

Visual Self-Refinement for Autoregressive Models

Jiamian Wang^{1*†}, Ziqi Zhou^{1†}, Chaithanya Kumar Mummadi²,
Sohail Dianat¹, Majid Rabbani¹, Raghuveer Rao³, Chen Qiu^{2‡} and Zhiqiang Tao^{1‡}
¹Rochester Institute of Technology, ²Bosch Research, ³DEVCOM Army Research Laboratory

Abstract

Autoregressive models excel in sequential modeling and have proven to be effective for vision-language data. However, the spatial nature of visual signals conflicts with the sequential dependencies of next-token prediction, leading to suboptimal results. This work proposes a plug-and-play refinement module to enhance the complex spatial correspondence modeling within the generated visual sequence. This module operates as a post-pretraining step to jointly refine all generated tokens of autoregressive model, enhancing vision-language modeling under a shared sequential prediction framework. By leveraging global context and relationship across the tokens, our method mitigates the error accumulation issue within the sequential generation. Experiments demonstrate that the proposed method improves the generation quality, enhancing the model’s ability to produce semantically consistent results.

1 Introduction

Autoregressive (AR) models have achieved remarkable success in recent years across multimodal tasks (Brown et al., 2020; Achiam et al., 2023; Touvron et al., 2023; Team et al., 2023; Bai et al., 2023, 2024; Yu et al., 2024a; Sun et al., 2024c; El-Nouby et al., 2024; Sun et al., 2025). Large language models (LLMs) based on autoregressive modeling (Brown et al., 2020; Achiam et al., 2023; Touvron et al., 2023; Team et al., 2023) encode text as sequences of tokens and predict each token sequentially based on the preceding tokens. This next-token prediction paradigm effectively captures sequential dependencies and complex semantic relationships in the text, and shown to excel in tasks like question answering (Trischler et al., 2016; Minaee et al., 2021; Sun et al., 2024a,b) and text generation (Hendrycks et al., 2021b,a).

Building on this success, recent works have extended autoregressive modeling to visual data (Bai et al., 2024; Yu et al., 2024a; Tian et al., 2024; Sun et al., 2024c; El-Nouby et al., 2024). Among them, Large Vision Model (LVM) (Bai et al., 2024) stands out for its focus on in-context tasks. LVM encodes images (or video frames) as a sequence of tokens and uses next-token prediction to solve various image translation and generation tasks. The focus on in-context learning makes LVM well-suited to handle vision tasks that require contextual information for generating coherent and consistent outputs.

Unlike text, visual tokens exhibit strong spatial correlations rather than purely sequential dependencies. This spatial structure necessitates attention to both prefix and future tokens for coherent generation. Consequently, generating tokens solely based on preceding tokens may compromise global contextual awareness, and the sequential nature of generation further exacerbates this by allowing early errors to accumulate, degrading both visual fidelity and semantic consistency.

To address these challenges, we propose a plug-and-play self-refinement module that operates on the generated tokens while keeping the pretrained backbone frozen for the post-pretraining finetuning (see Figure 1). Specifically, the proposed module jointly optimizes the generated tokens under the supervision of the target image. The discrete tokens are mapped to the embedding space to facilitate the learning. To this end, the broader view enables consistent corrections across the generated token sequence, mitigating errors stemming from the prefix context in the autoregressive generation.

We summarize the contributions as follows: 1) We introduce a post-pretraining finetuning operation to improve the next-token prediction for the visual data, without harming the generalizability of the pretrained AR model. 2) We propose to learn a plug-and-play lightweight module to facilitate the refinement, with small-scale data and negli-

*Work partially done during an internship at Bosch.

†Equal contributions.

‡Corresponding authors: Chen Qiu and Zhiqiang Tao

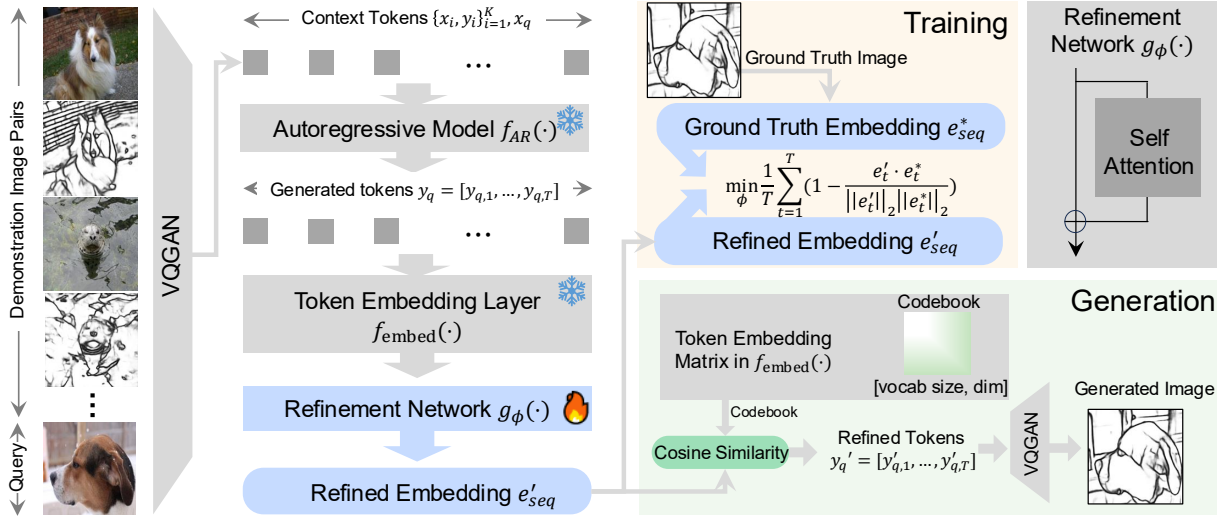


Figure 1: Training and generation pipeline of the proposed method. Input images (K demonstrations pairs and a query image) are encoded and fed into $f_{AR}(\cdot)$ for generating tokens iteratively. A refinement network $g_{\phi}(\cdot)$ is introduced to improve the generated tokens y_q in the embedding space, yielding e'_{seq} . During training, we use ground truth embedding e^*_{seq} for supervision and minimize the cosine distance. During generation, the refined embedding e'_{seq} is decoded to the discrete tokens via nearest-neighbor search. Finally, the refined tokens y'_q are passed to VQGAN decoder to generate the image. We take edge detection as an example.

ble running time cost. 3) Experiments demonstrate notable improvements and generalizability of the proposed method not only across various vision tasks but also on different pretrained AR backbones, with supporting empirical evidence showing improvement in token-wise prediction accuracy.

2 Related Work

Autoregressive modeling serves as main streamline for image generation. LlamaGen (Sun et al., 2024c) develops text-conditioned visual generation, AIM (El-Nouby et al., 2024) studies scalability in model capacity, data quantity and objective function, RAR (Yu et al., 2024b) rearranges token orders for bidirectional correspondence modeling, ELM (Liu et al., 2024) studies the design space of LLMs for vision tasks, and VAR (Tian et al., 2024) incorporates visual inductive bias via a next-scale prediction mechanism.

While these methods primarily focus on unconditional, and text/class-conditional generation, they fail to utilize images as context. LVM (Bai et al., 2024) address this gap by leveraging context image pairs for inferring tasks and generation, allowing a series of perception tasks (Wang et al., 2023) or reconstruction tasks (Wang et al., 2022, 2025, 2024). However, its reliance on next-token prediction leads to error accumulation and struggles to model spatial correspondences (Tian et al., 2024).

3 Method

3.1 Preliminaries on LVM

Autoregressive modeling. aims to maximize the likelihood of a discrete token sequence $\mathbf{x} = [x_1, x_2, \dots, x_T]$ using a forward autoregressive factorization, where each token x_t , *i.e.*, discrete value index, is predicted based on all preceding tokens $[x_1, x_2, \dots, x_{t-1}]$:

$$\max_{\theta} p_{\theta}(\mathbf{x}) = \prod_{t=1}^T p_{\theta}(x_t | x_1, x_2, \dots, x_{t-1}), \quad (1)$$

where p_{θ} represents a token distribution predictor parameterized by θ . This sequential next-token prediction captures dependencies within the data and thus dynamically adapt to new tasks based on the contextual information.

In-Context Visual Generation. LVM (Bai et al., 2024) extended this next-token prediction modeling to different image translation and generation tasks through in-context learning. Given the input consisting K demonstration pairs $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^K$ and a query image tokens $\mathbf{x}_q$ (e.g. rgb and sketch images in Fig. 1), the model dynamically adapts to the translation tasks such as inpainting or edge detection, and generates output tokens $\mathbf{y}_q = [y_{q,1}, y_{q,2}, \dots, y_{q,T}]$ sequentially using a autoregressive backbone f_{AR} with parameters θ :

$$y_{q,t} = \begin{cases} f_{\text{AR}}(\mathbf{x}_q, \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^K; \theta), & t = 1, \\ f_{\text{AR}}(y_{q,<t}, \mathbf{x}_q, \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^K; \theta), & t > 1, \end{cases} \quad (2)$$

f_{AR} is iteratively applied to generate the target tokens y_q , conditioned on the query and context.

3.2 Next-Token Prediction for Visual Tokens

The next-token prediction generates tokens one by one in a fixed order, where each step counts on the past context and fits the natural languages with a sequential order. However, next-token prediction lacks a full view of the whole sequence at each step, being improper for the image data whose information is spatially distributed and correlated (Tian et al., 2024). This makes it hard for the mechanism to maintain the global consistency and coherence for the generated image. Prior methods typically impose a predefined token generation order, such as raster-scan or spiral patterns (Yu et al., 2024a). However, these approaches generate each token based solely on preceding tokens and disregards useful information from future tokens that could provide additional contextual coherence. As a result, this sequential dependency leads to suboptimal outputs, particularly for complex visual tasks that requires holistic context of the entire image.

3.3 Self-Refinement for Visual Generation

To address these limitations, we propose a post-pretraining step to refine the generated visual token sequence $\mathbf{y}_q$. Specifically, we introduce a self-refinement module to jointly transform all the generated output tokens $\mathbf{y}_q = [y_{q,1}, y_{q,2}, \dots, y_{q,T}]$ into refined tokens $\mathbf{y}'_q = [y'_{q,1}, y'_{q,2}, \dots, y'_{q,T}]$, while keeping the autoregressive backbone frozen. By incorporating global context, the refinement module captures holistic relationships across all tokens, and thus significantly enhances the quality and coherence of the final visual output. The refinement process involves: 1) refining the generated token embeddings with a self-attention module, 2) decoding the refined embeddings to discrete tokens via nearest-neighbor search.

Self-Refinement Design. The autoregressive generated tokens $\mathbf{y}_q$ are processed by pretrained embedding layer $f_{\text{embed}}(\cdot)$ as the embeddings $\mathbf{e}_{seq} = [\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_T]$. A refinement network

g_ϕ parameterized by ϕ , jointly processes these embeddings to produce refined embeddings $\mathbf{e}'_{seq} = [\mathbf{e}'_1, \mathbf{e}'_2, \dots, \mathbf{e}'_T]$, which can be computed as:

$$\mathbf{e}'_{seq} = \mathbf{e}_{seq} + \text{SelfAttention}(\mathbf{e}_{seq}; \phi) \quad (3)$$

The self-attention module captures spatial relationships, semantic coherence and long-range dependencies across the sequence. Importantly, processing all the embeddings simultaneously make the refinement step to adjust for inconsistencies and deviations introduced during sequential generation.

Self-Refinement Optimization. For post-pretraining fine-tuning, the refinement module $g_\phi(\cdot)$ is optimized and operates as an plug-and-play component upon the pretrained LVM. Specifically, we jointly consider all the generated tokens in the latent space and minimize the cosine distance between the sequence of refined embeddings $\mathbf{e}'_{seq}$ and the ground truth token embeddings $\mathbf{e}_{seq}^*$:

$$\min_{\phi} \frac{1}{T} \sum_{t=1}^T \left(1 - \frac{\mathbf{e}'_t \cdot \mathbf{e}_t^*}{\|\mathbf{e}'_t\|_2 \|\mathbf{e}_t^*\|_2} \right), \quad (4)$$

which provides the supervision for the generated embedding to align with the target distribution, improving the fidelity and coherence of the output with small-scaled data.

Generation. A decoding step is then performed to transform the refined embeddings $\mathbf{e}'_{seq}$ into discrete tokens $\mathbf{y}'_q$ to ultimately reconstruct the output image. Given the embeddings of all tokens in the codebook (*i.e.*, embedding matrix from pretrained $f_{\text{embed}}(\cdot)$), we identify the nearest token embedding to each refined embedding based on cosine similarity. The token associated with the closest token embedding is selected as the refined token. Once determined, the refined tokens are decoded into images using the VQGAN decoder.

4 Experiment

We evaluate our method on three vision tasks: image colorization, inpainting, and edge detection.

Datasets. We adopt the large-scale Unified Vision Dataset (UVDv1) curated by LVM (Bai et al., 2024) for each task. The proposed self-refinement module is tuned with small-scale data (*e.g.*, 12K image pairs). The image resolution is 256×256 .

Baselines. We compare with three prevalent baselines. We adopt pretrained LVM (LLaMA-7B) (Bai et al., 2024) for vanilla in-context generation (LVM). We include one baseline (+Context

Table 1: Comparison of our self-refinement method against baselines across different vision tasks.

Method	Colorization					
	Perp ↓	LPIPS ↓	FID ↓	IS ↑	PSNR ↑	SSIM ↑
LVM	20.06	0.29	59.70	27.20	17.93	0.4925
+Context Retrieval	19.94	0.30	59.63	26.59	17.87	0.4936
+LoRA	19.04	0.29	59.76	27.14	18.09	0.5005
+Self-Refinement	19.01	0.28	59.24	27.34	18.78	0.5086

Method	Inpainting					
	Perp ↓	LPIPS ↓	FID ↓	IS ↑	PSNR ↑	SSIM ↑
LVM	175.93	0.31	63.95	25.35	17.38	0.4340
+Context Retrieval	183.22	0.31	64.31	23.79	17.41	0.4356
+LoRA	172.80	0.31	64.42	25.49	17.54	0.4395
+Self-Refinement	82.63	0.27	59.60	26.71	18.40	0.4508

Method	Edge Detection		
	Perp ↓	Acc ↑	Recall ↑
LVM	23.53	0.76	0.80
+Context Retrieval	23.24	0.76	0.81
+LoRA	22.17	0.84	0.86
+Self-Refinement	20.00	0.83	0.88

Table 2: Self-refinement for the image generation. We fix the VAR and only fine-tune self-refinement module.

Metrics	Vanilla VAR	VAR w/ self-refinement (ours)
FID ↓	3.55	3.43
IS ↑	284.2	287.4
Precision ↑	0.85	0.87
Recall ↑	0.49	0.51

Retrieval) that enhances the generation with context images retrieved based on similarity-search (Zhang et al., 2023). We include another baseline (+LoRA) that finetunes LVM with LoRA (Hu et al., 2021) on each task. We apply LoRA with the rank of 8 and insert them into query and value projection layers of the multi-head attention modules in each Transformer block. Our method only learns the refinement network $g_\phi(\cdot)$ without LoRA module (+Self-Refinement). Both LoRA fine-tuning and our method are trained with batch size of 4 using AdamW optimizer and learning rate $1e^{-4}$ for 2 epochs. We adopt 4 context pairs for all tasks. We provide more details on metrics in the appendix.

4.1 Performance Comparison

Image Perception. In Table 1, we demonstrate the results of our self-refinement method and baselines on colorization, inpainting and edge detection tasks. Our method demonstrates superior performance across all three tasks. For colorization, our method achieves the best performance across all metrics, demonstrating superior visual quality and consistency. For inpainting, our self-refinement method significantly improves perplexity (82.63), far outperforming all other methods. It also delivers the best LPIPS, FID, IS, PSNR, and SSIM, demonstrating its robustness and superiority in reconstructing realistic and coherent content. On

Table 3: Performance comparison under structural-error scenario on colorization.

Methods	Perplexity	LPIPS	FID	IS	PSNR
w/o Self-refinement	23.70	0.35	65.85	25.00	16.07
w/ Self-refinement	20.53	0.30	60.10	26.90	17.82

edge detection, our method achieves the lowest perplexity, indicating superior model confidence, and the highest recall, showcasing its ability to capture more relevant edges. While its accuracy (0.83) is slightly below LoRA finetuning (0.84), it maintains better perplexity and recall, demonstrating a balanced and robust performance. Note that the number of the trainable parameters is 14.7M for LoRA modules and 16.1M for the proposed self-refinement module, being comparable regarding the model size.

Image Generation. In Table 2, we extend the proposed self-refinement design to the visual autoregressive model of VAR (Tian et al., 2024) under the task of class-conditional generation. We fine tune the self-refinement module on ImageNet 256x256 and evaluation on its benchmark dataset. Overall, the proposed method demonstrates the generalizability on (1) different AR models and (2) on different tasks including perception and generation. We set the same temperature and the seed as the vanilla VAR to improve the statistical stability for the generation.

Error Accumulation Discussion. Fig. 2 compares the cosine distance of generated embeddings and the ground-truth embeddings between the LVM with LoRA and with self-refinement under inpainting. LVM with self-refinement achieves significantly reduced cosine distance for early tokens ($25 < t < 70$) and later tokens ($t > 160$), enabling smaller AUC value. The results highlight that the self-refinement process corrects early errors and mitigates error accumulation during generation. For the quantitative comparison, we feed the first 70 refined tokens (which manifest reduced cosine distance) into the LVM to generate the rest tokens, and then comparing the quality of the remaining generated tokens with and without refinement in Table 4. Both the token-level evaluation (e.g., accuracy for the remaining generated tokens) and the image-level evaluations are improved with the proposed method. The self-refinement module works differently with the next-token prediction by jointly optimizing the entire generated sequence and adjusting the structural inconsistency globally. Thus better handle the potential fundamental mis-

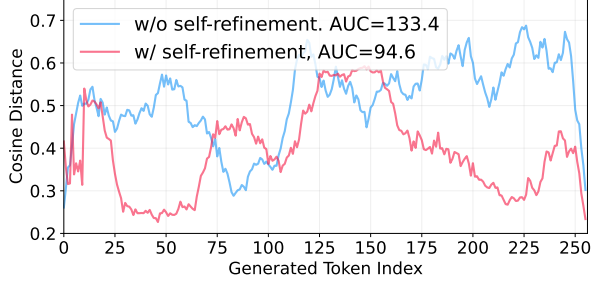


Figure 2: We compare the LVM with LoRA and the LVM with self-refinement in terms of the cosine distance among generated embeddings and the ground truth embeddings for 256 tokens. Self-refinement reduces the error accumulation especially for latter tokens (> 160). The accumulated errors (*i.e.*, by AUC) are provided.

takes, improving the quality of the discrete tokens and alleviates the error accumulation.

Structured Error Discussion. The proposed module works differently with the next-token prediction by jointly optimizing the entire generated sequence and adjusts the structural inconsistency globally. Thus better handle the potential fundamental mistakes that the next-token prediction could not solve. In Table 3, we simulate the scenario with structural error by locally permuting a segment of the token embeddings (randomly selecting 10% of the continuous tokens from the generated sequence) and compare the performance after decoding. The vanilla LVM (“w/o self-refinement”) experiences large performance descent compared to the baseline in Table 1, indicating severe sequential prediction errors. The proposed self-refinement module brings remarkable performance boost.

Breakdown Running Time Analysis. We compute the running time on a workstation with 2x NVIDIA RTX A6000 GPUs, Intel Core i9-10900X @ 3.70GHz, 10 cores / 20 threads CPU and system memory of 128GB. As shown in Table 5, the self-refinement module is lightweight and only requires 0.112s to process. We perform nearest-neighbor lookup for 256 generated tokens in parallel, thus being efficient. The token generation induces most time cost and depends on the number of tokens in the input sequence, which does not affect the self-refinement.

5 Conclusion

This work proposed a lightweight post-processing technique to improve the next-token generation for the image perception and generation. By jointly refining the tokens, the proposed method jointly incorporated the global contextual information in the

Table 4: Quality comparison of the remaining generated tokens on colorization.

Metrics	w/o Self-refinement	w/ Self-refinement
Gen Token Acc $\uparrow$	0.667	0.694
Perplexity $\downarrow$	20.06	19.50
LPIPS $\downarrow$	0.29	0.29
FID $\downarrow$	59.70	59.66
IS $\uparrow$	27.20	27.28
PSNR $\uparrow$	17.93	18.53
SSIM $\uparrow$	0.4925	0.5054

Table 5: Breakdown running time (s) analysis under $k = 4$ demonstration pairs. We compute average generation time (per image) and compare under the same platform.

Operations	LVM Baseline	Ours
Token generation	4.578	4.578
Self-Refinement module processing	–	0.112
Nearest-neighbor look-up	–	0.155
VQGAN decoding	0.233	0.233
Total time cost (s)	4.811	5.078

image, mitigating the error accumulation in vanilla next-token prediction without affecting the generative capabilities of the pretrained AR model, enhancing the visual coherence and the performance.

Limitations. As a post-processing step, refinement process does not directly influence the initial autoregressive generation. Errors introduced during the sequential token generation stage are mitigated with refinement but not eliminated. Future works could explore tighter integration between refinement and generation for improved results. Besides, the generalization and portability to other problems could be validated.

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Sheng-guang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.
- Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan L Yuille, Trevor Darrell, Jitendra

- Malik, and Alexei A Efros. 2024. Sequential modeling enables scalable learning for large vision models. In *CVPR*.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *NeurIPS*.
- Alaaeldin El-Nouby, Michal Klein, Shuangfei Zhai, Miguel Angel Bautista, Alexander Toshev, Vaishaal Shankar, Joshua M Susskind, and Armand Joulin. 2024. Scalable pre-training of large autoregressive image models. *arXiv preprint arXiv:2401.08541*.
- Patrick Esser, Robin Rombach, and Bjorn Ommer. 2021. Taming transformers for high-resolution image synthesis. In *CVPR*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021a. Aligning ai with shared human values. In *ICLR*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021b. Measuring massive multitask language understanding. In *ICLR*.
- Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.
- Jari Korhonen and Junyong You. 2012. Peak signal-to-noise ratio revisited: Is simple beautiful? In *2012 Fourth international workshop on quality of multimedia experience*, pages 37–38. IEEE.
- Xuantong Liu, Shaozhe Hao, Xianbiao Qi, Tianyang Hu, Jun Wang, Rong Xiao, and Yuan Yao. 2024. Elucidating the design space of language models for image generation. *arXiv preprint arXiv:2410.16257*.
- Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning-based text classification: a comprehensive review. *ACM computing surveys (CSUR)*, 54(3):1–40.
- Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *NeurIPS*.
- Guohao Sun, Can Qin, Yihao Feng, Zeyuan Chen, Ran Xu, Sohail Dianat, Majid Rabbani, Raghuveer Rao, and Zhiqiang Tao. 2025. Structured policy optimization: Enhance large vision-language model via self-referenced dialogue. In *ICCV*.
- Guohao Sun, Can Qin, Huazhu Fu, Linwei Wang, and Zhiqiang Tao. 2024a. Stllava-med: Self-training large language and vision assistant for medical question-answering. In *EMNLP*.
- Guohao Sun, Can Qin, Jiamian Wang, Zeyuan Chen, Ran Xu, and Zhiqiang Tao. 2024b. Sq-llava: Self-questioning for large vision-language assistant. In *ECCV*.
- Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. 2024c. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.
- Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. 2024. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *NeurIPS*.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.
- Jiamian Wang, Kunpeng Li, Yulun Zhang, Xin Yuan, and Zhiqiang Tao. 2025. S^A -2-transformer for mask-aware hyperspectral image reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Jiamian Wang, Huan Wang, Yulun Zhang, Yun Fu, and Zhiqiang Tao. 2023. Iterative soft shrinkage learning for efficient image super-resolution. In *ICCV*.
- Jiamian Wang, Zongliang Wu, Yulun Zhang, Xin Yuan, Tao Lin, and Zhiqiang Tao. 2024. Cooperative hardware-prompt learning for snapshot compressive imaging. In *NeurIPS*.
- Jiamian Wang, Yulun Zhang, Xin Yuan, Ziyi Meng, and Zhiqiang Tao. 2022. Modeling mask uncertainty in hyperspectral image reconstruction. In *ECCV*.
- Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612.

Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. 2024a. Randomized autoregressive visual generation. *arXiv preprint arXiv:2411.00776*.

Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. 2024b. [Randomized autoregressive visual generation](#). *ArXiv*, abs/2411.00776.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*.

Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. 2023. What makes good examples for visual in-context learning? In *NeurIPS*.

A Appendix

Metrics. On colorization and inpainting, we employ the Fréchet inception distance (FID) (Heusel et al., 2017), Inception Score (IS) (Salimans et al., 2016), Perplexity, and LPIPS (Zhang et al., 2018) to measure the generation quality, and incorporate PSNR (Korhonen and You, 2012) (dB) and SSIM (Wang et al., 2004) to assess the image quality comparing to ground-truth image. On edge detection, we measure the performance in terms of detection accuracy and recall.

UVDv1 Dataset. As introduced in Section 4, we adopt the UVDv1 dataset curated by LVM (Bai et al., 2024), which covers images from over 50 large-scale datasets including ImageNet, COCO, and LAION, etc. and inherits the multi-source nature. We split 12K image pairs for training and 1K for inference in each task. Notably, the dataset avoids randomness and ensures stability for two reasons. (1) We adopt a fixed split of image pairs for training across all experiments, ensuring the consistency and stability. (2) UVDv1 dataset covers a wide range of visual variations and ensuring that the initial selection in the split does not affect the reliability of the training.

More Details on LVM. LVM (Bai et al., 2024) fine tunes the VQGAN tokenizer for image encoding and reconstruction. For visual in-context learning, LVM generates the image token sequence y_q for the given query image token sequence x_q depending on tasks defined by the given K demonstration pairs, *i.e.*, $\{x_i, y_i\}_{i=1}^K$, where each pair consists of a RGB image token sequence x_i and a sketch image token sequence y_i (exampled by edge detection). For each token sequence (e.g., $\{x_i, y_i, x_q, y_q\}$), there are 256 tokens set by LVM. Owing to this property, LVM readily enables diverse image translation tasks or generation tasks.

More Details on VQGAN Tokenizer. In VQGAN tokenizer (Esser et al., 2021), each image is divided into non-overlapping patches (e.g., for LVM), and each patch is encoded into a latent vector by the pretrained VQGAN encoder. These vectors are then quantized by mapping to a discrete token. Specifically, A token refers to a discrete index value. These tokens are good representations as (1) they can reflect meaningful visual patterns like edges, textures, or object parts by capturing the local visual structures. (2) Their discrete nature allows autoregressive modeling, facilitating the image generation.

More Details on Decoding. In the baseline method of LVM, each image will be encoded into 256 discrete tokens. Besides, the “codebook” refers to the embedding matrix provided by the token embedding layer $f_{\text{embed}}(\cdot)$, which defines the mapping between discrete tokens and their corresponding embedding vectors. The embedding layer comes from the pretrained LVM model and is kept fixed throughout the training, as indicated by the “frozen” symbol in Fig. 1. It is also used during generation as shown in Fig. 1 (lower right block). Based on the codebook, the continuous refined embedding is mapped back to the discrete tokens via nearest-neighbor lookup.

More Technical Details of the Self-Refinement. The proposed method jointly adjusts the discrete tokens in the latent space. To achieve this, we firstly map the generated discrete tokens to the embedding space for fine-tuning and map the embedding back to the discrete token space for generation (inference). Specifically, as shown in Fig. 1, during training, the loss is computed purely under the embedding space. No nearest embedding lookup is needed. Differently, during inference, it involves finding the nearest embedding in the codebook to reconstruct the image.

The proposed self-refinement network $g_\phi(\cdot)$ adopts a Transformer architecture with multi-head self-attention and residual connections. Each block consists of (1) MultiHead-Attention + Add & LayerNorm and (2) Feedforward Network (FFN) + Add & LayerNorm. The full refinement network stacks multiple such blocks (e.g., #head= 8, #blocks= 1). This structure enables the model to capture global dependencies across generated token embeddings and the joint refinement. In Table 7, we discuss different network structures of the proposed self-refinement module, including self-attention, MLP, and 1D convolutional structure. All variants preserve residual structure and comparable parameter scale. Self-attention significantly outperforms both MLP and CNN. MLP does not bring the performance boost for LVM baseline, due to its token-wise nature. Convolutional network enables local structure modeling and improve over MLP but remain less effective than self-attention.

More Details on Image Generation. We choose VAR-d16 as the baseline model. The experiments are conducted on 2 A6000 GPUs. Table 2 shows that the proposed self refinement module enables further performance boost compared to pretrained VAR after training for 10K iterations.

Algorithm 1 Training with Refinement Network

- 1: **Input:** Context image pairs $\mathbf{x}_1, \mathbf{y}_1, \dots, \mathbf{x}_K, \mathbf{y}_K$, query image $\mathbf{x}_q$, ground truth label image $\mathbf{x}_{GT}$
 - 2: **Modules:** VQGAN tokenizer T , AR model $f_{AR}(\cdot)$, refinement network $g_\phi(\cdot)$, loss function $\mathcal{L}$
 - 3: **Output:** Refined token embeddings $\mathbf{e}'_{seq}$
 - 4: Encode all images into token sequences: $Input\ Ids \leftarrow T.encode([\mathbf{x}_1, \mathbf{y}_1, \dots, \mathbf{x}_K, \mathbf{y}_K, \mathbf{x}_q])$
 - 5: Generate output tokens with AR model: $\mathbf{y}_q \leftarrow f_{AR}(Input\ Ids)$
 - 6: Embed predicted tokens: $\mathbf{e} \leftarrow f_{embed}(\mathbf{y}_q)$
 - 7: Refine embeddings: $\mathbf{e}' \leftarrow g_\phi(\mathbf{e})$
 - 8: Compute target embeddings: $\mathbf{e}^* \leftarrow f_{embed}(T.encode(\mathbf{x}_{GT}))$
 - 9: Compute loss: $\mathcal{L} \leftarrow \text{CosineDistance}(\mathbf{e}', \mathbf{e}^*)$
 - 10: Backpropagate to update parameters of g_ϕ only
-

Table 6: Distance measuring discussion for self-refinement network learning, *e.g.*, upon colorization and inpainting.

Distance	Colorization						Inpainting					
	Perp	LPIPS	FID	IS	PSNR	SSIM	Perp	LPIPS	FID	IS	PSNR	SSIM
Cosine	19.01	0.28	59.24	27.34	18.78	0.5086	82.63	0.27	59.60	26.71	18.40	0.4508
L_2	20.06	0.30	59.75	27.24	17.89	0.4922	176.00	0.31	64.05	25.40	17.24	0.4337

Context Length Discussion. In Figure 3, we provide a detailed comparison of our self-refinement method (red) against the ICL baseline (yellow) across various metrics on colorization task as the context image length increases. The results demonstrate the consistent superiority of our approach. As shown in first row in Figure 3, our method consistently achieves significantly lower perplexity, LPIPS, and FID across nearly all context image lengths, demonstrating more confident predictions and better fidelity as well as alignment with real-world image distributions. While for IS, PSNR and SSIM, our self-refinement method maintains higher metric score compared with ICL baseline. We also observe that increasing the context image length does not always lead to improved performance, a trend that aligns with similar findings in LVM (Bai et al., 2024).

More Ablation Studies. In Table 6, we compare different similarity metrics to be used for learning the refinement network $g_\phi(\cdot)$. As shown by both colorization and inpainting, cosine distance enables promising performance and can be better than L_2 distance. This is because during decoding, VQGAN model adopts a cosine similarity as the distance measuring to identify the closest element in codebook for the given input. Accordingly, bridging the embedding with the cosine distance benefits the token identification.

Social Impacts. Like other generative models, the proposed method could be misused to create misleading or harmful content if applied irrespon-

sibly. However, our focus is on improving controllability and quality in image generation, which can benefit creative and commercial applications.

Metric	LVM	+MLP	+Convolutional network	+Self-Attention (proposed)
Perplexity ↓	20.06	19.70	19.26	19.01
LPIPS ↓	0.29	0.30	0.29	0.28
FID ↓	59.70	60.95	59.40	59.24
IS ↑	27.20	26.50	27.00	27.34
PSNR ↑	17.93	17.90	18.30	18.78
SSIM ↑	0.4925	0.4800	0.5001	0.5086

Table 7: Ablation study of refinement network structure on colorization.

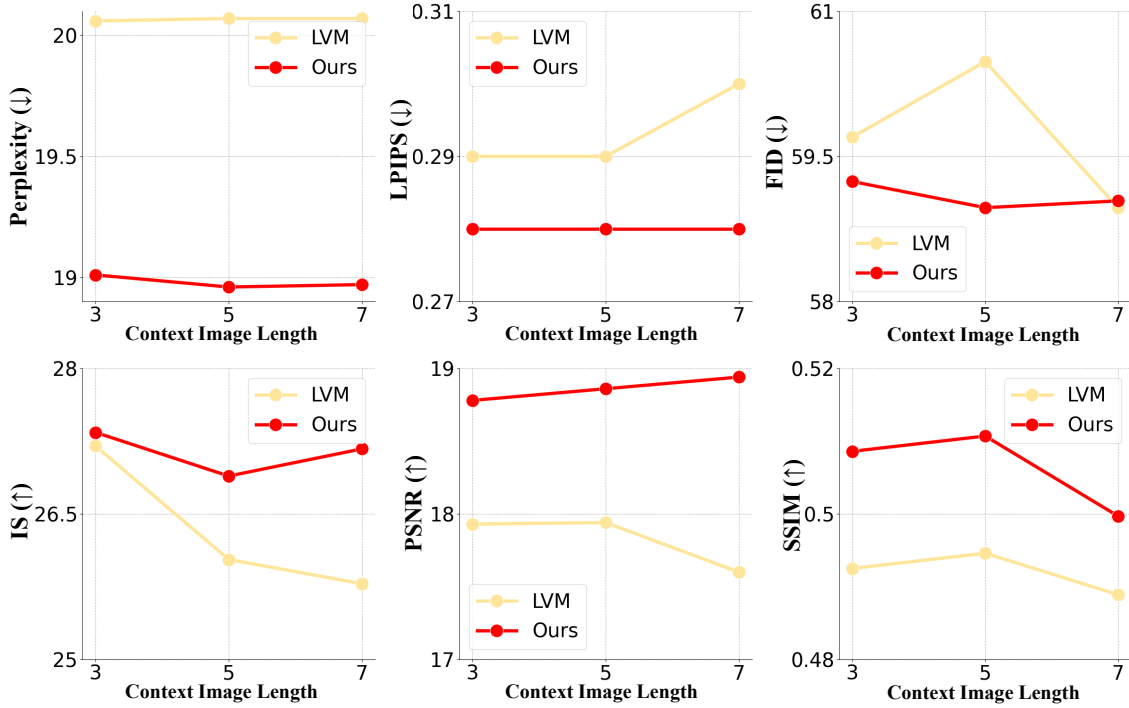


Figure 3: Effect of demonstration image length for the proposed self-refinement (e.g., upon colorization). The input image sequence is composed of one or more image pairs plus one query RGB image. The proposed method enables notable performance boost under different context lengths, demonstrating the robustness to the context length.