VQ-VAE + DDPM.

→ global dependency + diffusion 속도 해결

* Introduction.

VQVAE는 image compression에 효과적

↳ codebook을 AR과 transformer 등으로 생성.

↳ 생성 단계에 inductive bias 및 model size, sampling 시간 문제.

DDPM이 나타남

↳ 간단한 분포에서 복잡한 이미지 분포.

↳ 하지만 느림.

∴ VQ-DDM 제안.

1) codebook 학습

2) prior 분포 맞추기.

↳ Rebuild and finetune.(ReFiT)

-Contribution.

1) Diffusion으로 global에 효과적.

2) ReFiT로 codebook usage ↑.

3) param 수 감소, 속도↑.

* Preliminaries

* Diffusion model in Continuous space

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \tag{1}$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}). \tag{2}$$

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \tag{3}$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)). \tag{4}$$

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)] \\
&\quad = \mathbb{E}_{q(\mathbf{x}_0)} \log \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] \\
&\quad \geq \mathbb{E}_{q(\mathbf{x}_{0:T})} \left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] =: L_{\text{vlb}}.
\end{aligned} \tag{5}$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \tag{6}$$

$$\begin{aligned}
L_{\text{vlb}} = \mathbb{E}_{q(\mathbf{x}_0)}[&D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)) - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1) \\
&+ \sum_{t=2}^{T} D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))].
\end{aligned} \tag{7}$$

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right), \tag{8}$$

$$\begin{aligned}
\Sigma_\theta(\mathbf{x}_t, t) = \exp(&v_\theta(\mathbf{x}_t, t) \log \beta_t \\
&+ (1 - v_\theta(\mathbf{x}_t, t)) \log \tilde{\beta}_t).
\end{aligned} \tag{9}$$

$$L_{\text{simple}} = \mathbb{E}_{t, \mathbf{x}_0, \epsilon} \left[||\epsilon - \epsilon_\theta(\mathbf{x}_t, t)||^2 \right],$$

$$L_{\text{hybrid}} = L_{\text{simple}} + \lambda L_{\text{vlb}}. \tag{11}$$

* Discrete representation of Images

VQVAE. - dictionary learning process.

$$\mathbf{z} = \text{Quantize}(\mathbf{h}) := \arg\min_k ||h_{i,j} - z_k||, \tag{12}$$

$$\hat{\mathbf{x}} = D(\mathbf{z}) = D(\text{Quantize}(E(\mathbf{x}))). \tag{13}$$

$$L = ||\mathbf{x} - \hat{\mathbf{x}}||^2 + ||sg[E(\mathbf{x})] - \mathbf{z}|| + \beta||sg[\mathbf{z}] - E(\mathbf{x})||, \tag{14}$$

느리지만 codebook usage를 높여야함.

* Method.

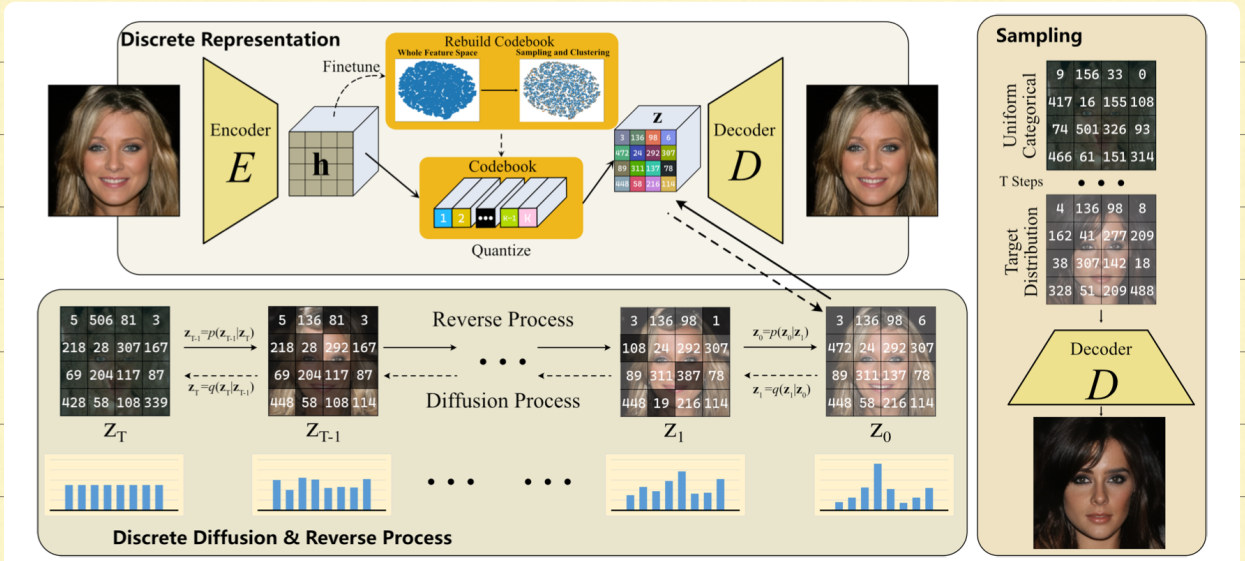

Figure 2. The proposed VQ-DDM pipeline contains 2 stages: (1) Compress the image into discrete variables via discrete VAE. (2) Fit a prior distribution over discrete coding by a diffusion model. Black squares in the diffusion diagram illustrate states when the underlying distributions are uninformative, but which become progressively more specific during the reverse process. The bar chart at the bottom of the image represents the probability of a particular discrete variable being sampled.

* Discrete Diffusion Model

z_t는 discrete에 대한 one-hot encoding. z_t^{logits}는 그에 해당하는 distribution.

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathrm{Cat}(\mathbf{z}_t; \mathbf{z}_{t-1}^{\mathrm{logits}}\mathbf{Q}_t), \qquad (15)$$

$\mathbf{Q}_t$ → transition Matrix.

$Q_t = (1-\beta_t)I + \beta_t/K$

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathrm{Cat}(\mathbf{z}_t; (1-\beta_t)\mathbf{z}_{t-1}^{\mathrm{logits}} + \beta_t/K). \qquad (16)$$

∴ DDPM과 유사하게 $\alpha_t = 1-\beta_t$, $\bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s$

$$q(\mathbf{z}_t|\mathbf{z}_0) = \mathrm{Cat}(\mathbf{z}_t; \bar{\alpha}_t\mathbf{z}_0 + (1-\bar{\alpha}_t)/K) \qquad (17)$$

$$or \quad q(\mathbf{z}_t|\mathbf{z}_0) = \mathrm{Cat}(\mathbf{z}_t; \mathbf{z}_0\bar{\mathbf{Q}}_t);\ \bar{\mathbf{Q}}_t = \prod_{s=0}^{t}\mathbf{Q}_s. \qquad (18)$$

$$\bar{\alpha} = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T+s}{1+s}\times\frac{\pi}{2}\right)^2. \qquad (19)$$

Bayes의 따라, posterior $q(z_{t-1}|z_t, z_0)$

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{z}_0) = \text{Cat}\left(\mathbf{z}_t; \frac{\mathbf{z}_t^{\text{logits}}\mathbf{Q}_t^\top \odot \mathbf{z}_0\bar{\mathbf{Q}}_{t-1}}{\mathbf{z}_0\bar{\mathbf{Q}}_t{\mathbf{z}_t^{\text{logits}}}^\top}\right) \tag{20}$$

$$= \text{Cat}(\mathbf{z}_t;\ \boldsymbol{\theta}(\mathbf{z}_t, \mathbf{z}_0)/\sum_{k=1}^{K}\theta_k(z_{t,k}, z_{0,k})),$$

→ $\theta(z_t, z_0)$의 normalize

$= N[\theta(z_t, z_0)]$

$$\boldsymbol{\theta}(\mathbf{z}_t, \mathbf{z}_0) = [\alpha_t \mathbf{z}_t^{\text{logits}} + (1-\alpha_t)/K] \odot [\bar{\alpha}_{t-1}\mathbf{z}_0 + (1-\bar{\alpha}_{t-1})/K]. \tag{21}$$

$$p_\theta(\mathbf{z}_0|\mathbf{z}_1) = \text{Cat}(\mathbf{z}_0|\hat{\mathbf{z}}_0),$$
$$p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t) = \text{Cat}(\mathbf{z}_t|\ \text{N}[\boldsymbol{\theta}(\mathbf{z}_t, \hat{\mathbf{z}}_0)]). \tag{22}$$

$$\hat{\mathbf{z}}_0 = \mu(\mathbf{Z}_t, t) + \mathbf{Z}_t. \tag{23}$$

μ = $N^{h \times w}$

$$\text{KL}(q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{z}_0)||p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)) = \sum_k \text{N}[\boldsymbol{\theta}(\mathbf{z}_t, \mathbf{z}_0)] \times \log \frac{\text{N}[\boldsymbol{\theta}(\mathbf{z}_t, \mathbf{z}_0)]}{\text{N}[\boldsymbol{\theta}(\mathbf{z}_t, \hat{\mathbf{z}}_0)]}. \tag{24}$$

* Re-build and fine-tune strategy

codebook의 개수는 k인데, 수를 풍부하게 하기 위해서, k는 커진다.

하지만 큰 k는 diffusion의 복잡도를 증가시킨다. $O(k^2T)$

∴ ReFit → k는 높이고 recon↑.

eq 내에서보면, 고려해야 term이 codebook 관련.

하지만 enc에서 선택된 codebook만 학습이 된다.

└ usage가 낮음.

↓

Training image 전체에 대해 code를 뽑고.

AFk-MC^{2}를 이용해서 k clustering, 그리고 fine-tune.

* Experiments and Analysis.

* Codebook Quality

Model	Latent Size	Capacity	Usage of $\mathbb{Z}$		FID ↓	
			CelebA	ImageNet	CelebA	ImageNet
VQ-VAE-2	Cascade	512	∼65%	-	-	∼10
DALL-E	32x32	8192	-	-	-	32.01
VQ-GAN	16x16	16384	-	5.96%	-	4.98
VQ-GAN	16x16	1024	31.85%	33.67%	10.18	7.94
ours ($P = 100k$)	16x16	1024	-	100%	-	4.98
ours ($P = 20k$)	16x16	1024	97.07%	100%	5.59	5.99
ours ($P = 20k$)	16x16	512	93.06%	100%	5.64	6.95

[1] All methods are trained straight-through, except DALL-E with Gumbel-Softmax [25].
[2] CelebA-HQ at 256×256. Reported FID is between $30k$ reconstructed data vs training data.
[3] Reported FID is between $50k$ reconstructed data vs validation data

Table 1. FID between reconstructed images and original images on CelebA-HQ and ImageNet

Figure 3. Reconstruction images 384 × 384 from ImageNet based VQ-GAN and ReFiT

(a) Source (b) VQ-GAN (c) ReFiT K=1024 (d) ReFiT K=512

Figure 4. Reconstruction images of CelebA HQ 256 × 256 from VQ-GAN and ReFiT.

→ Generation Quality

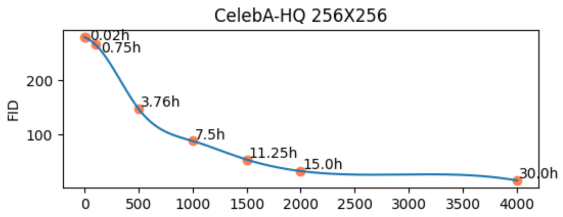

Figure 5. Steps and corresponding FID during the sampling. The text annotations are hours to sample 50k latent feature maps on 1 NVIDIA 2080Ti GPU

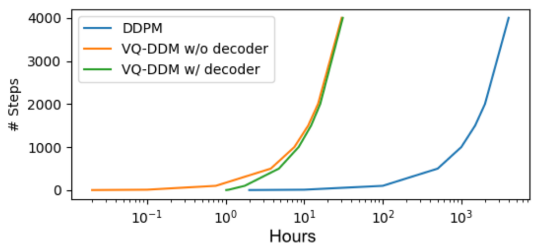

Figure 6. Hours to sampling 50k latent codes by VQ-DDM and generating 50k images with VQ-DDM and DDPM

Method	FID ↓	Params	FLOPs
Likelihood-based			
GLOW [17]	60.9	220 M	540 G
NVAE [33]	40.3	1.26 G	185 G
ours ($K = 1024$ w/o ReFiT)	22.6	117 M	1.06 G
VAEBM [39]	20.4	127 M	8.22 G
ours ($K = 512$ w/ ReFiT)	18.8	117 M	**1.04 G**
DC-VAE [24]	15.8	-	-
ours ($K = 1024$ w/ ReFiT)	13.2	117 M	1.06 G
DDIM(T=100) [31]	10.9	114 M	124 G
VQ-GAN + Transformer [8]	10.2	802 M	102 G[a]
Likelihood-free			
PG-GAN [14]	8.0	46.1 M	14.1 G

[a] VQ-GAN is an autoregressive model, and the number in the table is the computation needed to generate the full size latent feature map. The FLOPs needed to generate one discrete index out of 256 is 0.399 G.

Table 2. FID on CelebA HQ 256×256 dataset. All the FLOPs in the table only consider the generation stage or inference phase for one 256×256 images.

* Image Inpainting.

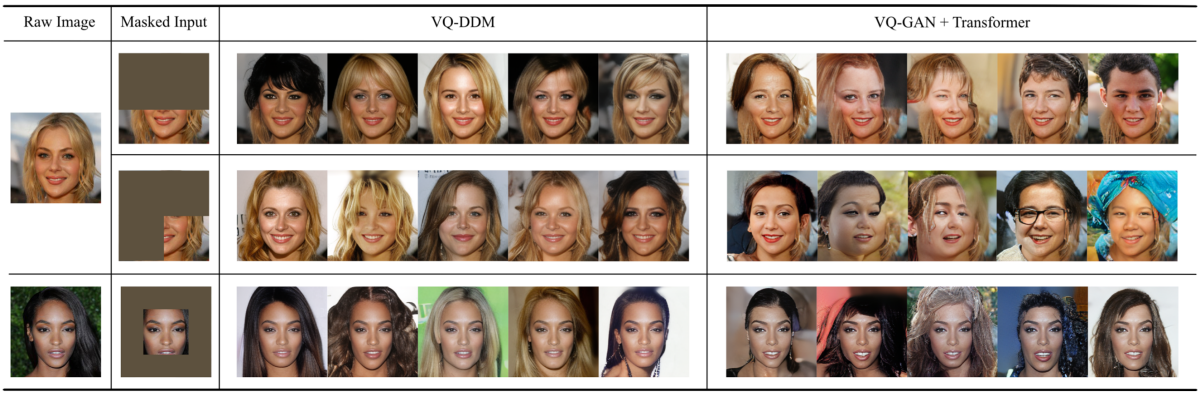

Figure 8. Completions with the arbitrary masks.

* Appendix

Encoder	Decoder
Conv2D	Conv2D
4×{ResDown}	Middle Block
Middle Block	4×{ResDown}
GN, Swish, Conv2D	GN, Swish, Conv2D

[1] ResDown is the combination of a Residual Block and Downsample Block, if the feature map size matches the preset value, there will be an addition non-local self-attention block.

[2] Middle Block is the cascade of one Residual Block, one Self-attention Block and one more Residual Block.

[3] GN means the group normalization [38]

Table 3. Brief Architecture of the VQ-GAN encoder and decoder

Diffusion model의 경우 PixelCNN++ backbone.