacts and lacks fine texture. Refinement improves local detail in the illustrated text, object, and facial regions. Together with the draft ablation, these observations support complementary roles for structure prediction and detail synthesis.



Figure 5: **Qualitative comparison for text-conditioned image generation.** PixeLLM and Janus-Pro-7B generate at 384×384; Tar-7B and Z-Image generate at 512×512.



Figure 6: **Effect of the AR draft.** Selected examples compare refinement conditioned on a PixeLLM draft with refinement conditioned directly on Qwen3-VL embeddings. The draft supplies explicit spatial structure.

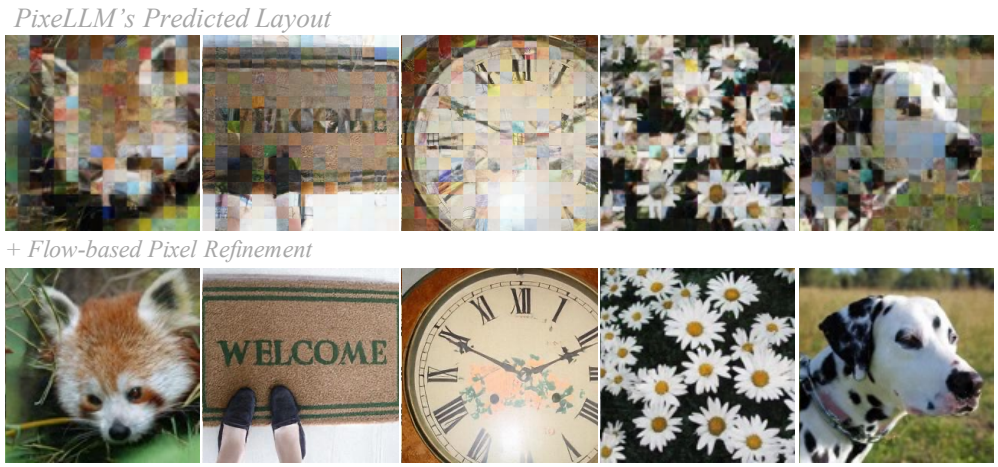


Figure 7: **Effect of pixel refinement.** AR drafts capture coarse composition; the conditional refiner improves texture and reduces local artifacts in the selected examples.

Table 4: **Ablations of spectrum serialization and refiner conditioning.** (a) GenEval2 Soft-TIFA AM/GM and ImageNet gFID for two tokenization/serialization designs. (b) ImageNet gFID for refiner conditioning and classifier-free guidance (CFG); training follows Table 3. Layout denotes predicted layout tokens and Text denotes text embeddings.

(a) Spectrum serialization.

Serialization	GenEval2		ImageNet
	AM↑	GM↑	gFID↓
Raster scan	65.1	20.8	3.01
Spectrum order	**70.5**	**31.3**	**2.25**

(b) Refiner conditioning and guidance.

Conditioning	ImageNet gFID↓
Layout	2.80
Layout + Text	2.59
Layout + Text + CFG	**2.25**

Effect of spectrum serialization. We compare spectrum-based serialization with a raster-scan baseline for Stage 1 at 256×256 resolution, holding the total codebook budget fixed at 262,144

entries. The raster-scan baseline uses a single codebook, whereas spectrum-based serialization allocates 65,536 entries to each of four frequency-specific codebooks. As shown in Table 4(a), spectrum-based serialization improves GenEval2 Soft-TIFA AM from 65.1 to 70.5 and GM from 20.8 to 31.3, while reducing ImageNet val-50k gFID from 3.01 to 2.25. This ablation evaluates frequency-specific codebooks and coarse-to-fine serialization jointly. The 262,144-entry budget matches the larger vocabulary in Figure 3; this ablation evaluates downstream generation with the main PixeLLM pipeline, whereas the figure isolates AR token prediction with Qwen3.5-2B.

Conditioning and guidance for pixel refinement. We evaluate the contributions of text conditioning and classifier-free guidance (CFG) to the flow-based pixel refiner on ImageNet val-50k. Starting from predicted layout tokens, we add text embeddings and then enable CFG. Table 4(b) shows that text conditioning reduces gFID from 2.80 to 2.59, and adding CFG further reduces it to 2.25. These results indicate that text conditioning and guidance complement the layout prior and improve generation quality under this evaluation protocol.

4.5 Computational overhead and inference efficiency

Table 5 compares model size and end-to-end inference efficiency across representative generation paradigms: Janus-Pro-7B [8] (AR with VQ-VAE), OmniGen2 [52] (flow-based DiT), Tar-7B [18] (AR with flow matching), and PixeLLM. All methods are evaluated at 384×384 resolution on a single NVIDIA H200 GPU with batch size 1 and BF16 precision. Input prompts are capped at 32 tokens, and all flow-based components use the Euler solver with 50 sampling steps and classifier-free guidance (CFG).

PixeLLM takes 6.98 s/image: 5.30 s for AR generation and 1.68 s for refinement. Its measured latency is 1.2% higher than OmniGen2 (6.90 s/image), 18.4% lower than Janus-Pro-7B (8.55 s/image), and $2.52\times$ faster than Tar-7B (17.58 s/image). Refinement accounts for 24.1% of PixeLLM's total latency, with AR decoding remaining the larger cost. PixeLLM has the smallest reported model size (5.8B) and the second-lowest latency among these four systems.

Table 5: **Model size and end-to-end inference efficiency at 384×384 resolution.** All methods are measured on a single NVIDIA H200 GPU with batch size 1, BF16 precision, and input prompts capped at 32 tokens. All flow-based components use 50-step Euler sampling with classifier-free guidance (CFG). † indicates CFG during AR sampling. Parameter counts are in billions (B), except VAE sizes in millions (M); totals sum the text/AR and flow components, with VAE sizes listed separately. Latency is in seconds per image.

Method	Pipeline	Parameters				Steps		Latency (s/image)		
		VAE (M)	Text/AR	Flow	Total	AR	Flow	Prompt/AR	Flow/decode	Total↓
Janus-Pro-7B	AR + VQ-VAE	71.95	7 AR	N/A	7	576†	N/A	8.53	0.02	8.55
OmniGen2	Flow	83.82	3 text	4	7	N/A	50	0.04	6.86	**6.90**
Tar-7B	AR + Flow	83.82	7 AR + 2 text	2.6	11.6	576	50	9.83	7.75	17.58
PixeLLM (ours)	AR + Flow	N/A	4.7 AR	1.1	**5.8**	576	50	5.30	1.68	<u>6.98</u>

5 Conclusion

PixeLLM revisits visual representation design for AR image synthesis through direct quantization of spectrum-transformed pixel blocks. Geometric diagnostics motivate this design, while controlled AR validation loss provides separate evidence of improved sequence predictability. Frequency-specific codebooks and coarse-to-fine serialization support an explicit division between AR structure prediction and pixel-space flow refinement. The resulting pipeline achieves competitive generation quality and inference efficiency without a separately pretrained image autoencoder.

Acknowledgements and disclosures

Yang Yang was supported by the 2035 Innovation Mission Project of CASIA (No. E4J10102). We thank Duxiaoman Technology for providing GPU resources for this work. Apart from the affiliations and support disclosed in this paper, the authors declare no other relevant financial relationships during the 36 months prior to submission.

References

[1] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. Technical report, OpenAI, 2023. URL https://cdn.openai.com/papers/dall-e-3.pdf.

[2] Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Xin Jin, Liangchen Li, et al. Z-image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.

[3] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024. URL https://arxiv.org/abs/2405.09818.

[4] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11315–11325, 2022.

[5] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

[6] Junying Chen, Zhenyang Cai, Pengcheng Chen, Shunian Chen, Ke Ji, Xidong Wang, Yunjin Yang, and Benyou Wang. Sharegpt-4o-image: Aligning multimodal models with gpt-4o-level image generation. *arXiv preprint arXiv:2506.18095*, 2025.

[7] Shoufa Chen, Chongjian Ge, Shilong Zhang, Peize Sun, and Ping Luo. Pixelflow: Pixel-space generative models with flow. *arXiv preprint arXiv:2504.07963*, 2025.

[8] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

[9] Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025. URL https://arxiv.org/abs/2505.14683.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[12] Runpei Dong, Chunrui Han, Yuang Peng, Zekun Qi, Zheng Ge, Jinrong Yang, Liang Zhao, Jianjian Sun, Hongyu Zhou, Haoran Wei, et al. Dreamllm: Synergistic multimodal comprehension and creation. *arXiv preprint arXiv:2309.11499*, 2023.

[13] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[14] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[15] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4239. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f0d629a734b56a642701bba7bc8bb3ed-Abstract-Conference.html.

[16] Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. Seed-x: Multimodal models with unified multi-granularity comprehension and generation. *arXiv preprint arXiv:2404.14396*, 2024.

[17] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.

[18] Jiaming Han, Hao Chen, Yang Zhao, Hanyu Wang, Qi Zhao, Ziyan Yang, Hao He, Xiangyu Yue, and Lu Jiang. Vision as a dialect: Unifying visual understanding and generation via text-aligned representations. *arXiv preprint arXiv:2506.18898*, 2025.

[19] Christopher E Heil and David F Walnut. Continuous and discrete wavelet transforms. *SIAM review*, 31(4): 628–666, 1989.

[20] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

[21] Minyoung Huh, Brian Cheung, Pulkit Agrawal, and Phillip Isola. Straightening out the straight-through estimator: Overcoming optimization challenges in vector quantized networks. In *International Conference on Machine Learning*, pages 14096–14113. PMLR, 2023.

[22] Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. Geneval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025.

[23] Dongwon Kim, Ju He, Qihang Yu, Chenglin Yang, Xiaohui Shen, Suha Kwak, and Liang-Chieh Chen. Democratizing text-to-image masked generative models with compact text-aware one-dimensional tokens. *arXiv preprint arXiv:2501.07730*, 2025.

[24] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

[25] Youngwan Lee, Kwanyong Park, Yoorhim Cho, Yong-Ju Lee, and Sung Ju Hwang. Koala: Empirical lessons toward memory-efficient and fast diffusion models for text-to-image synthesis. *Advances in Neural Information Processing Systems*, 37:51597–51633, 2024.

[26] Elizaveta Levina and Peter Bickel. Maximum likelihood estimation of intrinsic dimension. *Advances in neural information processing systems*, 17, 2004.

[27] Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, et al. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026.

[28] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

[29] Tianhong Li, Huiwen Chang, Shlok Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. Mage: Masked generative encoder to unify representation learning and image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2142–2152, 2023.

[30] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37:56424–56445, 2024.

[31] Bin Lin, Zongjian Li, Xinhua Cheng, Yuwei Niu, Yang Ye, Xianyi He, Shenghai Yuan, Wangbo Yu, Shaodong Wang, Yunyang Ge, et al. Uniworld: High-resolution semantic encoders for unified visual understanding and generation. *arXiv preprint arXiv:2506.03147*, 2025.

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

[33] Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, 2024. URL `https://www.ecva.net/papers/eccv_2024/papers_ECCV/papers/09828.pdf`.

[34] Ben Mann, Nick Ryder, Melanie Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 1 (3):3, 2020.

[35] Jordan Meyer, Nick Padgett, Cullen Miller, and Laura Exline. Public domain 12m: A highly aesthetic image-text dataset with novel governance mechanisms. *arXiv preprint arXiv:2410.23144*, 2024.

[36] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/papers/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.pdf.

[37] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[38] Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021.

[39] Liao Qu, Huichao Zhang, Yiheng Liu, Xu Wang, Yi Jiang, Yiming Gao, Hu Ye, Daniel K Du, Zehuan Yuan, and Xinglong Wu. Tokenflow: Unified image tokenizer for multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2545–2555, 2025.

[40] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[41] Christoph Schuhmann. LAION-Aesthetics. LAION, 2022. URL https://laion.ai/blog/laion-aesthetics/.

[42] Fengyuan Shi, Zhuoyan Luo, Yixiao Ge, Yujiu Yang, Ying Shan, and Limin Wang. Scalable image tokenization with index backpropagation quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16037–16046, 2025.

[43] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

[44] Keqiang Sun, Junting Pan, Yuying Ge, Hao Li, Haodong Duan, Xiaoshi Wu, Renrui Zhang, Aojun Zhou, Zipeng Qin, Yi Wang, et al. Journeydb: A benchmark for generative image understanding. *Advances in neural information processing systems*, 36:49659–49678, 2023.

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

[46] Quan Sun, Qiying Yu, Yufeng Cui, Fan Zhang, Xiaosong Zhang, Yueze Wang, Hongcheng Gao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. Emu: Generative pretraining in multimodality. In *The Twelfth International Conference on Learning Representations*, 2023.

[47] Quan Sun, Yufeng Cui, Xiaosong Zhang, Fan Zhang, Qiying Yu, Yueze Wang, Yongming Rao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. Generative multimodal models are in-context learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14398–14409, 2024.

[48] Onkar Kishor Susladkar, Tushar Prakash, Gayatri Deshmukh, Kiet A. Nguyen, Jiaxun Zhang, Adheesh Sunil Juvekar, Tianshu Bao, Lin Chai, Sparsh Mittal, Inderjit S Dhillon, and Ismini Lourentzou. Best of both worlds: Multimodal reasoning and generation via unified discrete flow matching. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 117699–117727. PMLR, 2026. URL https://proceedings.mlr.press/v306/susladkar26a.html.

[49] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *arXiv preprint arXiv:2404.02905*, 2024.

[50] Shuai Wang, Ziteng Gao, Chenhui Zhu, Weilin Huang, and Limin Wang. Pixnerd: Pixel neural field diffusion. *arXiv preprint arXiv:2507.23268*, 2025.

[51] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[52] Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, et al. Omnigen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*, 2025.

[53] Yecheng Wu, Junyu Chen, Zhuoyang Zhang, Enze Xie, Jincheng Yu, Junsong Chen, Jinyi Hu, Yao Lu, Song Han, and Han Cai. Dc-ar: Efficient masked autoregressive image generation with deep compression hybrid tokenizer. *arXiv preprint arXiv:2507.04947*, 2025.

[54] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[55] Tianwei Xiong, Jun Hao Liew, Zilong Huang, Jiashi Feng, and Xihui Liu. GigaTok: Scaling visual tokenizers to 3 billion parameters for autoregressive image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18770–18780, October 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Xiong_GigaTok_Scaling_Visual_Tokenizers_to_3_Billion_Parameters_for_Autoregressive_ICCV_2025_paper.html.

[56] Hu Yu, Hao Luo, Hangjie Yuan, Yu Rong, Jie Huang, and Feng Zhao. Frequency autoregressive image generation with continuous tokens. *arXiv preprint arXiv:2503.05305*, 2025. URL https://arxiv.org/abs/2503.05305.

[57] Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. Pixeldit: Pixel diffusion transformers for image generation. *arXiv preprint arXiv:2511.20645*, 2025.

[58] Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*, 2025. URL https://arxiv.org/abs/2510.11690.

[59] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=vTBjBtGioE.

[60] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.

A Limitations

Although PixeLLM achieves competitive results on text-conditioned and class-conditioned image generation benchmarks, these benchmarks cannot fully cover the diversity and ambiguity of real-world text-to-image prompts. In particular, prompts requiring precise counting, complex spatial reasoning, fine-grained text rendering, or rare object compositions may still expose failure cases. More comprehensive evaluations involving human preference studies and broader prompt distributions would provide a more complete assessment. The controlled AR comparison uses one backbone over five epochs at two vocabulary sizes, and jointly changes the representation and serialization. It does not establish asymptotic convergence or independently assess reconstruction fidelity. Full-scale experiments are not accompanied by repeated-run uncertainty estimates. The latency comparison covers inference on one hardware configuration, not training cost or scaling to higher resolutions.

B Broader impacts

PixeLLM contributes to text-to-image generation by reducing the dependence on separately trained learned image tokenizers and by exploring direct pixel-space autoregressive modeling. This direction may benefit visual content creation, design, education, and interactive media applications, where users require models that can generate images with faithful text alignment and detailed visual structure. By simplifying the representation pipeline, the proposed framework may also make future autoregressive image generators easier to analyze and extend.

At the same time, advances in text-to-image generation can introduce potential societal risks. More capable image generators may be misused to create misleading synthetic media, impersonation content, or visually realistic misinformation. They may also reproduce biases present in the training data or generate inappropriate content when prompted adversarially.

C Related work

Image generation with sequence models. Sequence modeling architectures originally developed for language (e.g., BERT [11] and GPT [34]) have been adapted to image generation by representing an image as a sequence of learned visual representations (discrete tokens or continuous latents) and learning to model that sequence. **GPT-style autoregressive generation** follows the next-token prediction strategy popularized by GPT, generating visual tokens sequentially, often under a raster-scan ordering [32, 8, 3, 60, 45, 13, 49]. Janus-Pro [8] extends autoregressive modeling toward unified multimodal generation and understanding, while Tar [18] introduces generative de-Tokenizers that generate images from discrete semantic tokens. Beyond next-token factorization, VAR [49] predicts successive spatial scales, and FAR [56] models frequency dependencies in continuous VAE latents. PixeLLM instead constructs discrete spectrum tokens directly from pixel blocks.

In contrast to next-token or next-scale prediction, **masked generative models** adopt a BERT-like objective that predicts masked tokens in parallel, reducing the degree of strictly token-by-token decoding and thereby enabling faster sampling in practice [4, 29]. MaskGIT [4] introduces iterative masked token prediction that updates multiple positions at each step. MAR [30] extends this idea to continuous latent spaces via "next set-of-tokens" prediction. DC-AR [53] combines masked generation with residual refinement by first generating global structure through discrete tokens and then refining details with residual predictions. MaskGen [23] presents a family of open-source text-to-image masked generative models that support both discrete and continuous token representations.

Discrete flow-matching models. Discrete flow matching [15] provides a non-autoregressive generative framework for categorical data. UniDFlow [48] extends this approach to multimodal understanding, image generation, and instruction-guided editing within a unified framework. Its image-generation process operates on discrete latent tokens from a pretrained visual tokenizer. The contribution concerns a unified discrete generative formulation, whereas PixeLLM investigates the representation and causal factorization used for AR synthesis.

Pixel-space generative models. Recent efforts also revisit image generation directly in pixel space, including diffusion/flow-based formulations with improved architectures and training. PixelFlow [7] develops hierarchical flow-based pixel-space models, and PixNerd [50] employs lightweight neural

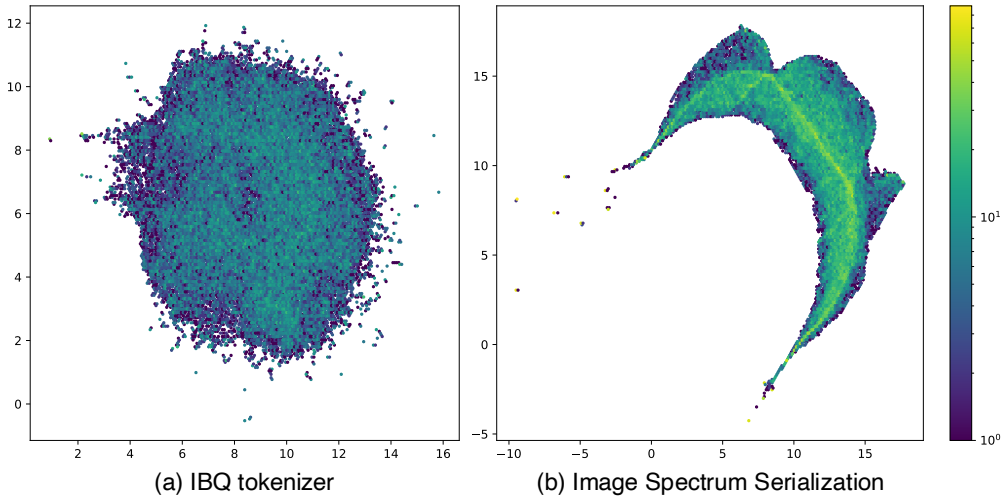


Figure 8: **UMAP visualization of quantized representations.** IBQ representations (left) and inverse-transformed spectrum representations (right) on ImageNet validation images. The projections illustrate differences in local organization; they are not quantitative tests of codebook utilization or generation quality.

field layers for efficient pixel-space diffusion. JiT [28] presents a purely transformer-based architecture that predicts clean image data directly. PixelDiT [57] introduces a dual-level design, using a patch-level DiT to capture global semantics and a pixel-level DiT to refine texture details.

Positioning of our work. PixeLLM combines three design choices: direct quantization of wavelet-transformed pixel blocks, frequency-specific codebooks with coarse-to-fine causal serialization, and pixel-space refinement conditioned on an AR draft. Unlike UniDFlow [48], it does not use discrete flow matching to unify understanding, generation, and editing. Unlike latent-token pipelines, it removes separate image-autoencoder pretraining, while still learning codebooks and a conditional refiner. The contribution lies in this representation and structure–detail decomposition, rather than a new flow-matching objective.

D Why spectrum serialization?

Spectrum factorization assigns separate codebooks to low-frequency structure and three orientations of detail. For a single spatial position in the transformed image, four codebooks of size K admit up to K^4 index tuples. This is a combinatorial property of the factorized representation, not a count of distinguishable natural images or a proof of higher capacity at fixed sequence length. A fair comparison must also account for spatial support, the number of emitted indices, and the distribution of occupied code combinations.

Our comparisons match the visual-token budget and total codebook size within each setting (Figure 3 and Table 4). They provide empirical support for the combined representation and serialization design, without independently identifying the effect of codebook factorization versus token ordering.

Figure 8 visualizes quantized representations using UMAP. For PixeLLM, quantized subband blocks are mapped back to the spatial domain by inverse DWT before embedding. The projections show different neighborhood structures, but UMAP is sensitive to the representation and projection procedure. Clustering in this visualization alone does not establish codebook collapse, information loss, or AR learnability. Quantitative evidence for AR predictability instead comes from held-out cross-entropy at matched vocabulary and sequence budgets in Figure 3.

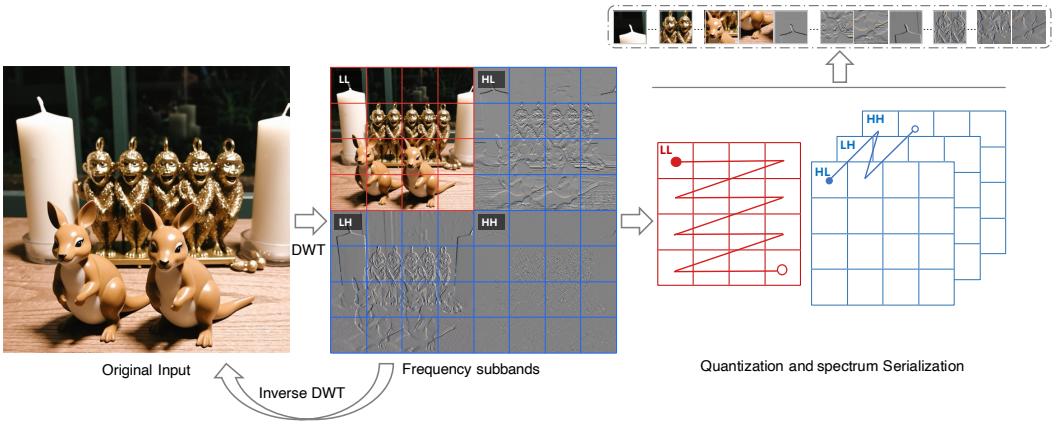


Figure 9: Illustration of spectrum serialization in PixeLLM.

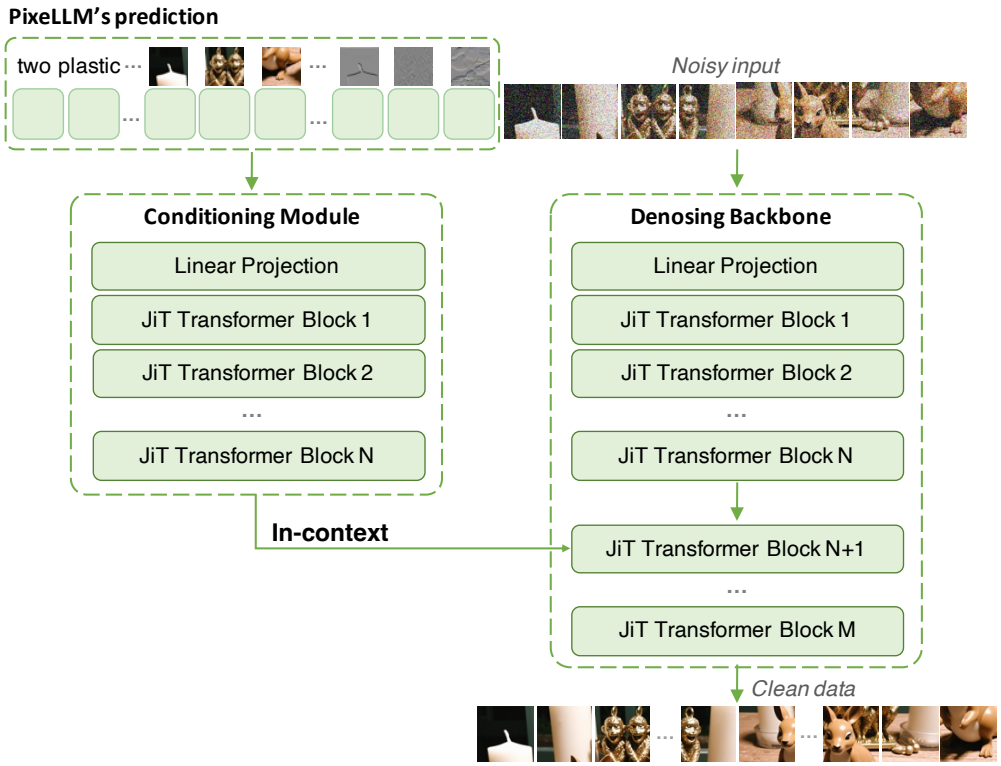


Figure 10: **Conditioning pathway in the flow-based pixel Transformer.** We fuse PixeLLM's predicted spectrum embeddings with text embeddings using a shallow conditional Transformer, and inject the resulting text-enriched features into the denoising backbone.

E Implementation details

E.1 Illustration of spectrum serialization

Figure 9 illustrates the spectral serialization procedure in PixeLLM. Starting from an input image, we apply a 2D Haar discrete wavelet transform to obtain four subbands: the low-frequency approximation (LL) and three high-frequency detail bands (LH, HL, HH). We encode the LL subband first: LL patches are quantized using a dedicated codebook and serialized in raster-scan order to form a coarse structural sequence. For the high-frequency content, we concatenate LH/HL/HH along the channel dimension, apply grouped vector quantization to better factorize the detail space, and serialize the

resulting tokens by spatial position. This yields a coarse-to-fine token stream—LL tokens followed by spatially aligned high-frequency tokens—that preserves local structure while exposing a natural refinement order. This ordering supplies low-frequency context when high-frequency tokens are predicted.

E.2 Model architecture

Parameter breakdown. Table 6 summarizes PixeLLM's parameter allocation across the LLM backbone and the flow-based pixel Transformer. All values are reported in millions of parameters (M).

LLM backbone and spectrum vocabulary expansion. We build PixeLLM on top of Qwen3-VL-4B-Instruct [27]. We expand the vocabulary with an additional $4 \times 65{,}536$ *image spectrum tokens*, which represent discrete tokens produced by pixel-space spectrum serialization. These tokens are treated as ordinary discrete tokens by the LLM during autoregressive training and generation. This design enables the model to predict a coarse spectrum-layout sequence conditioned on text prompts.

Flow-based pixel Transformer. We instantiate the pixel-space refiner as a JiT-style Transformer for flow modeling [28]. Our design follows JiT's overall architecture, but differs in how we incorporate conditioning. As shown in Figure 10, we first embed PixeLLM's predicted discrete spectrum blocks into a token sequence and concatenate these spectrum-token embeddings with the text-prompt embeddings. The concatenated sequence is then processed by a lightweight *conditioning encoder* (a conditional JiT module) of depth N (default $N{=}6$), yielding text-enriched representations in which the prompt provides semantic context for the spectrum-token stream.

The resulting conditioning features are subsequently provided to the JiT flow backbone (depth M, set to $M{=}32$; hidden dimension 1280), concatenated with the noised image input z_t (and timestep t). The backbone predicts the clean image in pixel space, i.e., $x_\theta(z_t, t, c)$ as used in Eq. (11).

Table 6: **Parameter breakdown of PixeLLM (in M parameters).** We separate token-related parameters inside the LLM backbone into *text tokens* (pretrained text vocabulary) and *visual tokens* (the added spectrum block tokens), and report the remaining LLM parameters as *Transformer*. The Pixel Transformer is decomposed into the conditioning module and the flow-based backbone.

Part	Module	Params (M)
LLM backbone	Text tokens	388.9
	Visual tokens	671.0
	Transformer	3633.5
Flow-based Pixel Refinement	Conditioning module	177.0
	Flow-based backbone	944.4
Total		**5814.8**

F Training recipe

Overview. We adopt a three-stage training recipe. First, we jointly pretrain the pixel-space spectrum serialization module and the flow-based Pixel Transformer. Second, we align the LLM text tokens with the spectrum block tokens by training the LLM to predict spectrum-token sequences conditioned on text prompts. Third, we further refine the flow-based Pixel Transformer for high-fidelity image synthesis, using both the LLM-predicted spectrum layouts and the text prompt embeddings as conditioning signals. The training hyperparameters are summarized in Table 7, and the complete

Table 7: **Training hyperparameters.** We report the training configurations for the flow-based Pixel Refiner and PixeLLM's autoregressive modeling.

Config	Setting	Flow-based Pixel Refinement	Autoregressive Modeling
Learning rate	–	1×10^{-4}	5×10^{-5}
LR schedule	–	cosine	cosine
Optimizer	–	AdamW	AdamW
Optimizer params	–	$\beta_1 = 0.9,\ \beta_2 = 0.99$	$\beta_1 = 0.9,\ \beta_2 = 0.99$
Weight decay	–	1×10^{-4}	0.0
Commitment loss weight	–	1.0	–
Conditioning drop for CFG	–	0.1	–
CFG scale	–	2.2	–
Input resolution	Class-cond.	256×256	256×256
	Text-cond.	384×384	384×384
Warmup epochs	Class-cond.	5	0.5
	Text-cond.	1	0.5
Epochs	Class-cond.	200	10
	Text-cond.	20	10
Total batch size	–	512	256

training procedure is provided in Algorithm 1. Training uses eight NVIDIA H200 GPUs; the inference-latency measurements use a single GPU as specified in Section 4.5.

Stage 1: Pretraining spectrum serialization and the pixel refiner. For text-to-image evaluation on GenEval [17], DPG-Bench [20], and GenEval2 [22], we pretrain the spectrum serialization module and the flow-based Pixel Transformer on a mixture of open-source image–text datasets, including PD12M [35], LAION-Aesthetic [41], LAION-Pop [25], and JourneyDB [44], comprising approximately 7M samples in total. In this stage, the text prompt p is not used, as described in Algorithm 1. The training objective jointly learns the spectrum-token representation and optimizes clean-image prediction through the flow-based Transformer.

Stage 2: Aligning text and spectrum tokens. In Stage 2, we train the LLM to predict the spectrum-token sequence conditioned on the user prompt, using the standard next-token negative log-likelihood defined in Algorithm 1. We freeze the pretrained text token embeddings and optimize the remaining parameters in ϕ, including the LLM Transformer blocks and the parameters associated with the newly introduced spectrum tokens.

Beyond the corpus used in Stage 1, we further incorporate instruction-following prompt–image data to improve text conditioning. Specifically, we use BLIP3o-60k [5], a curated set of approximately 60K prompt–image pairs distilled using GPT-4o, and ShareGPT-4o-Image [6], which contains approximately 45K text-to-image examples.

Stage 3: Refinement for high-fidelity synthesis. In Stage 3, we condition the flow-based Pixel Transformer on both (i) the spectrum layout tokens predicted by the LLM and (ii) the text prompt, and further optimize net_θ to synthesize images with improved fine-grained details. During this stage, the LLM parameters ϕ and the quantized spectrum blocks are kept fixed, and only the flow-based prediction objective is minimized.

For class-conditioned image generation on ImageNet val-50k, we use only the ImageNet training split to train all three stages, ensuring a fair comparison with strong class-conditioned baselines.

Algorithm 1 Overall training procedure of PixeLLM.

Require: LLM backbone llm_ϕ and flow-based Transformer net_θ
Require: Paired dataset $\mathcal{D} = \{(p_i, x_i)\}_{i=1}^{N}$ of text prompts p and images x

Stage 1: Pretrain spectrum serialization and net_θ.
for epoch $= 1, \ldots, M$ **do**
  for iter $= 1, \ldots, K$ **do**
    Sample $(p, x) \sim \mathcal{D}$ ▷ p is unused in this stage
    Sample $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and $\tilde{t} \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu = -0.8$, $\sigma = 0.8$; set $t \leftarrow \mathrm{sigmoid}(\tilde{t})$
    $z_t \leftarrow tx + (1-t)\epsilon$
    Obtain spectrum tokens $\{k_n^{\mathrm{low}}\}$ and $\{k_{n,g}^{\mathrm{high}}\}$.
    Predict clean image $\hat{x} \leftarrow \mathrm{net}_\theta\big(z_t, t \,\big|\, \{k_n^{\mathrm{low}}\}, \{k_{n,g}^{\mathrm{high}}\}\big)$
    Update θ and tokenization parameters by minimizing the loss in Eq. (5) and Eq. (11).
  end for
end for

Stage 2: Align text tokens with spectrum tokens in llm_ϕ.
for epoch $= 1, \ldots, M$ **do**
  for iter $= 1, \ldots, K$ **do**
    Sample $(p, x) \sim \mathcal{D}$
    Obtain spectrum tokens $\{k_n^{\mathrm{low}}\}$ and $\{k_{n,g}^{\mathrm{high}}\}$ from x.
    Construct a one-turn conversation sequence s:

```
<|im_start|> user {p} <|im_end|>
<|im_start|> assistant {spectrum-token sequence} <|im_end|>
```

    Compute next-token negative log-likelihood:

$$\mathcal{L}_{\mathrm{LLM}} = -\sum_{j=1}^{|s|} \log P_{\mathrm{llm}_\phi}(s_j \mid s_{<j})$$

    Freeze pretrained text token embeddings; update the remaining parameters in ϕ by minimizing $\mathcal{L}_{\mathrm{LLM}}$
  end for
end for

Stage 3: Refine net_θ conditioned on LLM-predicted layouts.
for epoch $= 1, \ldots, M$ **do**
  for iter $= 1, \ldots, K$ **do**
    Sample $(p, x) \sim \mathcal{D}$
    Sample $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and $\tilde{t} \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu = -0.8$, $\sigma = 0.8$; set $t \leftarrow \mathrm{sigmoid}(\tilde{t})$
    $z_t \leftarrow tx + (1-t)\epsilon$
    Generate LLM-predicted spectrum layout tokens $\{\tilde{k}_n^{\mathrm{low}}\}, \{\tilde{k}_{n,g}^{\mathrm{high}}\} \leftarrow \mathrm{llm}_\phi(p)$
    Predict clean image $\hat{x} \leftarrow \mathrm{net}_\theta\Big(z_t, t \,\Big|\, \{\tilde{k}_n^{\mathrm{low}}\}, \{\tilde{k}_{n,g}^{\mathrm{high}}\}, p\Big)$
    Keep ϕ fixed; update θ by minimizing the flow-based prediction objective in Eq. (11)
  end for
end for

G Additional experimental results

G.1 Additional benchmark-specific metrics

Table 8 illustrates additional benchmark-specific metrics on GenEval and DPG-Bench at 384×384 resolution. PixeLLM shows consistently strong performance across key categories, including entity grounding and relational understanding.

Table 8: Additional benchmark-specific metrics on GenEval and DPG-Bench.

(a) **GenEval**

Tokenizer	Generator	type	S. Obj.	T. Obj.	Count.	Colors	Position	C. Attri.	Overall↑
KL	SD3-Medium [14]	Diff	0.99	0.94	0.72	0.89	0.33	0.60	0.74
KL	MaskGen-XL [23]	Mask	0.99	0.61	0.55	0.81	0.13	0.31	0.57
KL	DC-AR [53]	Mask	1.0	0.75	0.52	0.90	0.45	0.51	0.69
VQ	LlamaGen [45]	AR	0.71	0.34	0.21	0.58	0.07	0.04	0.32
VQ	TokenFlow-XL [39]	AR	0.95	0.60	0.41	0.81	0.16	0.24	0.55
VQ	Emu3-Gen [51]	AR	0.99	0.81	0.42	0.80	0.49	0.45	0.66
VQ	Show-o [54]	Mask	0.95	0.52	0.49	0.82	0.11	0.28	0.53
VQ	Janus-Pro [8]	AR	0.99	0.89	0.59	0.90	**0.79**	**0.90**	0.80
KL	OmniGen2 [52]	Flow	1.0	0.95	0.64	0.88	0.55	<u>0.76</u>	<u>0.80</u>
–	PixeLLM	AR+Flow	**1.0**	<u>0.90</u>	**0.81**	**0.90**	<u>0.66</u>	0.67	**0.82**

(b) **DPG-Bench**

Tokenizer	Generator	type	Global	Entity	Attribute	Relation	Other	Overall↑
KL	DALL-E 3 [1]	Diff	90.97	89.61	88.39	<u>90.58</u>	<u>89.83</u>	83.50
KL	SD3-Medium [14]	Diff	<u>87.90</u>	**91.01**	88.83	80.70	88.68	84.08
VQ	Emu3-Gen [51]	AR	85.21	86.68	86.84	90.22	83.15	80.60
KL	UniWorld-V1 [31]	Flow	83.64	88.39	88.44	89.27	87.22	81.38
VQ	TokenFlow-XL [39]	AR	78.72	79.22	81.29	85.22	71.20	73.38
CLIP	BLIP3-o 8B [5]	Flow	-	-	-	-	-	81.60
KL	OmniGen2 [52]	Flow	**88.81**	88.83	**90.18**	89.37	**90.27**	83.57
VQ	Janus-Pro [8]	AR	86.90	88.90	<u>89.40</u>	89.32	89.48	<u>84.19</u>
–	PixeLLM	AR+Flow	83.44	<u>90.05</u>	87.66	**92.71**	84.63	**84.41**

G.2 Additional text-to-image examples

Figure 11 provides additional qualitative comparisons for text-conditioned image generation between **PixeLLM** and representative baselines, including autoregressive models (Janus-Pro-7B [8], Tar-7B [18]) and a diffusion-based model (Z-Image [2]). These examples are included to further illustrate differences in prompt adherence (e.g., object/attribute binding) and overall scene coherence under the same input prompts.

Figure 12 shows additional qualitative results of **PixeLLM** on DPG-Bench [20] at 384×384 resolution, focusing on cases with long, information-dense captions.

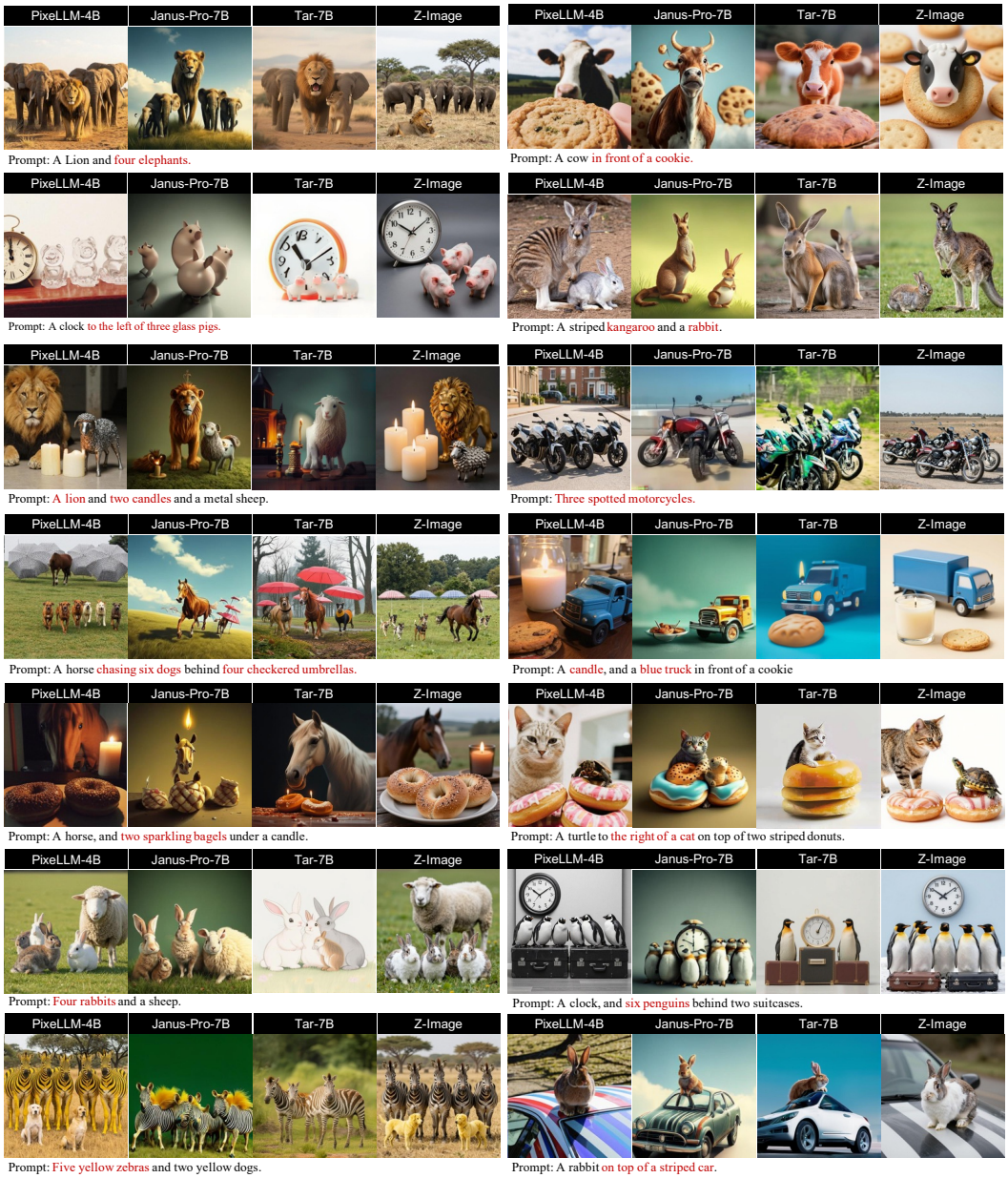
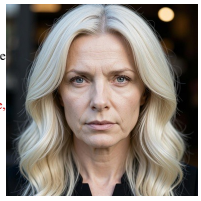


Figure 11: Additional qualitative comparisons on text-to-image generation.



A high-resolution portrait of the acclaimed actress Julianne Moore trending on Pinterest, captured by the lens of photographer Kyle Thompson. Her hair is styled in full platinum blonde waves that frame her intensely expressive, pale-skinned visage with high detail. The photograph, exuding realism and superior quality, showcases her piercing gaze that imparts a sense of subtle strength.



A striking portrait that captures the essence of Salvador Dalí, with one side of his face depicted in his iconic, surrealistic style, and the other half transformed into a metallic, robotic visage. The painting features a vivid array of colors, with the robotic side incorporating shades of silver and hints of circuitry, contrasting with the warm, flesh tones of Dalí's human side. The background is a simple, solid color to ensure the focus remains on the intricate details of Dalí's dual representation.



A unique flower with delicate pink petals arranged in a circular pattern, at the center of which is an intricate design resembling the face of a cat with green eyes. The flower's texture appears soft and velvety, and it is situated among a bed of green leaves that provide a contrasting backdrop to its whimsical feature.



A domestic cat with a sleek coat stands alert next to a plush brown teddy bear. The teddy bear is seated on a hardwood floor, its size comparable to the feline companion beside it. Both the cat and the teddy bear are positioned near a light-colored wall, with a hint of a houseplant's green leaves in the corner of the scene.



A piece of golden-brown toast resting on a white ceramic plate, topped with bright green, creamy slices of avocado arranged neatly. next to the plate, there's a silver knife with some avocado residue on the blade, and a sprinkle of red pepper flakes over the avocado slices adds a pop of color.



Two zebras, their black and white stripes creating a dizzying effect, are galloping side by side across the sprawling savanna, tinged golden by the sunlight. The texture of the grass appears rough and dry, a testament to the arid climate. Dust is being kicked up by their hooves, and in the background, sparse acacia trees punctuate the otherwise open landscape.



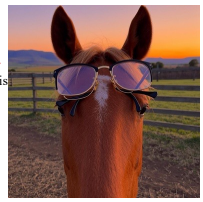
On the soft, warm sand of the beach, a fluffy white rabbit with rounded ears is caught in a curious moment, gently placing its paw on the ribbed surface of a pink scallop shell. The scallop, slightly open, reveals its smooth interior contrasting with its coarse outer texture, while hues of pink and orange from the setting sun reflect off its surface. There's a tranquil ocean backdrop with the gentle ebbing of the tide, and the fading daylight casts a golden glow over the scene, highlighting the rabbit's soft fur and the shell's subtle color.



A rustic wooden table with a natural grain finish, bathed in soft light. On its surface, a cluster of ripe oranges is arranged next to two glass jars filled with a vibrant orange marmalade. The jars catch the light, highlighting the rich color and texture of the contents within.



A detailed oil painting depicting a ginger cat with green eyes, intently focused on a game of checkers. The cat is seated at a small wooden table, with the checkerboard laid out in front of it, pieces strategically placed. The painting captures the texture of the cat's fur and the wood grain of the table, set against a soft, neutral background that draws attention to the subject.



A whimsical scene unfolds as two pairs of spectacles rest comically on the bridge of a horse's nose, glinting slightly against the fading light of a vibrant sunset. The backdrop is a quintessential rural farm, complete with wooden fences and distant rolling hills painting the horizon in hues of orange and purple. The glasses, one with round, golden frames and the other with square, jet-black rims, add a touch of eccentricity to the tranquil pastoral setting.



In a spacious room, an ornate vintage clock with a large, circular face and roman numerals stands prominently against a pastel-colored wall, its intricate hands pointing to the time. Beside it, a small, oval-shaped mirror with an elegant antique frame hangs neatly, reflecting a portion of the room's interior. The contrast in size between the commanding presence of the clock and the modest mirror creates a unique focal point in the space.



A curious vessel, with an architecture reminiscent of a giant green broccoli, basks in the bright sunlight, casting a shimmer across its intricate, leaf-like structures. It floats serenely in the midst of a vast ocean, the water around it sparkling as if sprinkled with diamonds due to the sun's reflection. The horizon stretches endlessly, with the clear blue sky meeting the deep azure of the sea at a distant line.

Figure 12: Additional qualitative results on DPG-Bench. PixeLLM generations at 384×384 resolution for DPG-Bench prompts with long captions, illustrating performance under longer, multi-constraint textual descriptions.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction distinguish geometric motivation from controlled AR evidence. Sections 2–4 present the analysis, method, and generation experiments supporting these claims.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss limitations in Appendix A.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The paper does not present formal theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide the model architecture, training objectives, datasets, evaluation protocols, and implementation details.

   Guidelines:

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

5. **Open access to data and code**

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

Answer: [No]

Justification: Implementation details are provided in the paper. The repository linked in the abstract currently contains only an empty README; code and model checkpoints have not yet been released.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We specify the datasets, preprocessing, optimizer, hyperparameters, training schedule, and evaluation settings.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: We do not report error bars for full-scale training runs due to the high computational cost of repeating large-scale generative model training, but we provide consistent evaluation protocols and ablation studies.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [No]

   Justification: Appendix F reports the GPU type, device count, and training epochs. Per-run wall-clock training time and total project compute are not reported.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that the research conforms to it.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss both positive applications and potential negative societal impacts in Appendix B.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper does not release models or datasets that pose high misuse risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [No]

    Justification: Existing datasets, models, and benchmarks are cited, but this version does not provide a consolidated statement of the licenses and terms of use for every asset.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: This version does not include a release of new code, model checkpoints, or datasets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: Qwen3-VL-4B-Instruct is the autoregressive backbone of PixeLLM. Its role, vocabulary expansion, conditioning, and training are described in Section 3 and Appendices E.2 and F. Qwen3.5-2B is used separately for controlled AR probing in Section 2.2.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.