1. GANs and StyleGANs
2. AutoEncoders
Progress for Image Generation

2014: GAN  
2015: DCGAN  
2017: PG-GAN  
2018: StyleGAN  
2018: BigGAN  
2019: StyleGANv2  
2020: StyleGAN-ADA  

2020: NeRF (Neural Radiance Fields)

2021: OpenAI DALL-E (VQ-VAE)

Arm chair in shape of avocado

Bolei, Zhou, Tutorial on Interpretable Machine Learning for Computer Vision at CVPR 2021
Progress for Image Generation

An astronaut riding a horse in a photorealistic style.

Google Imagen https://imagen.research.google/
Meta Make-A-Video https://makeavideo.studio/
GPT-4 ????
Generative Adversarial Networks (GANs)
Generative Adversarial Networks (GANs)

- The basic idea of GANs is to set up a game between two players.
  - **Generator**
    - Creates samples that are intended to come from the same distribution as the training data
    - The counterfeiter: Trained to fool the discriminator
  - **Discriminator**
    - Examines samples to determine whether they are real or fake
    - The police: Trained to distinguish between the generated or the real (training data)

- Formally, GANs are a structured probabilistic model containing latent variables $z$ and observed variables $x$. 
Generative Adversarial Networks (GANs)

Generator

Fake bedroom

Real/Fake

Discriminator

Real bedroom

training data

Goodfellow et al. NeurIPS’14
Generative Adversarial Networks (GANs)

<table>
<thead>
<tr>
<th>Latent</th>
<th>Generator</th>
<th>Fake bedroom</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Real/Fake</th>
<th>Discriminator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Real bedroom

Goodfellow et al. NeurIPS’14
Generative Adversarial Networks (GANs)

typically drawn from a pre-defined distribution (eg. Gaussian)

Goodfellow et al. NeurIPS’14
Generative Adversarial Networks (GANs)
Generative Adversarial Networks (GANs)
StyleGANs

Generator

(a) Traditional

(b) Style-based generator

Tero Karras et al., A Style-Based Generator Architecture for Generative Adversarial Networks, CVPR 2019
StyleGAN

Adain

adaptive instance normalization

Adain(x,y) = ys,i \frac{x_i - \mu(x_i)}{\sigma(x_i)} + yb,i,
StyleGANs

- StyleGAN embeds the input latent code $z$ into an intermediate latent space $w$
  - $w = F(z)$
- Now it is $w$, not $z$, that controls the style of the generated images

Tero Karras et al., A Style-Based Generator Architecture for Generative Adversarial Networks, CVPR 2019
StyleGANs

- StyleGAN embeds the input latent code \( z \) into an intermediate latent space \( w \)
  - \( w = F(z) \)
- Now it is \( w \), not \( z \), that controls the style of the generated images
  - The mapping \( F \) can “unwarp” \( z \) to \( w \) so that the factors of variation become more linear.
  - Style: factors of variation of the domain of interest

Tero Karras et al., A Style-Based Generator Architecture for Generative Adversarial Networks, CVPR 2019
StyleGANs - The Disentangled Latent Space

- There are various definitions for disentanglement.
- A common goal is a latent space that consists of linear subspaces, each of which controls one factor of variation.

Zongze Wu et al., StyleSpace Analysis: Disentangled Controls for StyleGAN Image Generation, CVPR 2021
Which latent space is more disentangled?

Reconstruction Error (Xu et al. CVPR’21)

<table>
<thead>
<tr>
<th>Space</th>
<th>MSE</th>
<th>FID</th>
</tr>
</thead>
<tbody>
<tr>
<td>W space</td>
<td>0.0601</td>
<td>22.24</td>
</tr>
<tr>
<td>S space</td>
<td>0.0464</td>
<td>18.48</td>
</tr>
</tbody>
</table>

Disentanglement (Wu et al. CVPR’21)

<table>
<thead>
<tr>
<th>Space</th>
<th>Disentanglement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z space</td>
<td>0.31</td>
</tr>
<tr>
<td>W space</td>
<td>0.54</td>
</tr>
<tr>
<td>S space</td>
<td>0.75</td>
</tr>
</tbody>
</table>
GAN Inversion

Applying the pretrained GAN model to image processing tasks

GAN inversion:
\[ x^* = \text{argmin}_x \| G(x) - I \| \]

Colorization:
\[ x^* = \text{argmin}_x \| \text{rgb2gray}(G(x)) - I_{\text{gray}} \| \]

Super-resolution:
\[ x^* = \text{argmin}_x \| \text{down}(G(x)) - I_{\text{small}} \| \]

Masked optimization
\[ x^* = \text{argmin}_x \| m \cdot G(x) - m \cdot I_{\text{context}} \| \]

- (a) Image Reconstruction
- (b) Image Colorization
- (d) Image Denoising
- (e) Image Inpainting


Zhu, Shen, Zhao, Zhou. *In-domain GAN Inversion*. ECCV’20

Bolei, Zhou, Tutorial on Interpretable Machine Learning for Computer Vision at CVPR 2021
Encoding Real Image into StyleGAN space

\[ I \rightarrow \text{Encoder} \rightarrow x = E(I) \rightarrow \text{StyleGAN Generator} \rightarrow I' = G(x) \]
AutoEncoders

https://lilianweng.github.io/posts/2018-08-12-vae/
AutoEncoders

- Encoder
  - Transforms the original high-dimension input (e.g., images) into the low-dimensional latent.
  - Hopefully lossless

- Decoder
  - Recovers the high-dimensional data from the encoded low-dimensional latents

- Dimensionality Reduction
  - Links to PCA

$$L_{AE}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} (x^{(i)} - f_{\theta}(g_{\phi}(x^{(i)})))^2$$
Variational AutoEncoders

● Vanilla autoencoder’s latent space is NOT well-organized/structured to be sampled from
  ○ Because there is no force for the latent space to do so

● Variational AutoEncoders
  ○ Autoencoders whose latent space is regularized to a structured distribution (e.g., Gaussian distribution)
  ○ The latent is now a distribution
Variational AutoEncoders

\[ L_{VAE}(\theta, \phi) = - \log p_\theta(x) + D_{KL}(q_\phi(z|x) \| p_\theta(z|x)) \]
\[ = - \mathbb{E}_{z \sim q_\phi(z|x)} \log p_\theta(x|z) + D_{KL}(q_\phi(z|x) \| p_\theta(z)) \]

https://miro.medium.com/max/1400/1*ejNnusxYrn1NRDZf4Kg2lw@2x.webp
Variational AutoEncoders

The reconstruction term:

$$L_{VAE}(\theta, \phi) = - \log p_\theta(x) + D_{KL}(q_\phi(z|x) || p_\theta(z|x))$$

$$= -E_{z \sim q_\phi(z|x)} \log p_\theta(x|z) + D_{KL}(q_\phi(z|x) || p_\theta(z))$$
Variational AutoEncoders

\[ L_{VAE}(\theta, \phi) = -\log p_\theta(x) + D_{KL}(q_\phi(z|x)\|p_\theta(z|x)) \]

the regularization term:  
KL divergence to the prior distribution
Summary

**GAN**: Adversarial training

**VAE**: maximize variational lower bound

**Diffusion models**: Gradually add Gaussian noise and then reverse

Recall Fidler’s 3D Neural Rendering Approach

\[ I \leftrightarrow W \leftrightarrow z \leftrightarrow I \]

- The whole system is an autoencoder
- \( I \rightarrow W \): An autoencoder approach where the decoder is a differentiable renderer \( I = F(W; \alpha) \) and \( P_\varphi(W \mid I) \) is the encoder.
- \( z \rightarrow I \): A learned StyleGAN generative model \( P_\theta(I \mid z) \)
- \( W \rightarrow z \): Learn \( f_\psi(z \mid W) \) using an autoencoder reconstruction loss
  - This can be done by the autoencoder because the latent variables of styleGANs are fairly interpretable and so the function \( f_\psi(z \mid W) \) cannot be too complicated