긴 AR은 long-dependency에 약하고, 시간도 오래 걸림.

∴ RQ-VAE는 VQ stack

RQ-Transformer는 다음 step을 예측하는 방식.

## * Introduction.

VQ로 양자화 후, AR로 sequence 생성.

AR 길이를 줄이는 것이 시간 제약에 중요하지만, 성능과 trade-off

그리고 너무 큰 codebook은 collapse 유발.

RQ로 feature map을 더 잘 근사.

↳ coarse 부터 fine 까지 재귀적으로.

RQ-Transformer에서 양자화된 feature map이 feature vector로 바뀌어 input.

↳ D개의 코드를 한번에 예측.

↳ soft labeling, stochastic sampling 추가.

Contribution.

1) stack VQ가 가능한 RQ-VAE 제안.

2) RQ-Transformer로, exposure bias 해결, codebook 크기↓

## * Method.

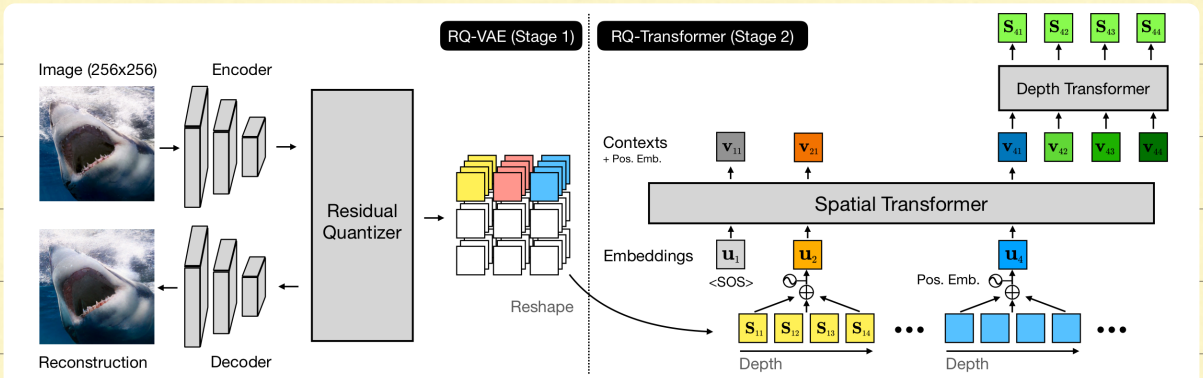



Figure 2. An overview of our two-stage image generation framework composed of RQ-VAE and RQ-Transformer. In stage 1, RQ-VAE uses the residual quantizer to represent an image as a stack of $D = 4$ codes. After the stacked map of codes is reshaped, RQ-Transformer predicts the $D$ codes at the next position. More details are available in Section 3.

## * Stage 1

VQVAE 공식 소개 후, codebook의 크기를 늘리지 않고도 feature map을 더 정확히 근사하는 RQ-VAE 소개.

### - Formulation.

Codebook $C$의 정의 $\{k, e(k)\}_{k\in[K]}$ , $e(k)\in R^{n_z}$, $n_z$는 codebook dim., $K$는 개수. (index, embedding)

$Q(z;C)$는 $z$가 Quantize 된 것을 의미.

$$\mathcal{Q}(\mathbf{z};\mathcal{C}) = \underset{k\in[K]}{\arg\min} \|\mathbf{z} - \mathbf{e}(k)\|_2^2. \quad (1)$$

각 feature vector에 Q 적용. → code map $M \in K^{H\times W}$, quantized feature map $\hat{Z}\in R^{H\times W\times n_z}$

$$\mathbf{M}_{hw} = \mathcal{Q}(\mathbf{Z}_{hw};\mathcal{C}), \quad \hat{\mathbf{Z}}_{hw} = \mathbf{e}(\mathbf{M}_{hw}), \quad (2)$$

recon : $\hat{X} = G(\hat{Z})$

M의 H, W가 ARel cost와 time에 중요.

그리고 VQVAE는 lossy compression이 가능하고. bit = $HW \log_2 K$.

∴ H/2, W/2가 되면서 recon loss를 유지하려면, $K^4$가 됨.

↳ 하지만 K를 키우면 codebook collapse가 일어남.

- Residual Quantization.

코드북 크기를 늘리는 대신 RQ. depth : D.

$$\mathcal{RQ}(\mathbf{z}; \mathcal{C}, D) = (k_1, \cdots, k_D) \in [K]^D, \quad (3)$$

$r_0$부터 시작해서, $k_d$만큼 recursively

$$k_d = \mathcal{Q}(\mathbf{r}_{d-1}; \mathcal{C}), \quad \rightarrow \text{index}$$
$$\mathbf{r}_d = \mathbf{r}_{d-1} - \mathbf{e}(k_d), \quad \rightarrow \text{차이가 다음 } r_d. \quad (4)$$

Vector를 coarse부터 fine까지.

$\hat{z}^{(1)}$이 $e(k_1)$과 가장 가깝고, feature level도 가까움

나머지는 error를 줄이는 것에 사용 → 본질적인 내용이 더 나은 근사값.

하지만 codebook은 다 공유함. → 그게 이득.

1. 코드북 크기가 작아짐.

2. utility ↑.

즉, 더 정확한 근사

VQVAE의 경우 $R^{n_z}$ dim으로 k-clustering이지만 RQ는 $k^D$ clustering

↳ D를 늘리면 exponentially increase.

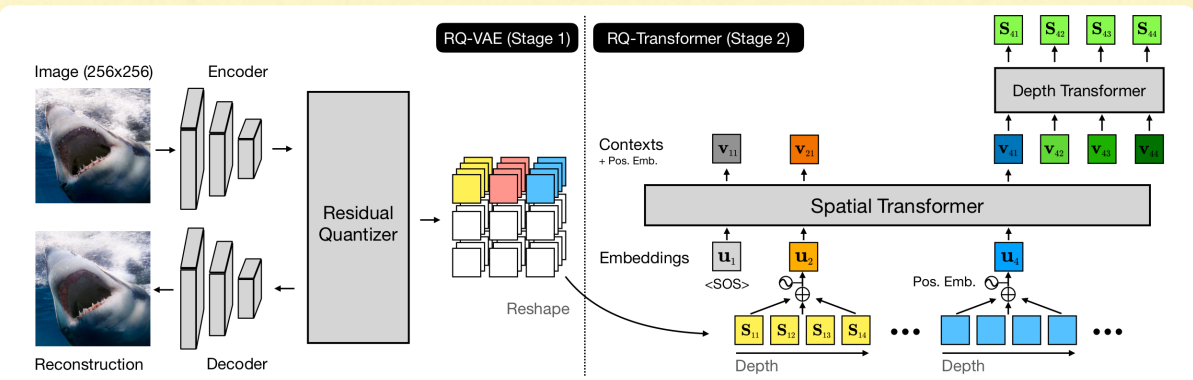



Figure 2. An overview of our two-stage image generation framework composed of RQ-VAE and RQ-Transformer. In stage 1, RQ-VAE uses the residual quantizer to represent an image as a stack of $D = 4$ codes. After the stacked map of codes is reshaped, RQ-Transformer predicts the $D$ codes at the next position. More details are available in Section 3.

- RQ-VAE

VQVAE → RQ

code $M: [K]^{H \times W \times D}$, $\hat{z}^{(d)} \in R^{H \times W \times n_z}$

$$\mathbf{M}_{hw} = \mathcal{RQ}(E(\mathbf{X})_{hw}; \mathcal{C}, D),$$
$$\hat{\mathbf{Z}}_{hw}^{(d)} = \sum_{d'=1}^{d} \mathbf{e}(\mathbf{M}_{hwd'}). \quad (5)$$

down sample factor f는 높고, (H,W)는 높아짐.

AR의 계산은 높일 수 있다.

- Training RQ-VAE.

$$\mathcal{L}_{\text{recon}} = \|\mathbf{X} - \hat{\mathbf{X}}\|_2^2, \quad (6)$$
$$\mathcal{L}_{\text{commit}} = \sum_{d=1}^{D} \left\| \mathbf{Z} - \text{sg}\left[\hat{\mathbf{Z}}^{(d)}\right] \right\|_2^2, \quad (7)$$

$\mathcal{L} = \mathcal{L}_{recon} + \beta \mathcal{L}_{commit}$.

Codebook은 coarse-to-fine으로 feature map 근사.

* Stage 2. RQ-Transformer.

RQ codebook을 학습하는 과정

- AR modeling for Coders for depth D.

AR의 순서를 raster로 볼때 Sequence의 길이는 $S \in [K]^{T \times D}$, $T = HW$

$$\mathbf{S}_t = (\mathbf{S}_{t1}, \cdots, \mathbf{S}_{tD}) \in [K]^D \quad \text{for } t \in [T]. \tag{8}$$

AR model은 $p(S)$를 학습

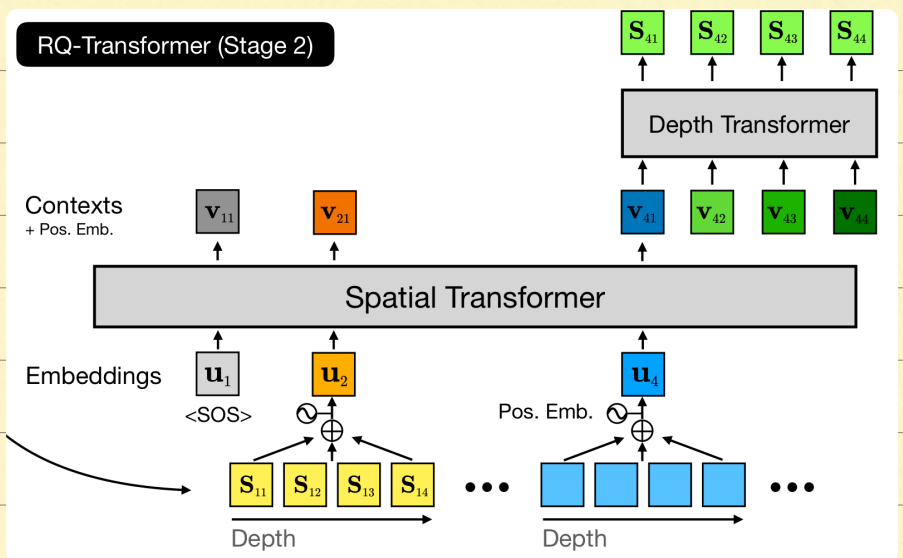
$$p(\mathbf{S}) = \prod_{t=1}^{T} \prod_{d=1}^{D} p(\mathbf{S}_{td} | \mathbf{S}_{<t,d}, \mathbf{S}_{t,<d}). \tag{9}$$

- RQ-Transformer Architecture.

TD개를 그냥 생성하는 것은 length도 길고 compute도 기하급수 늘어남.

∴ D를 한번에 생성

↳ depth transformer, spatial transformer.



- Spatial transformer.

input의 경우 이전 embedding과 raster-scan number

$$\mathbf{u}_t = \mathrm{PE}_T(t) + \sum_{d=1}^{D} \mathbf{e}(\mathbf{S}_{t-1,d}) \quad \text{for } t > 1, \tag{10}$$

$$\mathbf{h}_t = \mathrm{SpatialTransformer}(\mathbf{u}_1, \cdots, \mathbf{u}_t). \tag{11}$$

- Depth Transformer.

context vector h가 주어지면 (t의 대한 $h_t$)

그 t의 대한 depth를 계산

$$\mathbf{v}_{td} = \mathrm{PE}_D(d) + \sum_{d'=1}^{d-1} \mathbf{e}(\mathbf{S}_{td'}) \quad \text{for } d > 1, \tag{12}$$

t는 $u_t$의 이미 embedding 되어 있다고 가정.

∴ $P_{td}(k) = p(S_{td=k} | S_{<t,d}, S_{t,<d})$ 를 구함.

$$\mathbf{p}_{td} = \mathrm{DepthTransformer}(\mathbf{v}_{t1}, \cdots, \mathbf{v}_{td}). \tag{13}$$

최종 AR NLL loss는.

$$\mathcal{L}_{AR} = \mathbb{E}_{\mathbf{S}} \mathbb{E}_{t,d} \left[ -\log p(\mathbf{S}_{td} | \mathbf{S}_{<t,d}, \mathbf{S}_{t,<d}) \right]. \tag{14}$$

- Computational Complexity

TXD크 sequence를 만드는 것보다 적은 complexity

$O(NT^2D^2) \rightarrow O(N_s T^2 + N_s T D^2)$

- Soft Labeling and Stochastic Labeling

RQ는 오류가 누적되기 때문에 exposure bias ↑

soft labeling 과 stochastic sampling은 이를 해결하기 위함.

$$Q_\tau(k|\mathbf{z}) \propto e^{-\|\mathbf{z}-\mathbf{e}(k)\|_2^2/\tau} \quad \text{for } k \in [K]. \qquad (15)$$

- Soft labeling of target code.

코드는 embedding 거리 기반

- Stochastic sampling for codes of RQ-VAE

* Experiments.

Table 1. Comparison of FIDs for unconditional image generation on LSUN-{Cat, Bedroom, Church} [48] and FFHQ [25].

| | Cat | Bedroom | Church | FFHQ |
|---|---|---|---|---|
| VDVAE [7] | - | - | - | 28.5 |
| DDPM [21] | 19.75 | 4.90 | 7.89 | - |
| ImageBART [13] | 15.09 | 5.51 | 7.32 | 9.57 |
| StyleGAN2 [26] | 7.25 | 2.35 | 3.86 | 3.8 |
| BigGAN [3] | - | - | - | 12.4 |
| DCT [33] | - | 6.40 | 7.56 | 13.06 |
| VQ-GAN [14] | 17.31 | 6.35 | 7.81 | 11.4 |
| **RQ-Transformer** | 8.64 | 3.04 | 7.45 | 10.38 |

Table 2. Comparison of FIDs and ISs for class-conditioned image generation on ImageNet [9] 256×256. † denotes a model without our stochastic sampling and soft labeling. ‡ denotes the use of rejection sampling or gradient guidance by pretrained classifier. ∗ denotes the use of RQ-VAE trained for 50 epochs.

| | Params | FID | IS |
|---|---|---|---|
| ***without rejection sampling or gradient guidance*** | | | |
| ADM [11] | 554M | 10.94 | 101.0 |
| ImageBART [13] | 3.5B | 21.19 | 61.6 |
| BigGAN [3] | 164M | 7.53 | 168.6 |
| BigGAN-deep [3] | 112M | 6.84 | 203.6 |
| VQ-VAE2 [39] | 13.5B | ∼31 | ∼45 |
| DCT [33] | 738M | 36.5 | n/a |
| VQ-GAN [14] | 1.4B | 15.78 | 74.3 |
| **RQ-Transformer** | 480M | 15.72 | 86.8±1.4 |
| **RQ-Transformer**$^\dagger$ | 821M | 14.06 | 95.8±2.1 |
| **RQ-Transformer** | 821M | 13.11 | 104.3±1.5 |
| **RQ-Transformer** | 1.4B | 11.56 | 112.4±1.1 |
| **RQ-Transformer**$^*$ | 1.4B | 8.71 | 119.0±2.5 |
| **RQ-Transformer**$^*$ | 3.8B | 7.55 | 134.0±3.0 |
| ***with rejection sampling or gradient guidance*** | | | |
| ADM$^\ddagger$ [11] | 554M | 4.59 | 186.7 |
| ImageBART$^\ddagger$ [13] | 3.5B | 7.44 | 273.5±4.1 |
| VQ-VAE2$^\ddagger$ [39] | 13.5B | ∼10 | ∼330 |
| VQ-GAN$^\ddagger$ [14] | 1.4B | 5.20 | 280.3±5.5 |
| **RQ-Transformer**$^\ddagger$ | 1.4B | 4.45 | 326.0±3.5 |
| **RQ-Transformer**$^{*\ddagger}$ | 1.4B | 3.89 | 337.5±4.6 |
| **RQ-Transformer**$^{*\ddagger}$ | 3.8B | 3.80 | 323.7±2.8 |
| Validation Data | - | 1.62 | 234.0 |

Table 3. Comparison of FID and CLIP score [36] on the validation data of CC-3M [43] for text-conditioned image generation.

| | Params | FID | CLIP-s |
|---|---|---|---|
| VQ-GAN [14] | 600M | 28.86 | 0.20 |
| ImageBART [13] | 2.8B | 22.61 | 0.23 |
| **RQ-Transformer** | 654M | 12.33 | 0.26 |

Table 4. Comparison of FIDs between ImageNet validation images and their reconstructed images according to codebook size ($K$) and the shape of code map $H \times W \times D$. † denotes the reproduced performance, and $*$ denotes 50 epochs of training.

| | $H \times W \times D$ | $K$ | rFID |
|---|---|---|---|
| VQ-GAN [14] | 16×16×1 | 16,384 | 4.90 |
| VQ-GAN† | 16×16×1 | 16,384 | 4.32 |
| VQ-GAN | 8×8×1 | 16,384 | 17.95 |
| VQ-GAN | 8×8×1 | 65,536 | 17.66 |
| VQ-GAN | 8×8×1 | 131,072 | 17.09 |
| RQ-VAE | 8×8×2 | 16,384 | 10.77 |
| RQ-VAE | 8×8×4 | 16,384 | 4.73 |
| RQ-VAE* | 8×8×4 | 16,384 | 3.20 |
| RQ-VAE | 8×8×8 | 16,384 | 2.69 |
| RQ-VAE | 8×8×16 | 16,384 | 1.83 |

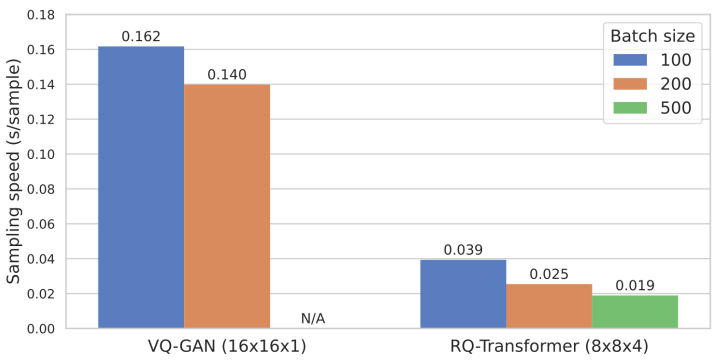



Figure 4. The sampling speed of RQ-Transformer with 1.4B parameters according to batch size and code map shape.

- Ablation

Figure 5. The examples of coarse-to-fine approximation by RQ-VAE. The first example is the original image, and the others are reconstructed from $\hat{\mathbf{Z}}^{(d)}$. As $d$ increases, the reconstructed images become clear and include fine-grained details of the original image.

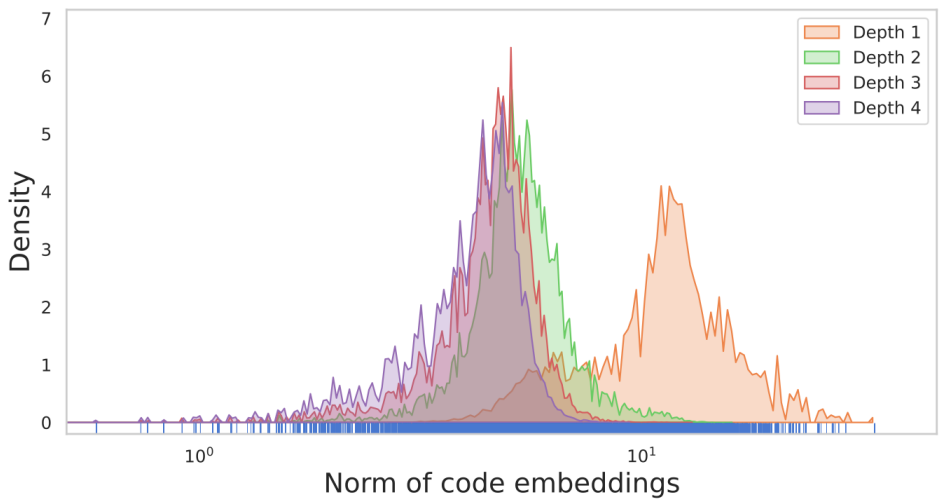



Figure 6. The distribution of used codes at each quantization depth. The blue bar plot represents the code distribution according to the norm of embeddings. ImageNet validation data is used.

* Appendix