Taming Transformers for High-Resolution Image Synthesis (VQ-GAN)

Members: Momal, Dane, Sanjay, Kasun, Mehdi
Group - I
Introduction
Two-Stage Approaches

- First learn an encoding of data with VAE (encoder/decoder)
- Then learn a probabilistic distribution of the encoding

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
VQ-VAE

- VQ-GAN built on top of VQ-VAE (Vector Quantized Variational Auto-Encoder)
- Learns discrete representations of images
- Only works on low resolution
- Global interactions aren’t captured well by convolution density estimation
Hypothesis

- Low-level image structure is well described by a local connectivity
- Higher semantic levels are not locally connected
Idea

- Utilize transformers
- Take convolutional and transformer architecture together
- Use a convolutional approach to learn a codebook
- Learn global compositions
- Use an adversarial approach to ensure codebook learns important parts
VQ-GAN Architecture
Preliminaries

- Adversarial Training Procedure
- Autoregressive Models
- Quantization (Codebook)
Adversarial Training Procedure

Shamane Siriwardhana
The notation AR(p) indicates an autoregressive model of order p. The AR(p) model is defined as

\[ X_t = c + \sum_{i=1}^{p} \phi_i X_{t-i} + \epsilon_t \]
Codebook explanation

- Bag of visual words

https://towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0fb
Quantization
Encoder Part

Encoder

\[ x \in \mathbb{R}^{H \times W \times C} \]

Conv2D \rightarrow \mathbb{R}^{H \times W \times C'}

\[ m \times \{ \text{Residual Block, Downsampling Block} \} \rightarrow \mathbb{R}^{h \times w \times C''} \]

Residual Block \rightarrow \mathbb{R}^{h \times w \times C''}

Non-Local Block \rightarrow \mathbb{R}^{h \times w \times C''}

Residual Block \rightarrow \mathbb{R}^{h \times w \times C''}

GroupNorm, Swish, Conv2D \rightarrow \mathbb{R}^{h \times w \times n_z}
\[ 1 - z_q = q(\hat{z}) := \left( \text{argmin}_{z_k \in Z} \| \hat{z}_{ij} - z_k \|_2 \right) \in \mathbb{R}^{h \times w \times n_z} \]
(Nearest Neighbor!)
Decoder Part

Decoder

\[ z_q \in \mathbb{R}^{h \times w \times n_z} \]

Conv2D \rightarrow \mathbb{R}^{h \times w \times C''} \\
Residual Block \rightarrow \mathbb{R}^{h \times w \times C''} \\
Non-Local Block \rightarrow \mathbb{R}^{h \times w \times C''} \\
Residual Block \rightarrow \mathbb{R}^{h \times w \times C''} \\
\text{m} \times \{ \text{Residual Block, Upsample Block} \} \rightarrow \mathbb{R}^{H \times W \times C'} \\
\text{GroupNorm, Swish, Conv2D} \rightarrow \mathbb{R}^{H \times W \times C}
Training The Codebook, Encoder and Decoder

- $\hat{x} = G(z_q) = G(q(E(x)))$

- non-differentiable because of quantization

- gradient estimator, which simply copies the gradients from the decoder to the encoder

$E(x)$ is the encoder, $G(x)$ is the decoder, and $z_q$ is the codebook.
Loss Function

\[ L_{VQ}(E, G, Z) = \|x - \hat{x}\| + \|sg[E(x)] - z_q\|^2_2 + \|sg[z_q] - E(x)\|^2_2 \]
VQ-VAE
Adversarial Training Part for Richer Codebook

$$\hat{x} = \arg\min_{z_i \in Z} \| \hat{z} - z_i \|$$

CNN Encoder $E(x)$

Code Book $(Z)$

CNN Decoder

Real/Fake Discriminator

$z_q$
Learning the Composition of Images with Transformers (Autoregressive)

- Latent Transformers
  - The quantized encoding of an image $x$ is given by
  - Which is equivalent to sequence of indices from codebook
  - Transformer basically generates the sequence when some indices of the sequence are given
  - This can be formulated as an autoregressive next index prediction

$$z_q = q(E(x)) \in \mathbb{R}^{h \times w \times n_z}$$
$$s \in \{0, \ldots, |\mathcal{Z}| - 1\}^{h \times w}$$
Learning the Composition of Images with Transformers

• Given indices $s < i$, the transformer learns to predict the distribution of possible next indices to compute the likelihood of the full representation as:

$$p(s) = \prod_{i} p(s_i | s_{<i})$$

• The transformer is trained directly by maximizing the log likelihood:

$$\mathcal{L}_{\text{transformer}} = \mathbb{E}_{x \sim p(x)} [-\log p(s)]$$
Transformer Training

\[ p(s) = \prod_i p(s_i|s_{<i}) \]

Transformer

\( s_{<i} \) \( s_i \)

\( z_q \)
Sliding Window Transformer
Transformer architecture of GPT2
Unsupervised Language model (NLP) developed by OpenAI
CNN Encoder

Code Book (Z)

Initialized Randomly

Transformer

\[ p(s) = \prod_i p(s_i | s_{<i}) \]

CNN Discriminator

Real/Fake

\[ G(x) \]

\[ E(x) \]

\[ \hat{z} \]

\[ \text{argmin}_{z_i \in \mathcal{Z}} \| \hat{z} - z_i \| \]

\[ \hat{x} \]
Data Synthesis (unconditional)

Trained Code Book (Z)

Trained Transformer

Transformer

\[ p(s) = \prod_i p(s_i | s_{<i}) \]

Sample

Trained Decoder

CNN Decoder

\[ Z_q \]
Data Synthesis (conditional)

Trained Code Book
(Z)

0
1
2
i
N-2
N-1

Transformer

\[ p(s) = \prod_i p(s_i | s_{<i}) \]

Sample

Trained Decoder

CNN Encoder

CNN Decoder
VQ-GAN Loss functions
Loss Function

- VQ-GAN losses
  - \textbf{Lvq} - VAE loss
  - \textbf{Lgan} - GAN loss
- Transformer Loss

\[
\mathcal{L}_{\text{Transformer}} = \mathbb{E}_{x \sim p(x)} \left[ -\log p(s) \right]
\]

\[
\mathcal{L}_{\text{VQ-GAN}}(E, G, Z)
\]
VAE Loss - Lvq

\[ \mathcal{L}_{\text{VQ}}(E, G, Z) \]

\[
L = \log p(x | z_q(x)) + \|sg[z_e(x)] - e\|^2_2 + \beta \|z_e(x) - sg[e]\|^2_2,
\]

\[
\mathcal{L}_{\text{VQ}}(E, G, Z) = \|x - \hat{x}\|^2 + \|sg[E(x)] - z_q\|^2_2 + \beta \|sg[z_q] - E(x)\|^2_2.
\]
Replace with Perceptual loss

\[ \mathcal{L}_{VQ}(E, G, Z) = \| x - \hat{x} \|^2 + \| \text{sg}[E(x)] - z_q \|^2 + \beta \| \text{sg}[z_q] - E(x) \|^2. \]
Perceptual loss - Reference from original paper
GAN Loss - $L_{GAN}$

$$L_{GAN}(\{E, G, Z\}, D) = \left[ \log D(x) + \log(1 - D(\hat{x})) \right]$$
Final loss function of VQ-GAN

\[
Q^* = \arg \min_{E,G,Z} \max_D \mathbb{E}_{x \sim p(x)} \left[ \mathcal{L}_{\text{VQ}}(E, G, Z) + \lambda \mathcal{L}_{\text{GAN}}(\{E, G, Z\}, D) \right]
\]

Where

\[
\lambda = \frac{\nabla_{GL} [\mathcal{L}_{\text{rec}}]}{\nabla_{GL} [\mathcal{L}_{\text{GAN}}] + \delta}
\]
Autoregressive Transformer

\[ p(s) = \prod_i p(s_i | s_{<i}) \]
Transformer Loss

\[ p(s) = \prod_i p(s_i | s < i) \]

- This sequence can be represented as a multiple of individual probabilities.

\[ P(s) = p(s_1 | s < 1) \cdot p(s_2 | s < 2) \cdot p(s_3 | s < 3) \cdot \ldots \]

\[ p(s) = \prod_i p(s_i | s < i) \]

\[ \mathcal{L}_{\text{Transformer}} = \mathbb{E}_{x \sim p(x)} [ - \log p(s) ] . \]

- This will be maximized when it reaches to 1, which followed by \( \log p(s) \) become 0.
Conditional Synthesis
Conditional Synthesis

- Controlling image generation process by conditioning the input
  \[ p(s|c) = \prod_i p(s_i|s_{<i}, c). \]

- Condition C can be
  - Class Label
  - Spatial Data – Depth image, low resolution image etc

- Image2Image [2]
  - Conditional synthesis is referenced from Pix2PixGAN
Conditional Image synthesis Process

1. C is the conditional Spatial data
2. Zq = Codebook of Unconditional VQ-GAN
3. S = Index based representation of c from Zq
4. Train a new VQ-GAN on the conditional data
   a. Conditional Codebook Zc
   b. Indexed representation of c from Zc --> r
5. Prepend r with the s
6. Feed this sequence to transformer
7. Obtain synthesized output from Conditional Decoder
The diagram illustrates the process of synthesizing an output image using a conditional codebook and a CNN decoder. The conditional codebooks are labeled $Z_c$ and $Z_q$, with each containing different levels and colors.

The equation $p(s) = \prod_i p(s_i | s_{<i})$ is shown, indicating the conditional probability. The synthesized output is depicted on the right side of the diagram.
Different Conditions - I

Cropped Image

Depth Image
Different Conditions - II

Semantically guided Condition

Pose Estimated Condition
CLIP + VQ-GAN

Text Input - “Old man with a moustache wearing a hat and smoking a cigarette art deco trending on Artstation”
Experiments and Results
How to evaluate quality of generated images?

- Absence of a clear objective for generative models
- Need a score
  - Correlates well with human evaluations
  - Quantify quality of generated images
- Two Scores
  - FID - Fetcher Inception Distance
  - IS - Inception Score
Scores

- **FID Scores:**
  - Calculates similarity between features of original and generated image
  - Lower FID → Better quality image

- **Inception score:**
  - Class prob of generated image by Inception Net [5]
  - Score seeks 2 props:
    - Image Quality
    - Image Diversity
Experiments - 1
Comparison with Conv Model

- Experimental setup
  - Num of Codebook Entries $|Z| = 1024$
  - Predicted sequence length = $256$

- Comparison
  - PIXEL-SNAIL[1]
  - Trained on same codebook
  - Constrained training

- Transformers **outperform** P-SNAIL
Experiments - 2
High Resolution Images

● Sliding Window Attention ➔ High res images beyond 256x
  ○ Upto 1024x576

● **Conditional Synthesis**
  ○ Depth Map
  ○ High Resolution
  ○ Segmentation Map
  ○ Edged Images

● **Unconditional Synthesis**
  ○ LSUN-CT, FaceHQ
  ○ Reduction Factor $f$
Experiments - 3
Class cond. Image Synthesis

- Model param
  - Codebook size = 16384
  - Image size = 256x
- Dataset: ImageNet
- VQ-GAN generates better or comparable quality images like SOTA networks

<table>
<thead>
<tr>
<th>Model</th>
<th>acceptance rate</th>
<th>FID</th>
<th>IS</th>
</tr>
</thead>
<tbody>
<tr>
<td>mixed k, p = 1.0</td>
<td>1.0</td>
<td>17.04</td>
<td>70.6 ± 1.8</td>
</tr>
<tr>
<td>k = 973, p = 1.0</td>
<td>1.0</td>
<td>29.20</td>
<td>47.3 ± 1.3</td>
</tr>
<tr>
<td>k = 250, p = 1.0</td>
<td>1.0</td>
<td>15.98</td>
<td>78.6 ± 1.1</td>
</tr>
<tr>
<td>k = 973, p = 0.88</td>
<td>1.0</td>
<td>15.78</td>
<td>74.3 ± 1.8</td>
</tr>
<tr>
<td>k = 600, p = 1.0</td>
<td>0.05</td>
<td>5.20</td>
<td>280.3 ± 5.5</td>
</tr>
<tr>
<td>mixed k, p = 1.0</td>
<td>0.5</td>
<td>10.26</td>
<td>125.5 ± 2.4</td>
</tr>
<tr>
<td>mixed k, p = 1.0</td>
<td>0.25</td>
<td>7.35</td>
<td>188.6 ± 3.3</td>
</tr>
<tr>
<td>mixed k, p = 1.0</td>
<td>0.05</td>
<td>5.88</td>
<td>304.8 ± 3.6</td>
</tr>
<tr>
<td>mixed k, p = 1.0</td>
<td>0.005</td>
<td>6.59</td>
<td>402.7 ± 2.9</td>
</tr>
<tr>
<td>DCTransformer [48]</td>
<td>n/a</td>
<td>36.5</td>
<td>n/a</td>
</tr>
<tr>
<td>VQVAE-2 [61]</td>
<td>1.0</td>
<td>n/a</td>
<td>330</td>
</tr>
<tr>
<td>VQVAE-2</td>
<td>n/a</td>
<td>~31</td>
<td>~45</td>
</tr>
<tr>
<td>BigGAN [4]</td>
<td>1.0</td>
<td>7.53</td>
<td>168.6 ± 2.5</td>
</tr>
<tr>
<td>BigGAN-deep</td>
<td>1.0</td>
<td>6.84</td>
<td>203.6 ± 2.6</td>
</tr>
<tr>
<td>IDDPM [49]</td>
<td>1.0</td>
<td>12.3</td>
<td>n/a</td>
</tr>
<tr>
<td>ADM-G, no guid. [15]</td>
<td>1.0</td>
<td>10.94</td>
<td>100.98</td>
</tr>
<tr>
<td>ADM-G, 1.0 guid.</td>
<td>1.0</td>
<td>4.59</td>
<td>186.7</td>
</tr>
<tr>
<td>ADM-G, 10.0 guid.</td>
<td>1.0</td>
<td>9.11</td>
<td>283.92</td>
</tr>
<tr>
<td>val. data</td>
<td>1.0</td>
<td>1.62</td>
<td>234.0 ± 3.9</td>
</tr>
</tbody>
</table>

**FID scores for class cond synthesis**
Experiments - 4
Uncond. Image Synthesis

- Comparison with existing generative models
  - GANs, VAEs, Flows, AR, Hybrid
  - Performance depended on dataset

- Decoding strategy
  - Beam Search best results $K=400$
  - VQ-GAN not SOTA results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ours</th>
<th>SPADE [53]</th>
<th>Pix2PixHD (+aug) [75]</th>
<th>CRN [9]</th>
</tr>
</thead>
<tbody>
<tr>
<td>COCO-Stuff</td>
<td>22.4</td>
<td>22.6/23.9(*)</td>
<td>111.5 (54.2)</td>
<td>70.4</td>
</tr>
<tr>
<td>ADE20K</td>
<td>35.5</td>
<td>33.9/35.7(*)</td>
<td>81.8 (41.5)</td>
<td>73.3</td>
</tr>
</tbody>
</table>

FID scores for 256x images $K=100$

<table>
<thead>
<tr>
<th>Method</th>
<th>CelebA-HQ 256 × 256</th>
<th>FFHQ 256 × 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLOW [37]</td>
<td>69.0</td>
<td></td>
</tr>
<tr>
<td>NVAE [69]</td>
<td>40.3</td>
<td></td>
</tr>
<tr>
<td>PIONEER (B.) [23]</td>
<td>39.2 (25.3)</td>
<td></td>
</tr>
<tr>
<td>NCP-VAE [1]</td>
<td>24.8</td>
<td></td>
</tr>
<tr>
<td>VAEBM [77]</td>
<td>20.4</td>
<td></td>
</tr>
<tr>
<td>Style ALAE [56]</td>
<td>19.2</td>
<td></td>
</tr>
<tr>
<td>DC-VAE [54]</td>
<td>15.8</td>
<td></td>
</tr>
<tr>
<td><strong>ours (k=400)</strong></td>
<td><strong>10.2</strong></td>
<td></td>
</tr>
<tr>
<td>PGGAN [31]</td>
<td>8.0</td>
<td></td>
</tr>
</tbody>
</table>

FID scores for 256x images $K=400$

<table>
<thead>
<tr>
<th>Method</th>
<th>CelebA-HQ 256 × 256</th>
<th>FFHQ 256 × 256</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLOW [37]</td>
<td>69.0</td>
<td></td>
</tr>
<tr>
<td>NVAE [69]</td>
<td>40.3</td>
<td></td>
</tr>
<tr>
<td>PIONEER (B.) [23]</td>
<td>39.2 (25.3)</td>
<td></td>
</tr>
<tr>
<td>NCP-VAE [1]</td>
<td>24.8</td>
<td></td>
</tr>
<tr>
<td>VAEBM [77]</td>
<td>20.4</td>
<td></td>
</tr>
<tr>
<td>Style ALAE [56]</td>
<td>19.2</td>
<td></td>
</tr>
<tr>
<td>DC-VAE [54]</td>
<td>15.8</td>
<td></td>
</tr>
<tr>
<td><strong>ours (k=400)</strong></td>
<td><strong>10.2</strong></td>
<td></td>
</tr>
<tr>
<td>PGGAN [31]</td>
<td>8.0</td>
<td></td>
</tr>
<tr>
<td>VDVAE ($t = 0.7$) [11]</td>
<td>38.8</td>
<td></td>
</tr>
<tr>
<td>VDVAE ($t = 1.0$)</td>
<td>33.5</td>
<td></td>
</tr>
<tr>
<td>VDVAE ($t = 0.8$)</td>
<td>29.8</td>
<td></td>
</tr>
<tr>
<td>VDVAE ($t = 0.9$)</td>
<td>28.5</td>
<td></td>
</tr>
<tr>
<td>VQGAN+P.SNAIL</td>
<td>21.9</td>
<td></td>
</tr>
<tr>
<td>BigGAN</td>
<td>12.4</td>
<td></td>
</tr>
<tr>
<td><strong>ours (k=300)</strong></td>
<td><strong>9.6</strong></td>
<td></td>
</tr>
<tr>
<td>U-Net GAN (+aug) [66]</td>
<td>10.9 (7.6)</td>
<td></td>
</tr>
<tr>
<td>StyleGAN2 (+aug) [34]</td>
<td>3.8 (3.6)</td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

- Combines strength of Convolutional networks with transformers
  - Local correlations - Conv
  - Long Range dependencies - Transformers

- Code book or Quantized vectors instead of pixels
  - Long range dependencies
  - No more Quadratic Computational complexity

- Proposed hybrid approach - Outperform SOTA convolutional approaches

- BottleNeck:
  - Training additional VQ-GAN for conditional synthesis
References


