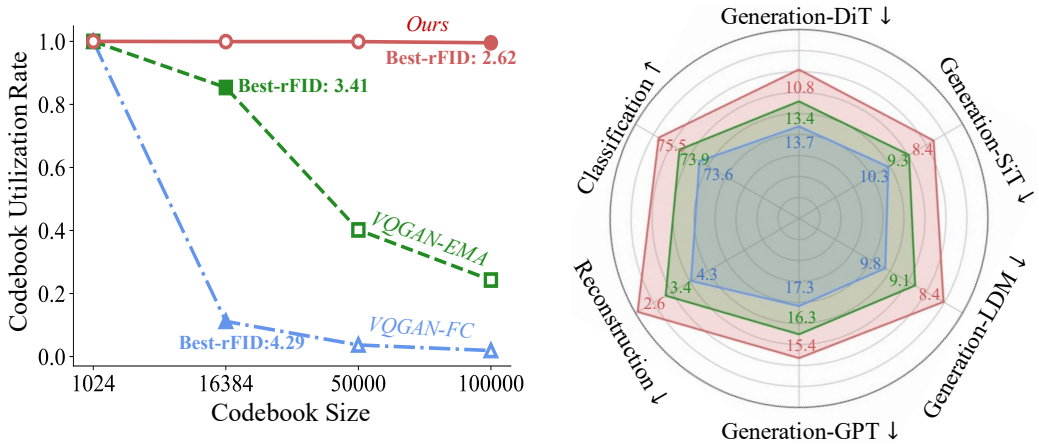
Scaling the Codebook Size of VQGAN to 100,000 with a Utilization Rate of 99%

Lei Zhu[1] **Fangyun Wei**[2*] **Yanye Lu**[1] **Dong Chen**[2]
[1]Peking University [2]Microsoft Research Asia
zhulei@stu.pku.edu.cn fawe@microsoft.com yanye.lu@pku.edu.cn doch@microsoft.com



(a) Codebook size v.s. utilization rate. (b) Evaluation on downstream tasks.

Figure 1: (a) Two enhanced versions of VQGAN [1], namely VQGAN-FC (Factorized Codes) and VQGAN-EMA (Exponential Moving Average), experience a decline in codebook utilization rate and performance as their codebook sizes expand. In contrast, our method, VQGAN-LC (Large Codebook), effectively leverages an extremely large codebook, persistently maintaining a utilization rate of up to 99% and achieving higher performance. We highlight the best reconstruction rFID for each model. (b) Comparison among three models across various tasks. For image generation, we evaluate the applications of these three VQGAN variants to GPT [2], LDM [3], DiT [4] and SiT [5].

Abstract

In the realm of image quantization exemplified by VQGAN, the process encodes images into discrete tokens drawn from a codebook with a predefined size. Recent advancements, particularly with LLAMA 3, reveal that enlarging the codebook significantly enhances model performance. However, VQGAN and its derivatives, such as VQGAN-FC (Factorized Codes) and VQGAN-EMA, continue to grapple with challenges related to expanding the codebook size and enhancing codebook utilization. For instance, VQGAN-FC is restricted to learning a codebook with a maximum size of 16,384, maintaining a typically low utilization rate of less than 12% on ImageNet. In this work, we propose a novel image quantization model named VQGAN-LC (Large Codebook), which extends the codebook size to 100,000, achieving an utilization rate exceeding 99%. Unlike previous methods that optimize each codebook entry, our approach begins with a codebook initialized with 100,000 features extracted by a pre-trained vision encoder. Optimization then focuses on training a projector that aligns the entire codebook with the feature distributions of the encoder in VQGAN-LC. We demonstrate the superior performance of our model over its counterparts across a variety of tasks, including image reconstruction, image classification, auto-regressive image generation using GPT, and image creation with diffusion- and flow-based generative models.

*Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Table 1: We conduct a comparative analysis of our VQGAN-LC against two advanced variants of VQGAN [1], namely VQGAN-FC and VQGAN-EMA, focusing on the effects of enlarging their codebook sizes from 1,024 to 100K. The only difference among the three models lies in the initialization and optimization of the codebook. The evaluation covers both reconstruction and generation using the latent diffusion model (LDM) [3] on the ImageNet dataset.

Method	Reconstruction (rFID)				Generation with LDM [3] (FID)			
	1,024	16,384	50K	100K	1,024	16,384	50K	100K
VQGAN-FC	4.82	**4.29**	4.96	4.65	10.81	**9.78**	10.37	10.12
VQGAN-EMA	4.93	**3.41**	3.88	3.46	10.16	**9.13**	9.29	9.50
VQGAN-LC (Ours)	4.97	3.01	2.75	**2.62**	9.93	8.84	8.61	**8.36**

1 Introduction

Image quantization [1, 6, 7] refers to the process of encoding an image into a set of discrete representations, also known as image tokens, each derived from a codebook of a pre-defined size. VQGAN [1] stands out as a prominent architecture, with an encoder-quantizer-decoder structure, playing a pivotal role in various applications, including: (1) training a GPT [2, 8, 9, 10, 11] on image tokens to create images; (2) serving as an autoencoder in latent diffusion models (LDMs) [3, 4] and generative models [12, 13, 14], with flow matching [5, 15]; and (3) functioning within large multi-modality models [16, 17, 18, 19], where its encoder processes input images and its decoder assists in image generation.

In contrast to natural languages, which typically maintain a static vocabulary, image quantization models necessitate a codebook of a pre-defined size to convert images into discrete image tokens. The nature of image signals—complex and continuous—makes translating images into token maps a form of lossy compression that is generally more severe than converting them into continuous feature maps. The capability of these models to represent images largely depends on the codebook size. Previous studies, such as VQGAN [1], its improved versions, including VQGAN with exponential moving average (EMA) update (VQGAN-EMA) and VQGAN using factorized codes (VQGAN-FC), and its predecessors, like VQVAE [6] and VQVAE-2 [7], have demonstrated that they can only learn a codebook with a maximum size of 16,384. These models often face unstable training or performance saturation issues when the codebook size is further increased, as shown in Table 1. Additionally, they typically exhibit a low codebook utilization rate—for instance, under 12% in VQGAN-FC, as shown in Figure 1(a)—indicating that a significant portion of the codebook remains unused, thereby diminishing the model's representational capacity. Furthermore, studies on large language models suggest that employing a tokenizer with an expanded vocabulary significantly enhances model efficacy. For example, the technical report for LLAMA 3[2] shows, "LLAMA 3 uses a tokenizer with a vocabulary of 128K tokens that encodes language much more efficiently, which leads to substantially improved model performance."

In this study, we investigate the scalability of codebook size in VQGAN and the improvement of its codebook utilization rate, thereby substantially enhancing the representational capabilities of VQGAN. Typically, as shown in Figure 2(a), image quantization models like VQGAN are structured with an encoder-quantizer-decoder architecture, where the quantizer is connected to a codebook. For a given image, the encoder produces a feature map that the quantizer then converts into a token map. Each token in this map corresponds to an entry in the codebook, based on their cosine similarity. This token map is subsequently used by the decoder to reconstruct the original image.

Generally, the codebook in VQGAN begins with a *random* initialization. Each entry (a.k.a. a token embedding) in the codebook is designated as *trainable* and undergoes optimization through either gradient descent [6, 1, 20, 21] (Figure 2(b)) or an exponential moving average (EMA) update [7, 22] (Figure 2(c)) during the training phase. Nevertheless, in each iteration, only a small amount of token embeddings, corresponding to the token maps of the current training batch, are optimized. As training progresses, these frequently optimized token embeddings gradually align more closely with the distributions of the feature maps generated by the encoder, compared to those less frequently or never optimized (referred to as inactive token embeddings). Consequently, these inactive token embeddings

[2] `https://ai.meta.com/blog/meta-llama-3/`

are excluded from the training process and subsequently remain unused during the inference phase, resulting in poor codebook utilization.

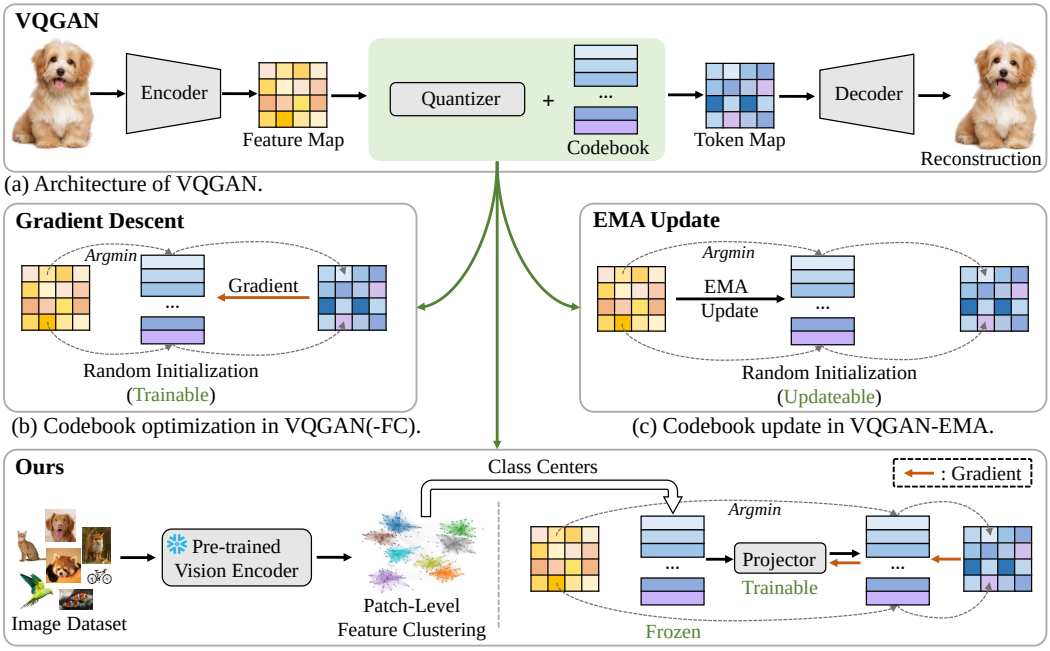
Our approach deviates from conventional image quantization models by initiating with a codebook composed of N *frozen* features, sourced from a pretrained image backbone like the CLIP-vision-encoder [23], and utilizing datasets like ImageNet [24]. A projector is then employed to transition the entire codebook into a latent space, producing token embeddings. During the training process, it is the projector that is optimized, not the codebook itself, which distinguishes our method from traditional models. By optimizing the projector, we adapt the aggregate distribution of the codebook entries to align with the feature maps generated by the encoder. This contrasts with methods like VQGAN [1], where adaptations are made to a limited number of codebook entries to match the feature map distributions during each iteration. Our simple quantization technique ensures that almost all token embeddings (over 99%) remain active throughout the training phase. The process is depicted in Figure 2(d).

Our newly developed quantizer can be integrated directly into the existing VQGAN architecture, replacing its standard quantizer without requiring any changes to the encoder and decoder. This innovative quantizer enables the expansion of the codebook to sizes up to 100,000, while maintaining an impressive utilization rate of 99%. By comparison, the conventional VQGAN is limited to a codebook size of 16,384 with a utilization rate of only 11.2% when applied to ImageNet. The advantages of a larger codebook with enhanced utilization are demonstrated across various applications, including image reconstruction, image classification, auto-regressive image generation using GPT, and image creation with diffusion models and flow matching. Figure 1 illustrates the performance of our improved VQGAN, termed VQGAN-LC (Large Codebook), compared to its counterparts.

2 Related Work

Image Quantization. Image quantization focuses on compressing an image into discrete tokens derived from a codebook [6, 1, 22, 20, 25, 26, 27]. VQVAE [6] introduces a method of quantizing patch-level features using the nearest codebook entry, with the codebook learned jointly with the encoder-decoder structure through reconstruction loss. VQVAE2 [7] enhances this by employing exponential moving average updates and a multi-scale hierarchical structure to improve quantization performance. VQGAN [1] further refines VQVAE by integrating adversarial and perceptual losses, enabling more accurate and detailed representations. ViT-VQGAN [21] replaces the CNN-based encoder-decoder [28] with a vision transformer and shows that using a factorized code mechanism with l_2 regularization can improve codebook utilization. Reg-VQ [29] introduces prior distribution regularization to prevent collapse and low codebook utilization. Additionally, some approaches use codebook reset strategies to reset unused codebook entries during training [26, 30, 31, 32] or utilizing stochastic quantization to enhance utilization rates [26, 29]. In contrast to these methods, the proposed VQGAN-LC initializes codebook entries using a pre-trained vision encoder on target datasets, ensuring nearly full codebook utilization throughout the training process and allowing for scaling up the codebook size to more than 100K.

Tokenized Image Synthesis. Substantial advancements have been achieved in the realm of image synthesis, particularly through image quantization techniques. Initial efforts such as PixelRNN [33] employs LSTM networks [34] to autoregressively model dependencies between quantized pixels. Building upon this, the groundbreaking VQVAE [6] introduces the quantization of image patches into discrete tokens, significantly enhancing generation capabilities when paired with PixelCNN [35]. The iGPT [36] further advances the field by leveraging the powerful Transformer [37] for sequence modeling of VQ-VAE tokens. Recently, there has been a shift towards using non-autoregressive Transformers for image synthesis [12, 13, 14, 11], which provide efficiency improvements over traditional raster-scan-based generation methods. Innovative approaches such as discrete diffusion models, including D3PMs [38] and VQ-Diffusion [39], utilize discrete diffusion processes to model the distribution of image tokens. Additional diffusion-based techniques [3, 15, 4, 5, 40, 41] compress images into latent representations using quantizers, thereby reducing both training and inference costs. Moreover, image quantizers can enhance large language models for both image synthesis and understanding [17, 16, 18, 42, 19]. Our work introduces a superior image quantizer, further refining the image synthesis process.



(a) Architecture of VQGAN.

(b) Codebook optimization in VQGAN(-FC).

(c) Codebook update in VQGAN-EMA.

(d) Codebook initialization and quantization process in our VQGAN-LC.

Figure 2: (a) The encoder-quantizer-decoder structure of VQGAN, with a codebook linked to the quantizer. (b) The codebook optimization strategy employed in VQGAN and VQGAN-FC. (c) The codebook update mechanism utilized in VQGAN-EMA. (d) The codebook initialization and quantization process implemented in our VQGAN-LC.

3 Method

3.1 Preliminary

VQGAN. Let $\mathcal{B} = \{\boldsymbol{b}_n \in \mathbb{R}^D\}_{n=1}^N$ denote a codebook containing N entries, with each entry $\boldsymbol{b}_i$ being a D-dimensional *trainable* embedding with *random* initialization. As shown in Figure 2(a), VQ-GAN adopts an encoder-quantizer-decoder structure. In this setup, an encoder processes an image $\boldsymbol{X}$ of height H and width W to generate a feature map $\boldsymbol{Z} \in \mathbb{R}^{h \times w \times D}$, with (h, w) representing the latent dimensions. Subsequently, the quantizer maps $\boldsymbol{Z}$ to a token map $\hat{\boldsymbol{Z}}$, where each token in $\hat{\boldsymbol{Z}}$ is an entry in $\mathcal{B}$ based on the cosine distance between $\boldsymbol{Z}$ and $\mathcal{B}$. Finally, the decoder reconstructs the original image from the token map $\hat{\boldsymbol{Z}}$. The entire network is optimized using a combination of losses, expressed as follows:

$$\mathcal{L} = \underbrace{|||\hat{\boldsymbol{X}} - \boldsymbol{X}||^2}_{\mathcal{L}_R} + \underbrace{\alpha||sg(\hat{\boldsymbol{Z}}) - \boldsymbol{Z}|| + \beta||sg(\boldsymbol{Z}) - \hat{\boldsymbol{Z}}||}_{\mathcal{L}_Q} + \mathcal{L}_P + \mathcal{L}_{GAN}, \tag{1}$$

where $sg(\cdot)$ denotes the stop-gradient operation. The terms $\mathcal{L}_R$, $\mathcal{L}_Q$, $\mathcal{L}_P$ and $\mathcal{L}_{GAN}$ represent the reconstruction loss, quantization loss, VGG-based perceptual loss [1], and GAN loss [1], respectively. Hyper-parameters α and β are set to 1.0 and 0.33 by default. As shown in Figure 2(b), we refer to the codebook optimization strategy used in the original VQGAN as "gradient descent".

VQGAN-FC. VQGAN faces significant challenges with inefficient codebook utilization. To address this issue, the factorized code (FC) mechanism, initially proposed by ViT-VQGAN [21], is employed. We refer to VQGAN integrated with the FC mechanism as VQGAN-FC. The key differences between VQGAN and VQGAN-FC are two-fold: 1) a linear layer is added to project the encoder feature $\boldsymbol{Z} \in \mathbb{R}^{h \times w \times D}$ into a low-dimensional feature $\boldsymbol{Z'} \in \mathbb{R}^{h \times w \times D'}$, where $D' \ll D$; 2) the codebook $\mathcal{B}$, consisting of N D'-dimensional trainable embeddings, is randomly initialized. Consequently, the

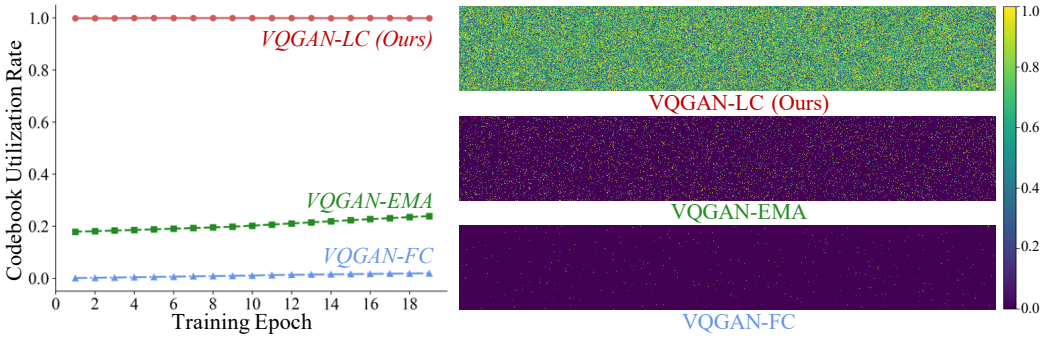


Figure 3: (Left) The codebook utilization rate over the training epoch. A codebook entry is considered utilized for the epoch if it is used at least once. (Right) The average utilization frequency of each codebook entry over all epochs, with each pixel representing a single entry. All models adopt a codebook with a size of 100K and use images with a resolution of 256×256 on ImageNet.

quantization loss in Eq. 1 is reformulated as:

$$\mathcal{L}_Q = \alpha||sg(\hat{\boldsymbol{Z}}) - \boldsymbol{Z'}|| + \beta||sg(\boldsymbol{Z'}) - \hat{\boldsymbol{Z}}||. \tag{2}$$

However, as illustrated in Figure 1(a), the utilization rate of VQGAN-FC is only 11.2% on ImageNet when the codebook size is configured to 16,384, and increasing the size of the codebook fails to enhance performance as demonstrated in Table 1.

VQGAN-EMA. As depicted in Figure 2(c), this variant of VQGAN adopts an exponential moving average (EMA) strategy to optimize the codebook. Specifically, Let $\hat{\mathcal{B}} \subset \mathcal{B}$ denote the set of token embeddings used for all token maps in the current training batch. The set $\hat{\mathcal{B}}$ is updated through the EMA mechanism using the corresponding encoder features $\boldsymbol{Z}$ in each iteration. As a result, the codebook does not receive any gradients. Therefore, the quantization loss in Eq. 1 is defined as:

$$\mathcal{L}_Q = \alpha||sg(\hat{\boldsymbol{Z}}) - \boldsymbol{Z}||. \tag{3}$$

Our results, highlighted in Figure 1, indicate that VQGAN-EMA outperforms VQGAN-FC on various downstream tasks, leading to enhanced utilization of the codebook. However, expanding the codebook size continues to pose a significant challenge for VQGAN-EMA, as detailed in Table 1.

3.2 VQGAN-LC

Analysis of VQGAN-FC and VQGAN-EMA. In these enhanced versions of VQGAN, the codebook is initialized randomly. During each iteration, only a small subset of entries related to the current training batch are optimized. As a result, the frequently optimized entries become more aligned with the feature map distributions generated by the encoder, while the less frequently optimized entries remain underutilized. Consequently, a significant portion of the codebook remains unused during both the training and inference stages. Figure 3 shows the codebook utilization rate over the training epoch and visualizes the utilization frequency of each codebook entry once training is completed.

Overview. We present VQGAN-LC (Large Codebook), which allows for the expansion of the codebook to sizes of up to 100,000 while achieving a remarkable utilization rate of 99%. As illustrated in Figure 2(d), our method diverges from VQGAN-FC and VQGAN-EMA in its design of the quantizer. We maintain a static codebook and train a projector to map the entire codebook into a latent space, aligning the distributions of the feature maps generated by the encoder. This approach allows us to scale the codebook size effectively without modifying the encoder and decoder, achieving an extremely high utilization rate and resulting in superior performance across various tasks, as shown in Figure 1, Table 1 and Figure 3. It is important to note that *increasing the codebook size incurs almost no additional computational cost.*

Codebook Initialization. To initialize a static codebook, we first utilize a pre-trained vision encoder (e.g., CLIP with a ViT backbone) to extract patch-level features from the target dataset (e.g., ImageNet) containing M images. This extraction results in a set of features denoted as $\mathcal{F} = \{\boldsymbol{F}_m^{(i,j)} \in$

$\mathbb{R}^D\}_{i=1,j=1,m=1}^{\bar{h},\bar{w},M}$, where $\boldsymbol{F}_m^{(i,j)}$ represents a D-dimensional patch-level feature at location (i,j) in the m-th image, and $(\bar{h},\bar{w})$ indicate the spatial dimensions of $\boldsymbol{F}$. Subsequently, we apply K-means clustering to $\mathcal{F}$, resulting in N cluster centers (with a default value of $N = 100,000$). These cluster centers form the set $\mathcal{C} = \{\boldsymbol{c}_n \in \mathbb{R}^D\}_{n=1}^N$, where $\boldsymbol{c}_n$ is the n-th center. Our codebook $\mathcal{B}$ is then initialized using $\mathcal{C}$.

Quantization. Unlike VQGAN, VQGAN-FC and VQGAN-EMA, which optimize the codebook directly, our approach involves training a projector $P(\cdot)$, implemented as a simple linear layer, to align the static codebook $\mathcal{B}$ with the feature distributions generated by the encoder $E(\cdot)$ of our VQGAN-LC. Let $\mathcal{B}' = P(\mathcal{B}) = \{\boldsymbol{b}'_n \in \mathbb{R}^{D'}\}_{n=1}^N$ denote the projected codebook. For a given input image $\boldsymbol{X}$, the quantizer transforms the feature map $\boldsymbol{Z} = E(\boldsymbol{X}) \in \mathbb{R}^{h\times w\times D'}$ into a token map $\hat{\boldsymbol{Z}}$. This quantization process can be expressed as $\hat{\boldsymbol{Z}} := \underset{\boldsymbol{b}'_n\in\mathcal{B}'}{\operatorname{argmin}}||\boldsymbol{Z}^{(i,j)} - \boldsymbol{b}'_n||$.

Loss Function. We employ the same loss function as specified in Eq.1. However, the key distinction is that our codebook $\mathcal{B}$ remains frozen, while the newly introduced projector $P(\cdot)$ undergoes optimization.

3.3 Evaluation of Image Quantization Models

We evaluate the performance of VQGAN-FC, VQGAN-EMA, and our proposed VQGAN-LC across image reconstruction, image classification and image generation tasks.

Image Reconstruction. Images are processed through the encoder, quantizer, and decoder to produce reconstructed images. These reconstructed images are then compared to their original images using the rFID metric as the evaluation criterion.

Image Classification. Initially, the encoder and quantizer convert each image into a token map. Subsequently, we utilize a ViT-B model [37], pre-trained with MAE [43], to train on all token maps for the purpose of image classification. Top-1 accuracy is used as the evaluation metric.

Image Generation. Image quantization models can be integrated with different image generation frameworks, such as auto-regressive causal Transformers (GPT [2]), latent diffusion models (LDM [3]), diffusion Transformers (DiT [4]), and flow-based generative models (SiT [5]), to facilitate image creation.

GPT. The encoder and quantizer transform each image into a token map $\hat{\boldsymbol{Z}}$, which is then flattened into a token sequence. Ultimately, GPT is trained on the collection of these token sequences.

LDM. It progressively adds noise onto the encoder feature $\boldsymbol{Z}$. The training objective is to denoise and reconstruct $\boldsymbol{Z}$. During the inference phase, the output from LDM is inputted into the quantizer and decoder of image quantization models to generate images.

DiT. This model is a variant of LDM, distinguished by its use of a Transformer architecture as the backbone. The incorporation of image quantization models into DiT follows the same approach as their integration into LDM.

SiT. This method presents a flow-based generative framework utilizing the DiT backbone. The integration of image quantization models in SiT follows the same methodology as in LDM and DiT.

4 Experiments

4.1 Setup

Implementation Details of Image Quantization. All image quantization models, including VQGAN, VQGAN-FC, VQGAN-EMA, and our proposed VQGAN-LC, utilize the same encoder and decoder of the original VQGAN. The input images are processed at a resolution of 256×256 pixels. The encoder (U-Net [28]) downsamples the input image by a factor of 16, yielding a feature map $\boldsymbol{Z}$ with dimensions of 16×16. The quantizer then converts this feature map into a token map $\hat{\boldsymbol{Z}}$ of the same size, which is subsequently fed into the decoder (U-Net) for image reconstruction. In our observations, the optimal codebook size for VQGAN, VQGAN-FC, and VQGAN-EMA is 16,384, whereas for our VQGAN-LC, the optimal codebook size is 100,000. Training is conducted on the

Table 2: Reconstruction performance on ImageNet-1K. The term "# Tokens" refers to the number of tokens used to represent an image. The codebook utilization rate is computed across all training images. The FC and EMA mechanisms are originally introduced by ViT-VQGAN [21] and VQVAE [6, 7], respectively. It is important to note that *increasing the codebook size incurs almost no additional computational cost.*

Method	# Tokens	Codebook Size	Utilization (%)	rFID	LPIPS	PSNR	SSIM
DQVAE [20]	256	1,024	-	4.08	-	-	-
DF-VQGAN [46]	256	12,288	-	5.16	-	-	-
DiVAE [47]	256	16,384	-	4.07	-	-	-
RQVAE [22]	256	16,384	-	3.20	-	-	-
RQVAE [22]	512	16,384	-	2.69	-	-	-
RQVAE [22]	1,024	16,384	-	1.83	-	-	-
DF-VQGAN [46]	1,024	8,192	-	1.38	-	-	-
VQGAN [1]	256	16,384	3.4	5.96	0.17	23.3	52.4
	256	50,000	1.1	5.44	0.17	22.5	52.5
	256	100,000	0.5	5.44	0.17	22.3	52.5
VQGAN-FC [21]	256	16,384	11.2	4.29	0.17	22.8	54.5
	256	50,000	3.6	4.96	0.15	23.1	54.7
	256	100,000	1.9	4.65	0.15	22.9	55.1
VQGAN-EMA [7]	256	16,384	83.2	3.41	0.14	23.5	56.6
	256	50,000	40.2	3.88	0.14	23.2	55.9
	256	100,000	24.2	3.46	0.13	23.4	56.2
VQGAN-LC (Ours)	256	16,384	**99.9**	3.01	0.13	23.2	56.4
	256	50,000	**99.9**	2.75	0.13	23.8	58.4
	256	100,000	**99.9**	2.62	0.12	23.8	58.9
	1,024	100,000	99.5	**1.29**	**0.07**	**27.0**	**71.6**

ImageNet-1K [24] and FFHQ [44] datasets, utilizing 32 Nvidia V100 GPUs. For ImageNet-1K, we train for 20 epochs, whereas for FFHQ, we train for 800 epochs. The Adam optimizer [45] is used, starting with an initial learning rate of $5e^{-4}$. This learning rate follows a half-cycle cosine decay schedule after a linear warm-up phase of 5 epochs.

Codebook Initialization of Our VQGAN-LC. Unless otherwise specified, we use the CLIP image encoder [23] with a ViT-L/14 backbone, adding an additional 4×4 average pooling layer, to extract patch-level features from images in the training split of the target dataset (either ImageNet or FFHQ). These features are then clustered into N groups using the K-Means algorithm with CUDA acceleration. The cluster centers constitute the codebook. By default, N is configured to 100,000. We specify the codebook entries to have a dimension of 8.

Image Generation Models. For LDM [3], DiT [4] and SiT [5], we adopt their original architectures. For generation using GPT [2], we follow VQGAN [1], using a causal Transformer decoder with 24 layers, 16 heads per attention layer, a latent dimension of 1,024 and a total of 404M parameters. For ImageNet, we employ class-conditional generation, whereas for FFHQ, we use unconditional generation. In LDM, DiT, and SiT, classifier-free guidance [3] is implemented for class-conditional generation. More implementation details can be found in Section A.

Evaluation. In the image reconstruction task, we evaluate performance using rFID, LPIPS, PSNR, and SSIM metrics on the validation sets of ImageNet and FFHQ. For image classification, we measure the top-1 accuracy on ImageNet. For image generation, we calculate the FID score on ImageNet using 50K generated images compared against the ImageNet training set. For FFHQ, the FID score is determined using 50K generated images in comparison with the combined training and validation sets of FFHQ. The codebook utilization rate is also reported for comparison, which is calculated as the ratio of active entries (tokens/codes) to the total size of the codebook.

4.2 Main Results

Image Reconstruction. Tables 2 and 3 present the reconstruction performance for ImageNet and FFHQ, respectively. We make three key observations: 1) Our method consistently achieves a

Table 3: Reconstruction performance on FFHQ.

Method	# Tokens	Codebook Size	Utilization (%)	rFID	LPIPS	PSNR	SSIM
RQVAE [22]	256	2,048	-	7.04	0.13	22.9	67.0
VQWAE [48]	256	1,024	-	4.20	0.12	22.5	66.5
MQVAE [49]	256	1,024	78.2	4.55	-	-	-
VQGAN [1]	256	16,384	2.3	5.25	0.12	24.4	63.3
VQGAN-FC [21]	256	16,384	10.9	4.86	0.11	24.8	64.6
VQGAN-EMA [7]	256	16,384	68.2	4.79	0.10	25.4	66.1
VQGAN-LC (Ours)	256	100,000	**99.5**	**3.81**	**0.08**	**26.1**	**69.4**

Table 4: Image generation on ImageNet-1K.

Method	# Tokens	Codebook Size	Utilization (%)	FID
RQTransformer *(GPT-480M)* [22]	256	16,384	-	15.7
ViT-VQGAN *(GPT-650M)* [21]	256	8,192	-	11.2
DQTransformer *(GPT-355M)* [20]	640	1,024	-	7.34
DQTransformer *(GPT-655M)* [20]	640	1,024	-	5.11
ViT-VQGAN *(GPT-650M)* [21]	1,024	8,192	-	8.81
Stackformer *(GPT-651M)* [49]	1,024	1,024	-	6.04
LDM [3]	1,024	16,384	-	8.11
with GPT-404M [2]				
VQGAN-FC [21]	256	16,384	11.2	17.3
VQGAN-EMA [7]	256	16,384	83.1	16.3
VQGAN-LC (Ours)	256	16,384	99.9	16.1
VQGAN-LC (Ours)	256	100,000	97.0	15.4
with SiT-XL [5]				
VQGAN-FC [21]	256	16,384	11.2	10.3
VQGAN-EMA [7]	256	16,384	83.1	9.31
VQGAN-LC (Ours)	256	16,384	99.9	9.06
VQGAN-LC (Ours)	256	100,000	99.6	8.40
with DiT-XL [4]				
VQGAN-FC [21]	256	16,384	11.2	13.7
VQGAN-EMA [7]	256	16,384	85.3	13.4
VQGAN-LC (Ours)	256	16,384	99.9	11.2
VQGAN-LC (Ours)	256	100,000	99.4	10.8
with LDM [3]				
VQGAN-FC [21]	256	16,384	11.2	9.78
VQGAN-EMA [7]	256	16,384	83.1	9.13
VQGAN-LC (Ours)	256	16,384	99.9	8.36
VQGAN-LC (Ours)	256	100,000	99.4	8.36
VQGAN-LC (Ours)	1,024	100,000	99.4	**4.81**

codebook utilization rate of over 99% across all codebook sizes on both datasets. 2) The reconstruction performance improves consistently with the scaling of codebook size using our method. 3) Increasing the codebook size (e.g., VQGAN-LC with a codebook size of 100,000 and 256 tokens), and the number of tokens to represent an image (e.g., RQVAE with a codebook size of 16,384 and 512 tokens) both enhance performance, with the former introducing almost no additional computational cost compared to the latter.

Image Generation. Table 4 shows the results of class-conditional image generation on ImageNet. All models (GPT, LDM, DiT, and SiT) demonstrate improved performance with the integration of our VQGAN-LC, regardless of their underlying architectures, which include auto-regressive causal Transformers, diffusion models, diffusion models with Transformer backbones, and flow-based generative models. The diversity of the generated images increases due to the utilization of a large codebook, which has a size of up to 100,000 and a utilization rate exceeding 99%. Table 5 displays the unconditional generation results on the FFHQ dataset. Notably, DiT and SiT, which use the Transformer architecture, require more extensive training data for optimizing diffusion- and flow-

Table 5: Image generation on FFHQ.

Method	# Tokens	Codebook Size	Utilization (%)	FID
Stackformer *(GPT-307M)* [49]	256	1,024	-	7.67
DQTransformer *(GPT-308M)* [20]	640	1,024	-	4.91
Stackformer *(GPT-307M)*[49]	1,024	1,024	-	6.84
Stackformer *(GPT-651M)* [49]	1,024	1,024	-	5.67
ViT-VQGAN *(GPT-650M)* [21]	1,024	8,192	-	3.13
LDM [3]	4,096	8,192	-	4.98
with LDM [3]				
VQGAN-FC	256	16,384	11.2	13.2
VQGAN-EMA	256	16,384	68.2	12.5
VQGAN-LC (Ours)	256	100,000	99.7	12.3
with GPT (404M) [2]				
VQGAN-FC	256	16,384	10.9	3.23
VQGAN-EMA	256	16,384	68.2	4.87
VQGAN-LC (Ours)	256	100,000	99.1	**2.61**

Table 6: Ablation study of using various codebook initialization strategies on ImageNet.

Strategy	Dataset	Model	Utilization (%)	rFID	LPIPS	PSNR	SSIM
Random Initialization	-	-	5.4	108.7	0.46	18.2	36.4
Random Selection	ImageNet	ViT-L	99.8	2.95	**0.12**	**23.8**	**58.9**
K-Means Clutering	ImageNet	ResNet-50	99.8	2.71	**0.12**	23.7	58.3
K-Means Clutering	ImageNet	ViT-B	**99.9**	2.70	**0.12**	**23.8**	58.7
K-Means Clutering	ImageNet	ViT-L	**99.9**	**2.62**	**0.12**	**23.8**	**58.9**

based generative models. Given that FFHQ is significantly smaller than ImageNet, we limit our training on FFHQ to GPT and LDM.

Image Classification. In Section 3.3, we discuss the training of an image classifier on a dataset containing tokenized images. We fine-tune three ViT-B [37] models, pre-trained by MAE [43], using the tokenized images produced by the top-performing VQGAN-FC, VQGAN-EMA, and our proposed VQGAN-LC, on ImageNet. Both VQGAN-FC and VQGAN-EMA demonstrate optimal reconstruction performance when utilizing a codebook with 16,384 entries. As illustrated in Figure 1(b), our method achieves a top-1 accuracy of 75.7 on ImageNet, surpassing VQGAN-FC and VQGAN-EMA by margins of 1.6 and 1.9, respectively.

Visualizations. Section C presents images generated by GPT [2], LDM [3], DiT [4], and SiT [5], incorporating our VQGAN-LC.

4.3 Ablation Studies

Unless otherwise specified, we evaluate reconstruction performance on ImageNet across all studies.

Codebook Initialization. In Section 3.2, we describe the default codebook initialization approach. This involves using a CLIP vision encoder with a ViT-L backbone to extract patch-level features from ImageNet, followed by a K-means clustering algorithm to generate N cluster centers, resulting in the static codebook $\mathcal{B}$. In Table 6, we evaluate two factors: 1) employing a CLIP vision encoder with different backbones (e.g., ViT-B [37] and ResNet-50 [50]) to extract patch-level features; and 2) utilizing non-clustering strategies to initialize the static codebook, including random initialization and random selection, where N features are randomly chosen from all patch-level features of ImageNet. Our findings are threefold: 1) random initialization leads to extremely poor performance since the codebook remains frozen in our VQGAN-LC; 2) using a CLIP vision encoder with a ViT-L backbone outperforms those using ViT-B and ResNet-50 backbones; and 3) K-means clustering produces a more robust codebook.

Codebook Size. In Table 7, we incrementally increase the codebook size in our VQGAN-LC from 1,000 to 200,000. The performance shows minimal improvements beyond a codebook size of

Table 7: Ablation study of using different codebook sizes on ImageNet.

Codebook Size	Utilization (%)	rFID	LPIPS	PSNR	SSIM
1,000	**100.0**	4.98	0.17	22.9	55.3
10,000	99.8	3.80	0.14	23.3	57.2
50,000	99.9	2.75	0.13	23.8	58.4
100,000	99.9	**2.62**	**0.12**	23.8	58.9
200,000	99.8	2.66	**0.12**	**23.9**	**59.2**

Table 8: The computational cost of VQGAN-LC with different codebook sizes.

Codebook Size	MACs	Model Size
16,384	195.08G	71.71M
100,000	195.70G	71.72M

Table 9: Ablation study on codebook transferability. The term "source dataset→target dataset" indicates that the codebook is initialized using the source dataset, while our VQGAN-LC is trained on the target dataset.

Setting	Utilization (%)	rFID	LPIPS	PSNR	SSIM
FFHQ→FFHQ	**99.5**	**3.81**	**0.08**	**26.1**	**69.4**
ImageNet→FFHQ	99.4	4.08	**0.08**	**26.1**	**69.4**
ImageNet→ImageNet	**99.9**	**2.62**	**0.12**	**23.8**	**58.9**
FFHQ→ImageNet	**99.9**	2.91	**0.12**	**23.8**	**58.9**

100,000—specifically, only 0.01 PSNR and 0.03 SSIM gains are observed, when the codebook size reaches 200,000. Therefore, we consistently use a codebook size of 100,000 in all experiments. The codebook utilization rate consistently exceeds 99% across all configurations.

Computational Cost. The primary inference cost of VQ-GAN is attributed to the encoder and decoder. Increasing the codebook size N incurs minimal additional cost since the matrix multiplication between F and B is negligible compared to the encoder and decoder processing time. Table 8, we present the multiply-accumulates (MACs) and model sizes for our VQGAN-LC models with codebook sizes of 16,384 and 100,000, respectively, when inferring an image of size 256×256.

Codebook Transferability. In our standard setup, both the codebook initialization and the VQGAN-LC training are conducted using the same dataset, either ImageNet or FFHQ. In Table 9, we examine the transferability of the codebook by initializing it with one dataset and training our VQGAN-LC on a different dataset. Our findings indicate that our method exhibits significant codebook transferability, highlighting the robustness of our codebook initialization process.

5 Conclusion

In this work, we introduce a novel image quantization model, VQGAN-LC, which extends the codebook size to 100,000, achieving a utilization rate exceeding 99%. Our approach significantly outperforms prior models like VQGAN, VQGAN-FC and VQGAN-EMA, across various tasks including image reconstruction, image classification and image synthesis using numerous generation models such as auto-regressive causal Transformers (GPT), latent diffusion models (LDM), diffusion Transformers (DiT), and flow-based generative models (SiT), while incurring almost no additional costs. Extensive experiments on ImageNet and FFHQ verify the effectiveness of our approach.

6 Acknowledgements

This work was supported in part by the Natural Science Foundation of China under Grant 623B2001, Grant 62394311, and Grant 82371112; in part by Beijing Natural Science Foundation under Grant Z210008; and in part by the High-Grade, Precision and Advanced University Discipline Construction Project of Beijing under Grant BMU2024GJJXK004.

References

[1] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[4] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[5] 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. *arXiv preprint arXiv:2401.08740*, 2024.

[6] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[7] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[8] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[10] OpenAI. Gpt-4 technical report, 2023.

[11] Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. Sequential modeling enables scalable learning for large vision models. *arXiv preprint arXiv:2312.00785*, 2023.

[12] 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.

[13] 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.

[14] 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.

[15] Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

[16] Lijun Yu, Yong Cheng, Zhiruo Wang, Vivek Kumar, Wolfgang Macherey, Yanping Huang, David A Ross, Irfan Essa, Yonatan Bisk, Ming-Hsuan Yang, et al. Spae: Semantic pyramid autoencoder for multimodal generation with frozen llms. *arXiv preprint arXiv:2306.17842*, 2023.

[17] Hao Liu, Wilson Yan, and Pieter Abbeel. Language quantized autoencoders: Towards unsupervised text-image alignment. *arXiv preprint arXiv:2302.00902*, 2023.

[18] Lei Zhu, Fangyun Wei, and Yanye Lu. Beyond text: Frozen large language models in visual signal comprehension. *arXiv preprint arXiv:2403.07874*, 2024.

[19] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. In *The Eleventh International Conference on Learning Representations*, 2022.

[20] Mengqi Huang, Zhendong Mao, Zhuowei Chen, and Yongdong Zhang. Towards accurate image coding: Improved autoregressive image generation with dynamic vector quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22596–22605, 2023.

[21] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

[22] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532, 2022.

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[24] 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.

[25] Chuanxia Zheng, Tung-Long Vuong, Jianfei Cai, and Dinh Phung. Movq: Modulating quantized vectors for high-fidelity image generation. *Advances in Neural Information Processing Systems*, 35:23412–23425, 2022.

[26] Will Williams, Sam Ringer, Tom Ash, David MacLeod, Jamie Dougherty, and John Hughes. Hierarchical quantized autoencoders. *Advances in Neural Information Processing Systems*, 33:4524–4535, 2020.

[27] Jialun Peng, Dong Liu, Songcen Xu, and Houqiang Li. Generating diverse structure for image inpainting with hierarchical vq-vae. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10775–10784, 2021.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[29] Jiahui Zhang, Fangneng Zhan, Christian Theobalt, and Shijian Lu. Regularized vector quantization for tokenized image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18467–18476, 2023.

[30] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[31] Chuanxia Zheng and Andrea Vedaldi. Online clustered codebook. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22798–22807, 2023.

[32] 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.

[33] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International conference on machine learning*, pages 1747–1756. PMLR, 2016.

[34] Alex Graves and Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012.

[35] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

[36] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[38] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

[39] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022.

[40] Zhicong Tang, Shuyang Gu, Jianmin Bao, Dong Chen, and Fang Wen. Improved vector quantized diffusion models. *arXiv preprint arXiv:2205.16007*, 2022.

[41] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023.

[42] Chen Wei, Chenxi Liu, Siyuan Qiao, Zhishuai Zhang, Alan Yuille, and Jiahui Yu. De-diffusion makes text a strong cross-modal interface. *arXiv preprint arXiv:2311.00618*, 2023.

[43] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv:2111.06377*, 2021.

[44] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[45] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[46] Minheng Ni, Xiaoming Li, and Wangmeng Zuo. Nuwa-lip: language-guided image inpainting with defect-free vqgan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14183–14192, 2023.

[47] Jie Shi, Chenfei Wu, Jian Liang, Xiang Liu, and Nan Duan. Divae: Photorealistic images synthesis with denoising diffusion decoder. *arXiv preprint arXiv:2206.00386*, 2022.

[48] Tung-Long Vuong, Trung Le, He Zhao, Chuanxia Zheng, Mehrtash Harandi, Jianfei Cai, and Dinh Phung. Vector quantized wasserstein auto-encoder. *arXiv preprint arXiv:2302.05917*, 2023.

[49] Mengqi Huang, Zhendong Mao, Quan Wang, and Yongdong Zhang. Not all image regions matter: Masked vector quantization for autoregressive image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2002–2011, June 2023.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Table 10: The impact of maintaining a static codebook and incorporating a projector on ImageNet.

Static	Projector	Utilization (%)	rFID	LPIPS	PSNR	SSIM
✓		3.9	18.6	0.36	19.2	40.4
	✓	99.1	2.65	0.12	23.7	58.0
✓	✓	**99.9**	**2.62**	**0.12**	**23.8**	**58.1**

Table 11: Ablation study on the dimension of the projected codebook on ImageNet.

Dimension	Utilization (%)	rFID	LPIPS	PSNR	SSIM
8	99.8	2.66	0.12	23.9	59.2
16	99.8	2.36	0.12	23.8	58.8
32	99.8	2.75	0.13	23.6	58.7
128	99.8	2.37	0.12	23.6	58.3
256	99.8	2.49	0.12	23.9	59.4
512	99.9	2.76	0.12	23.5	58.1

A More Implementation Details

GPT. We train GPT with a batch size of 1024 across 32 Nvidia V100 GPUs. The Adam optimizer is employed with an initial learning rate of 4.5e^{-4}, and the model is trained for 100 epochs. Moreover, a linear decay schedule is used for adjusting the learning rate. A 5-epoch linear warm-up phase is adopted. Top-k sampling is adopted for auto-regressive generation, where k is set as 10% of the vocabulary size.

LDM. The implementation utilizes four UNet layers with channel dimensions of $\{256, 1024, 1024, 256\}$. Conditions are integrated through a cross-attention mechanism at each UNet layer. The model is trained using the Adam optimizer with an initial learning rate of 4.5e^{-4} and a batch size of 448, distributed across 8 Nvidia V100 GPUs. The training process is conducted over 100 epochs. The classifier-free guidance scale is set as 1.4 for class-conditional generation.

DiT. We employ the 28-layer DiT-XL model with a patch size of 2, consisting of 675 million parameters. The model features 16 attention heads and an embedding dimension of 1152. To handle class conditions, we utilize the AdaLN-Zero block. For optimization, the Adam optimizer is used with an initial learning rate of 4.5e^{-4}, and the model is trained for 400,000 iterations on the ImageNet dataset. The training is performed with a batch size of 256 across 8 Nvidia V100 GPUs. The classifier-free guidance scale is set as 8 for class-conditional generation.

SiT. We utilize SiT-XL as our flow-based generative model, mirroring the architecture of DiT-XL. For optimization, the Adam optimizer is employed with an initial learning rate of 4.5e^{-4}. The model undergoes training for 400,000 iterations on the ImageNet dataset. The training process is conducted with a batch size of 256, distributed across 8 Nvidia V100 GPUs. The classifier-free guidance scale is set to 8 for class-conditional generation.

B More Experiments

Projector and Static Codebook. As described in Section 3.2, the codebook is initialized using a CLIP vision encoder to extract patch-level features on ImageNet. During the training of our VQGAN-LC, we optimize a projector to map the entire codebook to a latent space. Table 10 illustrates the significance of tuning the projector on the static codebook by comparing our default strategy with two alternatives: 1) omitting the projector; and 2) making each entry in the initialized codebook trainable. Our results show that incorporating the projector markedly improves performance, whereas making the codebook entries trainable has minimal impact.

Dimension of the Projected Codebook. In our VQGAN-LC, the projector transforms the codebook $\mathcal{B}$ into a latent codebook $\mathcal{B}'$, with each entry in $\mathcal{B}'$ having a dimension denoted as D'. Table 11 shows the results of varying D' from 8 to 512. We find that our model's performance remains stable regardless of the value of D', consistently achieving a codebook utilization rate of over 99% in all configurations.

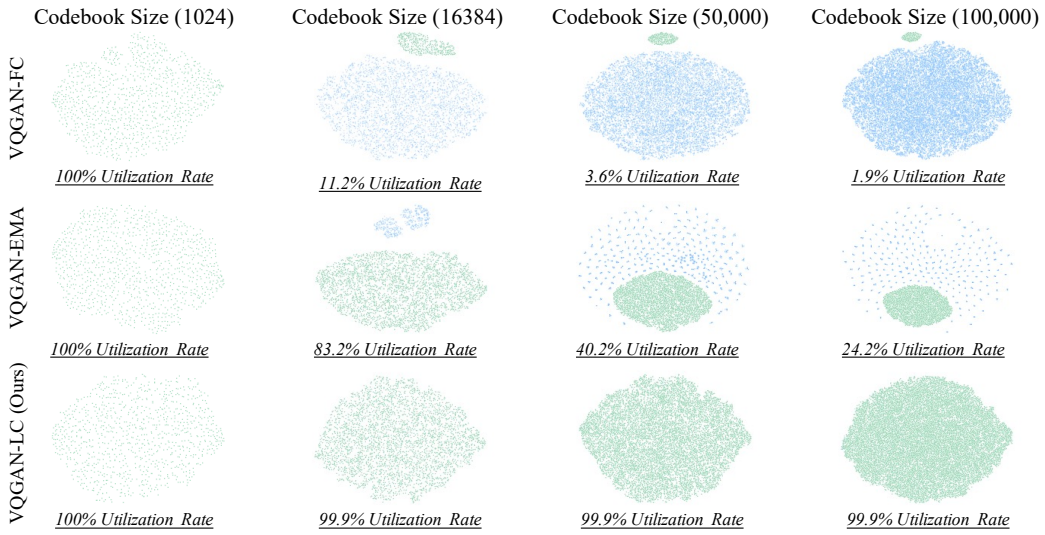


Figure 4: Visualization of the active and inactive codes for three models (VQGAN-FC, VQGAN-EMA, and our VQGAN-LC) using t-SNE.

Visualization of Active and Inactive Codes. Figure 4 shows the distribution of the active and inactive codes for three models: VQGAN-FC, VQGAN-EMA, and our VQGAN-LC. The visualization is created using t-SNE. Active codebook entries are highlighted in green, while inactive ones are shown in blue. As the codebook size increases, more codes tend to be inactive in both VQGAN-FC and VQGAN-EMA models.

Token Replacement. VQGAN, VQGAN-FC, and VQGAN-EMA all utilize a codebook of 16,384 entries to achieve optimal performance. In contrast, our VQGAN-LC can scale the codebook size up to 100,000 while maintaining an exceptionally high utilization rate of over 99%, allowing each token to represent more detailed visual elements. This is verified through an ablation study: for each input image, we use VQGAN-FC (-EMA/-LC)'s encoder to convert the image into a token map. We then replace each token in the map with the M^{th} nearest entry from its codebook. The modified token map is fed into the decoder for reconstruction. Figure 5 shows the PSNR results on a subset of ImageNet, using images from 100 randomly selected categories, and visualizes the results of our VQGAN-LC, VQGAN-FC, and VQGAN-EMA when M is set to 1, 50, 100, and 1000.

C Visualizations

In Figures 6-9, we present the class-conditional generation results at a resolution of 256×256 for our VQGAN-LC with GPT [2], LDM [3], DiT [4], and SiT [5], respectively, using 256 (16×16) tokens on ImageNet. Additionally, Figure 10 illustrates the results of VQGAN-LC with LDM using 1024 (32×32) tokens on ImageNet. Figure 11 shows the unconditional generation results at a resolution of 256×256 for VQGAN-LC with LDM on the FFHQ dataset, utilizing 256 (16×16) tokens. The introduction of a large-scale codebook facilitates the generation of images with diverse poses, intricate textures, and complex backgrounds.

D Limitations and Broad Impacts

We present a new image quantization technique called VQGAN-LC, which expands the codebook size to 100,000 with a utilization rate of 99%. This approach has the potential to enhance any downstream applications that involve image quantization models. However, VQGAN-LC is trained on the ImageNet and FFHQ datasets, which restricts downstream applications, like image generation, to producing images from the limited categories found in these datasets. While training VQGAN-LC on larger datasets like LAION-5B may improve its utility in downstream applications, it would also be more costly and computationally demanding. Furthermore, please refrain from using this model to generate malicious or inappropriate content. It is intended solely for positive and constructive purposes in research and creativity.

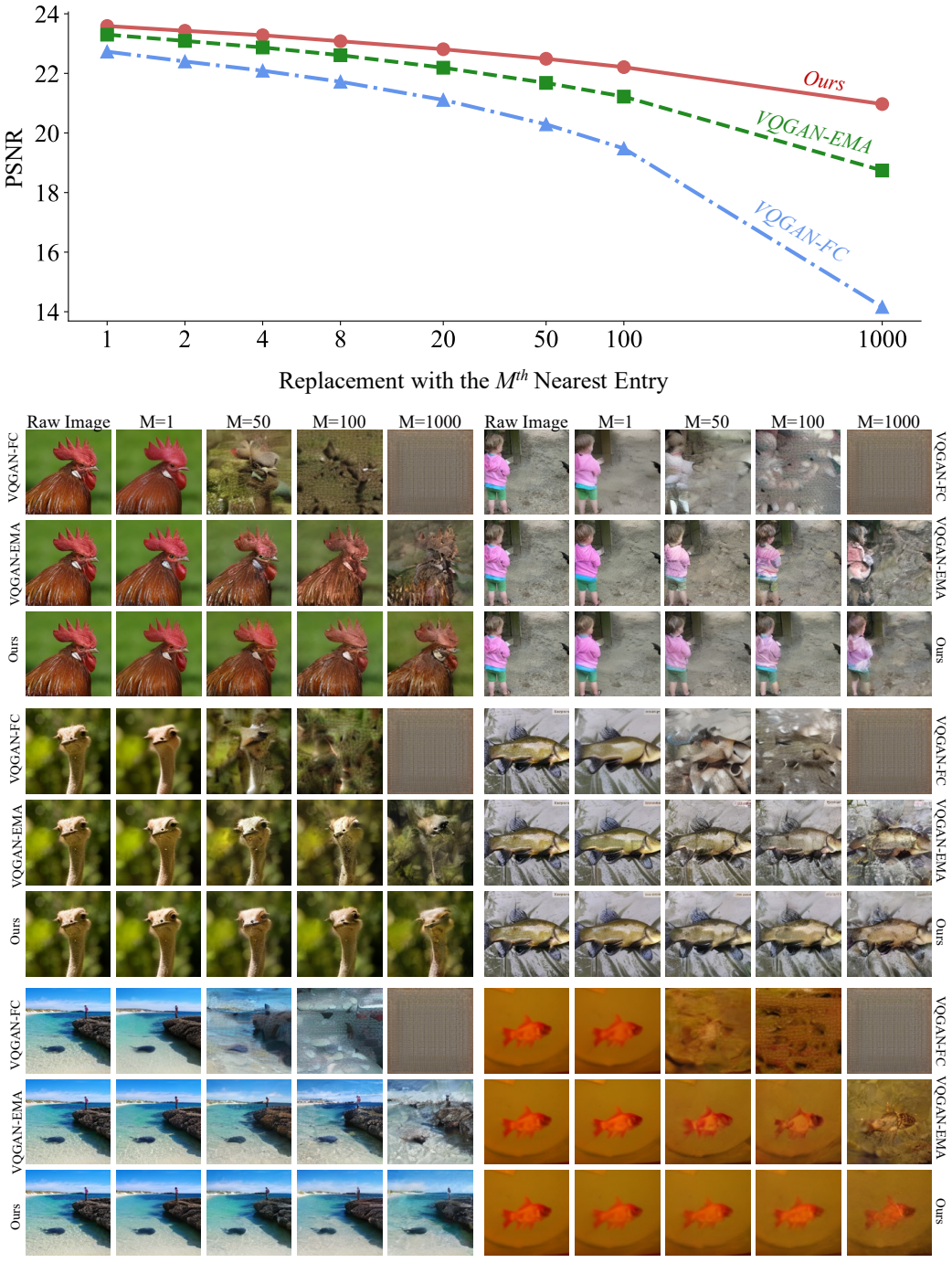


Figure 5: For a given image, we employ an image quantization model (VQGAN-FC, VQGAN-EMA, or our VQGAN-LC) to transform it into a token map. Each token in this map is then substituted with the M^{th} nearest entry from the codebook. This altered token map is subsequently fed into the decoder for reconstruction. (Top) PSNR for each configuration. (Bottom) Reconstruction visualizations for the three models.

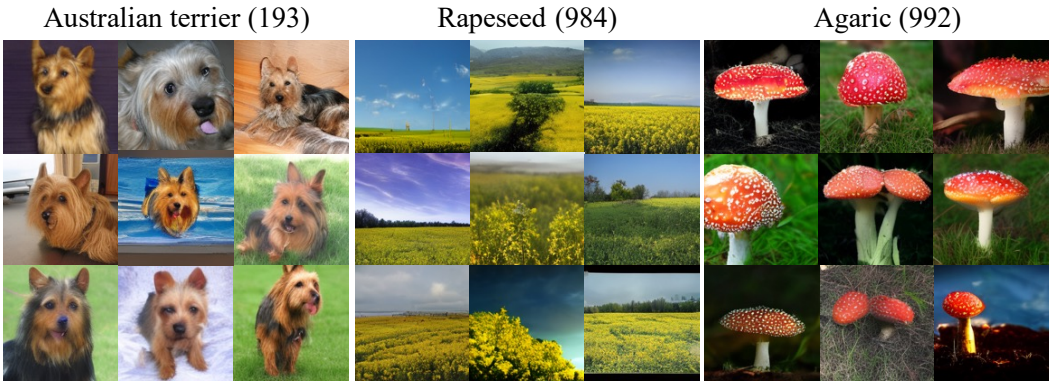


Figure 6: Qualitative results of class-conditional generation using our VQGAN-LC with GPT [2] on ImageNet, utilizing 256 (16×16) tokens. We display the category name and corresponding category ID for each group.

Figure 7: Qualitative results of class-conditional generation using our VQGAN-LC with LDM [3] on ImageNet, utilizing 256 (16×16) tokens and a classifier-free guidance scale of 1.4. We display the category name and corresponding category ID for each group.

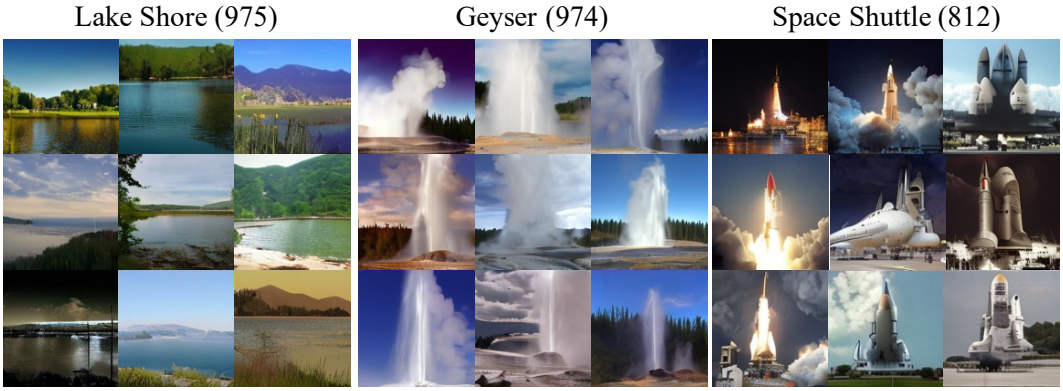


Figure 8: Qualitative results of class-conditional generation using our VQGAN-LC with DiT [4] on ImageNet, utilizing 256 (16×16) tokens and a classifier-free guidance scale of 8.0. We display the category name and corresponding category ID for each group.

Figure 9: Qualitative results of class-conditional generation using our VQGAN-LC with SiT [5] on ImageNet, utilizing 256 (16×16) tokens and a classifier-free guidance scale of 8.0. We display the category name and corresponding category ID for each group.

Figure 10: Qualitative results of class-conditional generation using our VQGAN-LC with LDM [3] on ImageNet, utilizing 1024 (32×32) tokens and a classifier-free guidance scale of 1.4. We display the category name and corresponding category ID for each group.

Figure 11: Qualitative results of unconditional generation using our VQGAN-LC with LDM [3] on FFHQ, utilizing 256 (16×16) tokens.

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: This paper introduces a novel approach named VQGAN-LC, which scales the codebook size of VQGAN to 100,000, with a utilization rate of 99%.

   Guidelines:

   - The answer NA 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 NA 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: Limitations are discussed in the appendix.

   Guidelines:

   - The answer NA 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: [NA]

Justification: No theory assumptions and proofs are introduced in this work.

Guidelines:

- The answer NA 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 all the information including training and inference to reproduce the main experimental results.

   Guidelines:

   - The answer NA 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: [Yes]

Justification: The code will be made publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: we specify the training and test details, hyperparameters, optimizers and other implementation details.

   Guidelines:

   - The answer NA 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: [NA]

   Justification: we report rFID, LPIPS, PSNR and SSIM for image reconsturction, top-1 accuracy for image classification, and FID for image generation.

   Guidelines:

   - The answer NA 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: [Yes]

   Justification: We use Nvidia V100 GPUs across experiments.

   Guidelines:

   - The answer NA 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 strictly comply with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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 the societal impacts in the appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: this is discussed in the appendix.

    Guidelines:

    - The answer NA 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: [Yes]

    Justification: we use public datasets like ImageNet and FFHQ.

    Guidelines:

    - The answer NA 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: [No]

    Justification: no new assets are introduced.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: No crowdsourcing experiments and research with human subjects are included in this work.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: No such potential risks in this work.

    Guidelines:

    - The answer NA 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.