Improving Autoregressive Visual Generation with Cluster-Oriented Token Prediction (Supplementary Material)

Teng Hu[1*], Jiangning Zhang[2,3*], Ran Yi[1†], Jieyu Weng[1], Yabiao Wang[3,2], Xianfang Zeng[3], Zhucun Xue[3], Lizhuang Ma[1]
[1]Shanghai Jiao Tong University, [2]Youtu Lab, Tencent, [3]Zhejiang University
{hu-teng, ranyi, w.jerry, lzma}@sjtu.edu.cn,
{186368, yabiaowang, zzlongjuanfeng, 12432038}@zju.edu.cn

A. Overview

The supplementary material is composed of:

- Implementation details (Sec. B);
- More details on optimization relaxation in codebook rearrangement (Sec. C)
- Comparison with VAR [14] (Sec. D)
- More analysis on our model (Sec. E);
- Analysis on the cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$ (Sec. F);
- Analysis on the influence of code-rearrangement quality (Sec. G);
- Experiments on different VQVAEs (Sec. H);
- More visualization results (Sec. I);
- Future work (Sec. J).

B. Implementation Details

Metrics. We employ four metrics to evaluate the effectiveness of the models:

- **Fréchet inception distance (FID)** [4] measures the similarity between the features of the source data and the generated data according to their mean values and covariance. A smaller FID indicates better generation ability.
- **Inception Score (IS)** [12] measures the quality and diversity of images by computing the information entropy of the generated images. A higher IS indicates better generation quality and diversity.
- **Precision/Recall** [8] measures the class-conditional generation accuracy. A higher precision or recall indicates a better class-conditional generation performance.

Experiment settings. We follow the experiment settings as LlamaGen [13] and keep the hyperparameters consistent with it. The experiment details are shown in Tab. A1 and Tab. A2, where Tab. A2 is the inference settings for Tab. 2 of the main paper.

Sampling hyperparameters: Among the hyperparameters used in the inference process (Tab. A2), there are several important parameters, whose meanings are explained in detail below:

(1) Classifier-free guidance: Classifier-Free Guidance (CFG) [5] is originally a sampling method to improve diffusion models by combining conditional and unconditional score estimates. Beyond diffusion models, CFG can also be applied to the autoregressive image generation process [13]. Denoting the input image token sequence as q, our model as ϵ_θ, and the class condition as c, the autoregressive CFG is defined as:

$$\tilde{\epsilon}_\theta(q, c) = (1 + w)\epsilon_\theta(q, c) - w\epsilon_\theta(q, \phi), \qquad \text{(A1)}$$

where ϕ denotes the empty condition and $\epsilon(,)$ represents the predicted probability distribution for the next image token.

(2) Top-K: Top-K sampling [11] is a decoding strategy that selects tokens from the top k highest-probability candidates. It focuses on the most likely tokens, but the fixed k size may exclude important low-probability options.

(3) Top-P: Top-P sampling [6], also known as nucleus sampling, selects tokens dynamically from the smallest set whose cumulative probability meets or exceeds a threshold p. This approach adapts to the output distribution, balancing coherence and diversity in text generation.

(4) Temperature: In large language models (LLMs), temperature [1, 9] is a hyperparameter that controls the randomness of the generated token by adjusting the sharpness of the probability distribution: lower values make the output more deterministic, while higher values increase diversity and randomness. The probability P_i for each token is calculated as:

$$P_i = \frac{\exp(l_i/T)}{\sum_j \exp(l_j/T)},$$

*Equal contribution.
†Corresponding author.

where l_i represents the predicted probability distribution, and T is the temperature.

C. Complexity Analysis of Codebook Rearrangement Target

In Sec. 3.2 of the main paper, we aim to rearrange the codebook such that the neighboring embeddings are as close to each other. We summarize this code rearrangement problem as an optimization problem, where we aim to find a surjective mapping $M(\cdot)$ that satisfies:

$$M = \arg\min_{M} \sum_{i=1}^{N-1} \|z_{M(i)}, z_{M(i+1)}\|. \tag{A2}$$

After reordering each embedding z_i to index $M(i)$, the sum of distances between adjacent embeddings is minimized. And $\hat{\mathcal{Z}} = M(\mathcal{Z})$ is the rearranged codebook.

However, this optimization can be reduced to the Shortest Hamiltonian path problem, which is a classical NP-hard problem. The Shortest Hamiltonian Path Problem is a variation of the Hamiltonian Path Problem. Its goal is to find a path that visits each vertex exactly once and minimizes the total weight (or distance) of the path. Formally, given a weighted graph $G = (V, E)$ with a weight function $w : E \to \mathbb{R}^+$, the goal is to find a Hamiltonian path $\pi^* = (\pi_1, \pi_2, \ldots, \pi_N)$ such that the sum of the weights of the edges in the path, *i.e.*, $\sum_{i=1}^{N-1} w(\pi_i, \pi_{i+1})$, is minimized, which is formulated as:

$$\pi^* = \arg\min_{\pi} \sum_{i=1}^{N-1} w(\pi_i, \pi_{i+1}). \tag{A3}$$

Next, we prove that solving the optimization problem in Eq. (A2) can be reduced to the Shortest Hamiltonian path problem in Eq. (A3):

Proposition: Solving Eq. (A3) $\leq_p$ Solving Eq. (A2)

Proof.

Step 1: Construct a Complete Weighted Graph:

Define a complete graph $G = (V, E)$ where the vertex set $V = \{0, 1, \ldots, N-1\}$ corresponds to the N embeddings in the codebook $\mathcal{Z}$. Each edge $(i, j) \in E$ is assigned a weight $w(i, j)$ that equals to the distance between embeddings z_i and z_j:

$$w(i, j) = \|z_i, z_j\|. \tag{A4}$$

Step 2: Find the Minimum Weight Hamiltonian Path:

Finding the shortest Hamiltonian path $\pi = (\pi_1, \pi_2, \ldots, \pi_N)$ in G aims to minimize the total weight:

$$\pi^* = \arg\min_{\pi} \sum_{i=1}^{N-1} w(\pi_i, \pi_{i+1}). \tag{A5}$$

Step 3: Mapping to the Original Problem:

The shortest Hamiltonian path π^* provides the optimal permutation M^* for the optimization problem in Eq. (A2), where $M^*(i) = \pi_i$, for $i = 1, 2, \ldots, N$.

In summary, the original optimization problem (in Eq. (A2)) of finding the optimal surjective mapping $M(\cdot)$ to minimize the sum of distances between consecutive embeddings, can be reduced to finding the minimum weight Hamiltonian path in a complete weighted graph, where the weights are given by the distances between embeddings.

Therefore, the original optimization problem is also NP-hard. And it is necessary to relax this optimization target to a clustering problem (main paper Sec. 3.2), which ensures the embeddings with a cluster share high similarities.

D. Comparing IAR+VAR with VAR

VAR [14] extends the next-token prediction in autoregressive image generation to next-scale prediction, enabling the model to generate images progressively from small to large scales. At each scale, VAR predicts all tokens simultaneously, significantly enhancing the inference speed of the autoregressive image generation process. Our design is independent of the model structure in autoregressive image generation, allowing us to integrate our IAR with VAR, referred to as VAR+IAR. Given that most official VAR models are trained on 256 A100 GPUs, which is highly resource-intensive, we only train the VAR-d16 model for 100 epochs on ImageNet [2] and subsequently compare it with VAR+IAR.

Both models (VAR and VAR+IAR) are trained for 100 epochs with a batch size of 768, maintaining the same hyperparameters as the official VAR code. We then evaluate the trained models on different CFGs. The results, presented in Tab. A3, demonstrate that incorporating IAR into VAR enhances the original VAR in terms of generation quality and diversity, as evidenced by improved FID and IS scores. This validates the effectiveness of our model across different autoregressive image generation frameworks, showing the great potential of our IAR in the field of autoregressive image generation.

E. More Analysis on Our Model

Comprehensive metrics for models under different CFGs. This section presents the comprehensive metrics (FID, IS, Precision, Recall) for the models compared in Fig. 4 (a) of the main paper. As shown in Table A4, an increase in CFG leads to higher IS and precision, while recall decreases. Unlike these three metrics, FID initially improves and then deteriorates, achieving its optimal value at an intermediate CFG. Furthermore, the optimal CFG for FID varies with model size (e.g., CFG=2.25 for IAR-B and CFG=1.75 for IAR-L).

Model	B	L	XL	XXL	B	L	XL	XXL
Parameter Num	111M	343M	775M	1.4B	111M	343M	775M	1.4B
Token Num	16×16				24×24			
Optimizer	AdamW							
Weight decay	0.05							
Learing Rate Scheduler	Constant							
Batch Size	256	256	256	256	256	256	256	512
Learning Rate	1E-04	1E-04	2E-04	2E-04	1E-04	1E-04	2E-04	2E-04
GPU Num	8	8	8	8	8	8	16	32
Epoch	300	300	50	50	300	300	300	300
FSDP	No	No	No	Yes	No	No	No	Yes

Table A1. The training settings and hyperparameters used in our model.

Model	B	L	XL	XXL	B	L	XL	XXL
Parameter Num	111M	343M	775M	1.4B	111M	343M	775M	1.4B
Token Num	16×16				24×24			
Batch Size	32							
Random Seed	0							
Top K	0							
Top P	1.0							
Temperature	1.0							
CFG	2.0	2.0	1.75	2.0	2.25	1.75	1.75	1.65

Table A2. The inference settings and hyperparameters used in Tab. 2 of the main paper.

Comprehensive metrics for models under different training epochs. Table A5 presents a comparison between our IAR and LlamaGen [13] over various training epochs, illustrating that our model consistently outperforms LlamaGen at all stages of training. Notably, the 200-epoch IAR-B exceeds the performance of the 300-epoch LlamaGen-B, while the 200-epoch IAR-L performs similarly to the 300-epoch LlamaGen-L, highlighting the high training efficiency of our model. (Note that all B-version models use CFG=2.25, whereas all L-version models use CFG=1.75)

Training losses for different model sizes. We show the training loss curves for both the two losses: 1) cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$ and the token-oriented cross-entropy loss $\mathcal{L}_{TCE}$ when training on 24×24 image tokens (Fig. A1) and 16×16 image tokens (Fig. A2). It can be seen that as the model size increases, both the two losses decrease faster and converge to a lower value, which aligns with the scaling law [7]. Note that since we follow the training setting of LlamaGen [13], we only train IAR-XL and IAR-XXL on 16×16 image tokens for 50 epochs.

Effectiveness of $\mathcal{L}_{\mathbf{CCE}}$. In the main paper, we introduce the cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$, designed to enhance the model's awareness of cluster information, thereby increasing the likelihood of predicting tokens within the target cluster. It is hard to directly illustrate the effectiveness of $\mathcal{L}_{CCE}$ by its loss value directly. Therefore, we design an alternative way where we visualize the loss curves for token-oriented cross-entropy loss $\mathcal{L}_{TCE}$ and their corresponding FIDs for LlamaGen-B and our model in Fig. A3. The results indicate that, compared to LlamaGen, our model exhibits a higher token-oriented cross-entropy loss but achieves a superior FID. This suggests that our model has slightly lower token-oriented prediction accuracy, which is expected since the introduction of $\mathcal{L}_{CCE}$ partially diverts the original loss $\mathcal{L}_{TCE}$. Therefore, the improvement of FID can only come from our proposed cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$. Since $\mathcal{L}_{CCE}$ effectively increases the likelihood of predicting the correct cluster, combined with the embedding similarities within the cluster, it ultimately leads to the generation of images with better FID, demonstrating the efficacy of $\mathcal{L}_{CCE}$ in our model.

Token prediction accuracy. We compute the token prediction accuracy of our model and LlamaGen [13] on different model sizes (24×24 tokens). Specifically, for an image token sequence $q = \{q^1, q^2, \cdots q^{576}\}$ with corresponding image embedding sequence $z_q = \{z_q^1, z_q^2, \cdots z_q^{576}\}$ and class label c, we enumerate i from 1 to 575 and predict

Classifier-free Guidance	VAR-d16 + **IAR** FID↓	IS↑	Precision↑	Recall↑	VAR-d16 FID↓	IS↑	Precision↑	Recall↑
1.5	**4.12**	**58.15**	**0.839**	**0.482**	4.28	56.66	0.830	0.479
1.75	**4.07**	**60.54**	**0.857**	0.458	4.25	59.00	0.846	**0.460**
2.0	**4.43**	**63.11**	**0.865**	**0.435**	4.52	61.00	0.860	0.435

Table A3. Comparing VAR-d16 [14] with VAR+IAR on ImageNet [2]. It shows that our IAR also performs well in the next-scale prediction model, validating that our method can be widely applied to various autoregressive image generation models, enhancing their generative capabilities.

Classifier-free Guidance	IAR-B FID↓	IS↑	Precision↑	Recall↑	IAR-L FID↓	IS↑	Precision↑	Recall↑
1	29.70	43.96	0.566	**0.632**	20.56	62.96	0.595	**0.666**
1.5	10.69	103.59	0.732	0.532	4.39	178.78	0.778	0.566
1.75	7.43	135.55	0.783	0.501	**3.18**	234.79	0.824	0.530
2	6.06	165.22	0.822	0.454	3.49	279.09	0.855	0.499
2.25	**5.77**	192.45	0.850	0.421	4.43	311.08	0.873	0.466
2.5	6.11	213.76	0.869	0.381	5.61	340.18	0.890	0.425
2.75	6.73	**232.35**	**0.884**	0.360	6.74	**358.48**	**0.898**	0.401

Table A4. The Quantitative metrics on our model under different classifier-free guidance scales.

$\hat{q}^{i+1} \sim P^{i+1} = \epsilon_\theta(\hat{q}^{i+1}|c, q^1, q^2, \cdots q^i)$ using the model ϵ_θ. We then compute the Top-1 and Top-5 accuracy Acc^i between $\hat{q}^{i+1}$ and the ground truth q^{i+1}. The average accuracy for an image is calculated as $Acc = \frac{1}{575}\sum_{i=1}^{575} Acc^i$. Finally, we compute the cluster-level accuracy and token-level accuracy for all images in ImageNet [2] and record the average values in Tab. A6. Specifically, to compute the cluster-level accuracy for LlamaGen, we employ the balanced K-means clustering algorithm to decompose the codebook into n clusters and then determine the target cluster index. We then assess whether the predicted token is located in the target cluster, thereby obtaining the cluster-level accuracy. From Tab. A6, it can be seen that our cluster-level accuracy is higher than that of LlamaGen, indicating the effectiveness of our cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$. Although our token-level accuracy is slightly lower, this is expected as the newly included loss $\mathcal{L}_{CCE}$ affects the original token-oriented cross-entropy loss $\mathcal{L}_{TCE}$, resulting in a slight decrease in token-level accuracy. However, our model still achieves better FID and IS compared to LlamaGen, further validating the effectiveness of our cluster-oriented token prediction strategy.

F. Analysis on the Cluster-oriented Cross-entropy Loss

In Tab. 3 of the main paper, the model trained with only the cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$ (without codebook rearrangement) also improves generation performance (FID 6.96 vs. 7.14 for the baseline). This improvement arises because $\mathcal{L}_{CCE}$ enhances the probability of predicting the correct cluster, which is computed based on the probabilities of all tokens within the cluster. Minimizing $\mathcal{L}_{CCE}$ consequently increases the probability of the target token. Furthermore, since $\mathcal{L}_{CCE}$ boosts the probabilities of all tokens in the target cluster, it can be viewed as a variant of label smoothing, which is known to improve generalization and model calibration in classification networks [10]. However, as shown in Table A7, standard label smoothing is not well-suited for autoregressive visual generation models, where token prediction accuracy is typically low (e.g., 2%–4% top-1 accuracy, as shown in Table A6), in contrast to traditional classification tasks with significantly higher accuracy (e.g., $> 70\%$). In contrast, $\mathcal{L}_{CCE}$ performs structured smoothing within specific ranges rather than uniformly smoothing all tokens, leading to improved AR generation quality. While $\mathcal{L}_{CCE}$ enhances model performance, the best results are achieved only when it is combined with our codebook rearrangement strategy. Therefore, the good performance of our IAR is attributed to both the codebook rearrangement strategy and the cluster-oriented cross-entropy loss.

G. Analysis on the Influence of Code-rearrangement Quality

Our IAR employs a code-rearrangement strategy to cluster similar codes, with the hypothesis that better clustering quality theoretically enhances generation performance. To validate this, we average the mean distance of each cluster as a metric to evaluate clustering quality, where a lower mean distance indicates a higher codebook rearrangement quality. We train LlamaGen-B on codebooks with varying clustering qualities (different mean distances, obtained by

Model Size	Epoch	LlamaGen				**IAR**			
		FID↓	IS↑	Precision↑	Recall↑	FID↓	IS↑	Precision↑	Recall↑
B Version	50	8.67	136.62	0.818	**0.413**	**7.80**	**153.31**	**0.839**	0.394
	100	7.26	152.50	0.827	0.416	**6.77**	**171.73**	**0.839**	**0.416**
	200	6.54	167.82	0.833	0.428	**5.86**	**185.28**	**0.845**	**0.428**
	300	6.09	182.54	0.845	0.416	**5.77**	**192.45**	**0.850**	**0.421**
L Version	50	4.25	191.46	0.819	0.504	**4.35**	**197.23**	**0.819**	**0.507**
	100	3.96	199.96	0.803	**0.532**	**3.81**	**205.63**	**0.805**	0.528
	200	3.33	219.57	0.804	0.538	**3.31**	**225.95**	**0.814**	**0.551**
	300	3.29	227.83	0.818	**0.532**	**3.18**	**234.79**	**0.824**	0.530

Table A5. The Quantitative metrics on our model and LlamaGen under different training epochs. The B version employs CFG=2.25 and the L-version employs CFG=1.75.

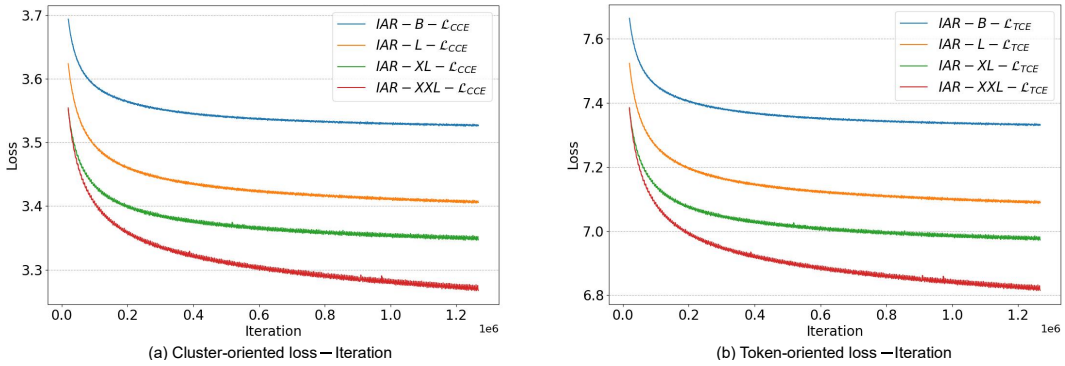

Figure A1. The training loss curves for the cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$ (a) and token-oriented cross-entropy loss $\mathcal{L}_{TCE}$ (b) on $\mathbf{24 \times 24}$ **image tokens**.

setting different clustering iterations in Balanced k-means Clustering) for 100 epochs to investigate how clustering quality affects performance. Results in Table A8 demonstrate that improved clustering quality leads to better model generation performance.

H. Experiments on Different VQVAEs

To show the effectiveness and generalization ability of our method, we further conduct experiments on the VQVAE from VQGAN [3], where we train LlamaGen-B on it for 100 epochs. As shown in Tab. A9, our IAR consistently improves the generation quality for the model based on VQGAN, demonstrating the effectiveness of our IAR across different VQVAEs.

We further compute the mean / closest / largest $\mathcal{L}_2$ distance in each cluster for the rearranged and original codebook from LlamaGen and VQGAN, to validate the effectiveness of our code rearrangement strategy. The results in Tab. A10 show that our method stably decreases the three distances. Moreover, it can be seen that the closest distance in VQGAN is 5×10^{-7}, indicating that some embeddings are almost the same, resulting in some embeddings being wasted (e.g., the closest distance of VQGAN (5e-7) is very low, causing the worst reconstruction PSNR (20.00)). Therefore, it motivates us that the closest distance can be used as a metric to evaluate the training of a VQVAE.

I. More Visualization Results

We exhibit more generated images from our model in Fig. A4∼A7, where the images are generated by The XXL-version with 4.0 CFG, with image size 384×384. We show 16 classes of images, including alp, promontory, volcano, coral reef, sports car, balloon, convertible, space shuttle, castle, church, beacon, cinema, bridge, ocean liner, white stork, and Pomeranian.

J. Future Work

The main idea of our IAR is to ensure a high similarity between the predicted image embedding and the target embedding, so that even if the model incorrectly predicts the target token, the output image still closely resembles the target image. This can be naturally considered as a continuous constraint on the image embedding, aiming to minimize the distance between the predicted and target image embeddings. However, this approach cannot be easily applied to LLM-based image generation models due to

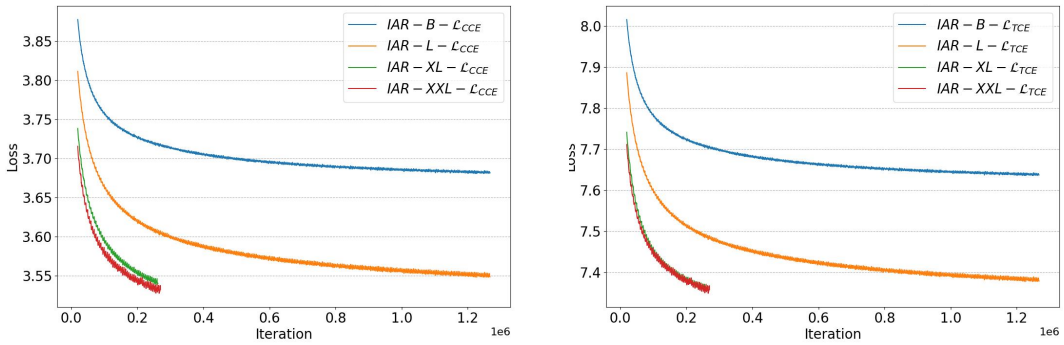

Figure A2. The training loss curves for the cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$ (a) and token-oriented cross-entropy loss $\mathcal{L}_{TCE}$ (b) on $\mathbf{16 \times 16}$ **image tokens**.

Model	IAR				LlamaGen [13]			
	B	L	XL	XXL	B	L	XL	XXL
Cluster-level Accuracy (Top-1, %)	**15.54**	**17.12**	**18.01**	**19.02**	13.44	14.81	15.71	16.49
Cluster-level Accuracy (Top-5, %)	**41.48**	**44.68**	**46.30**	**48.29**	30.37	33.13	34.87	36.38
Token-level Accuracy (Top-1, %)	2.62	3.17	3.56	3.88	**2.64**	**3.19**	**3.59**	**3.95**
Token-level Accuracy (Top-5, %)	7.34	8.86	9.90	10.75	**7.37**	**8.91**	**9.98**	**10.96**

Table A6. Comparison of the token-level prediction accuracy, cluster-level prediction accuracy, and the embedding-level MSE distance between our IAR and LlamaGen.

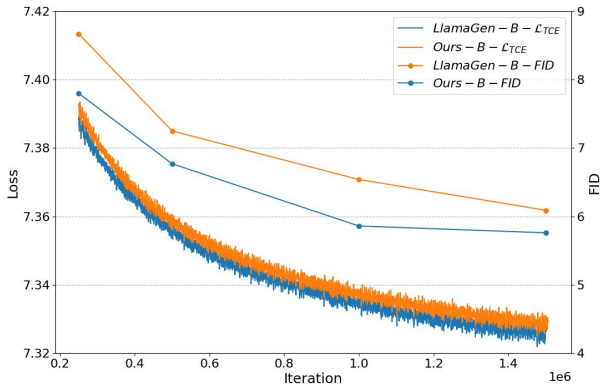

Figure A3. Comparison between LlamaGen-B and ours on the **token-oriented cross-entropy loss** $\mathcal{L}_{\mathbf{TCE}}$ and the **FID** score in different training iterations. Our model has a higher $\mathcal{L}_{TCE}$ than that of LlamaGen but achieves a better FID, indicating the effectiveness of our cluster-oriented cross-entropy loss $\mathcal{L}_{CCE}$.

Model	FID↓	IS↑	Precision↑	Recall↑
LlamaGen	6.05	182.5	0.84	0.42
LlamaGen + IAR	**5.77**	**192.5**	**0.85**	**0.42**
LlamaGen + LS	8.58	166.81	0.80	0.40

Table A7. Comparison with the model with label smoothing, where we train LlamaGen-B with default smoothing factor 0.1 for 300 epochs. It shows that the vanilla label smoothing is not suitable for AR generation models with low token prediction accuracy.

Mean Dis	FID↓	IS↑	Precision↑	Recall↑
0.689	**6.77**	**171.73**	**0.84**	**0.42**
0.850	6.85	170.54	0.84	0.41
1.239	6.91	169.32	0.84	0.41

Table A8. Generation performance from codebooks with different clustering quality.

the non-differentiable nature of the embedding quantization operation. Therefore, this paper relaxes the problem into a cluster-oriented token prediction problem, which can be easily integrated into the current autoregressive image generation model. We believe that in future work, employing this continuous constraint in autoregressive image generation may further enhance the model performance.

References

[1] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 1985. 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 4

[3] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming

VQVAE	IAR	FID↓	IS↑	Precision↑	Recall↑
LlamaGen		7.14	166.38	0.84	0.40
LlamaGen	✓	**6.77**	**171.73**	**0.84**	**0.42**
VQGAN		6.90	176.71	0.84	0.40
VQGAN	✓	**6.75**	**194.64**	**0.85**	**0.40**

Table A9. Experiments on different VQVAEs. Our IAR consistently improves the performance of the model trained on different VQVQEs.

VQVAE	IAR	Mean↓	Closest↓	Largest↓	PSNR ↑
LlamaGen		2.06	0.20	4.30	20.79
LlamaGen	✓	**0.69**	**0.18**	**1.91**	20.79
VQGAN		24.75	5e-7	1072.06	20.00
VQGAN	✓	**8.69**	**5e-7**	**44.68**	20.00

Table A10. Comparison between the original and rearranged codebooks from Llamagen and VQGAN. Our code rearrangement strategy can consistently improve the inner-cluster similarity for both the two codebooks.

transformers for high-resolution image synthesis. In *CVPR*, 2021. 5

[4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 1

[5] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 1

[6] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *ICLR*, 2019. 1

[7] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 3

[8] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *NeurIPS*, 2019. 1

[9] Enrique Manjavacas, Folgert Karsdorp, Ben Burtenshaw, and Mike Kestemont. Synthetic literature: Writing science fiction in a co-creative process. In *CCNLG*, 2017. 1

[10] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019. 4

[11] Andrea Pietracaprina, Matteo Riondato, Eli Upfal, and Fabio Vandin. Mining top-k frequent itemsets through progressive sampling. *DATAMINE*, 2010. 1

[12] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, 2016. 1

[13] 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. 1, 3, 6

[14] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *NeurIPS*, 2024. 1, 2, 4

Figure A4. The generated images for alp, promontory, volcano, and coral reef by IAR-XXL with 4.0 CFG.

Figure A5. The generated images for sports car, balloon, convertible, and space shuttle by IAR-XXL with 4.0 CFG.

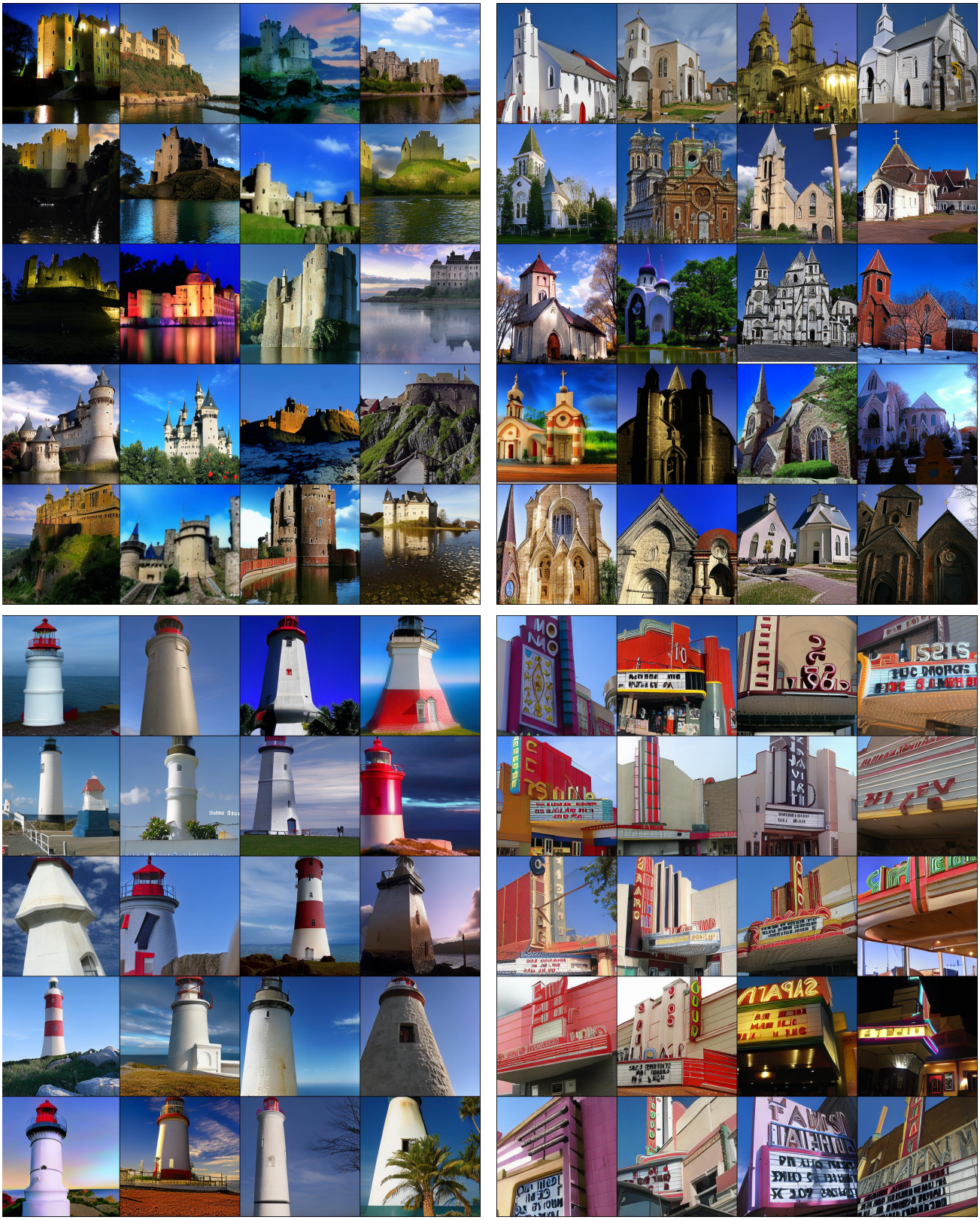

Figure A6. The generated images for castle, church, beacon, and cinema by IAR-XXL with 4.0 CFG.

Figure A7. The generated images for bridge, ocean liner, white stork, and Pomeranian by IAR-XXL with 4.0 CFG.