GANs part 3: Algorithms and Applications

Yossi Keshet
Bar-Ilan University

May 31, 2021
GAN: Generative Adversarial Network

$z = (0.3, 0.2, -0.6, \ldots) \xrightarrow{G(z)} \text{a blue round cup}$

$z \sim \mathcal{N}(0, 1)$
or
$z \sim \mathcal{U}(-1, 1)$

$z = (-0.1, 0.1, 0.2, \ldots) \xrightarrow{G(z)} \text{a yellow tall cup}$
GAN: Generative Adversarial Network
GAN: Generative Adversarial Network

\[
\max_D V(D) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]
\]

- recognize real images better
- recognize generated images better
GAN: Generative Adversarial Network

$$\max_D V(D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_{z}(z)}[\log(1 - D(G(z)))]$$

- recognize real images better
- recognize generated images better

$$\min_G V(G) = \mathbb{E}_{z \sim p_{z}(z)}[\log(1 - D(G(z)))]$$

Optimize $G$ that can fool the discriminator the most.
GAN: Generative Adversarial Network

\[
\max_D V(D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]
\]

recognize real images better \hspace{1cm} \text{recognize generated images better}

\[
\min_G V(G) = \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]
\]

Optimize \( G \) that can fool the discriminator the most.

\[-\nabla_{\theta_g} \log \left(1 - D \left( G \left( z^{(i)} \right) \right) \right) \rightarrow 0 \hspace{0.5cm} \text{change to} \hspace{0.5cm} \nabla_{\theta_g} \log \left( D \left( G \left( z^{(i)} \right) \right) \right)\]
Last lectures

- Basic GAN
  - Problem in the generative loss
- Deep Convolutional GAN (DCGAN)
- Analysis of GAN convergence
- Noisy GAN
- Wasserstein GAN
Today
Today

• Measuring performance of GAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)

- Conditional GAN

- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
  - StarGAN (will not be covered)
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)

- Conditional GAN

- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
  - StarGAN (will not be covered)

- High quality image generation
  - Progressive GAN
  - StackGAN
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
  – StyleGAN and StyleGAN2 (will not be covered)
• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
  – StyleGAN and StyleGAN2 (will not be covered)
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
  - StarGAN (will not be covered)
- High quality image generation
  - Progressive GAN
  - StackGAN
  - StyleGAN and StyleGAN2 (will not be covered)
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
  – StyleGAN and StyleGAN2 (will not be covered)
Measuring GAN Performance

How do we measure the GAN performance?
Measuring GAN Performance

How do we measure the GAN performance?

• Image quality
Measuring GAN Performance

How do we measure the GAN performance?

• Image quality
Measuring GAN Performance

How do we measure the GAN performance?

- Image quality

- The images have variety (e.g. each image is a different breed of dog)
Measuring GAN Performance

How do we measure the GAN performance?

- Image quality

- The images have variety (e.g. each image is a different breed of dog)
Measuring GAN Performance

How do we measure the GAN performance?

- Image quality

- The images have variety (e.g. each image is a different breed of dog)
Measuring GAN Performance

How do we measure the GAN performance?

- Image quality

- The images have variety (e.g. each image is a different breed of dog)
Measuring GAN Performance

How do we measure the GAN performance?

• Image quality

• The images have variety (e.g. each image is a different breed of dog)

• Each image distinctly looks like something (e.g. one image is clearly a Poodle, the next a great example of a French Bulldog)

but not combined:
Measuring GAN Performance: Inception Score (IS)

The image contains one distinct object, then the output of the classifier is a narrow distribution, e.g. focused on one peak.
Measuring GAN Performance: Inception Score (IS)

The image contains one distinct object, then the output of the classifier is a narrow distribution, e.g. focused on one peak.

The image is a jumble, or contains multiple things, it’s closer to the ‘uniform’ distribution of many similar height bars (e.g. it’s equally likely to be any of the labels).
Measuring GAN Performance: Inception Score (IS)

We average over many examples generated from our GAN:

Similar labels sum to give focused distribution

Different labels sum to give uniform distribution
Measuring GAN Performance: Inception Score (IS)

We want each image to be distinct (left below) and to collectively have variety (right below). These ideal distributions are opposite shapes, the label distribution is narrow, the marginal distribution is uniform.
Measuring GAN Performance: Inception Score (IS)

Salimans et al., Improved Techniques for Training GANs, 2016
Measuring GAN Performance: Inception Score (IS)

To produce this score we use the KL-divergence:

Salimans et al., Improved Techniques for Training GANs, 2016
Measuring GAN Performance: Inception Score (IS)

Images that contain meaningful objects should have a conditional label distribution $p(y|x)$ with low entropy.

We expect the model to generate varied images, so the marginal $p(y)$ should have high entropy:

$$p(y) = \int p(y|x = G(z)) \, dz$$

Salimans et al., Improved Techniques for Training GANs , 2016
Measuring GAN Performance: Inception Score (IS)

Images that contain meaningful objects should have a conditional label distribution \( p(y|x) \) with low entropy.

We expect the model to generate varied images, so the marginal \( p(y) \) should have high entropy:

\[
p(y) = \int p(y|x = G(z)) \, dz
\]

Combining these two requirements, the metric that we propose is:

Salimans et al., Improved Techniques for Training GANs, 2016
Measuring GAN Performance: Inception Score (IS)

Images that contain meaningful objects should have a conditional label distribution $p(y|x)$ with low entropy. We expect the model to generate varied images, so the marginal $p(y)$ should have high entropy:

$$p(y) = \int p(y|x = G(z)) \, dz$$

Combining these two requirements, the metric that we propose is:

$$IS(G) = \exp \left\{ \mathbb{E}_x \left[ D_{KL}(p(y|x) \| p(y)) \right] \right\}$$

Salimans et al., Improved Techniques for Training GANs, 2016
Measuring GAN Performance: Inception Score (IS)

Images that contain meaningful objects should have a conditional label distribution $p(y \mid x)$ with low entropy. We expect the model to generate varied images, so the marginal $p(y)$ should have high entropy:

$$p(y) = \int p(y \mid x = G(z)) \, dz$$

Combining these two requirements, the metric that we propose is:

$$IS(G) = \exp \left\{ \mathbb{E}_x \left[ D_{KL}(p(y \mid x) \parallel p(y)) \right] \right\}$$

The exponent is there so the values are easier to compare.

Salimans et al., Improved Techniques for Training GANs, 2016
Measuring GAN Performance: Inception Score (IS)

The limitations of IS:
Measuring GAN Performance: Inception Score (IS)

The limitations of IS:

- If you’re learning to generate something not present in the classifier’s training data - then you may always get low IS despite generating high quality images since that image doesn’t get classified as a distinct class.
Measuring GAN Performance: Inception Score (IS)

The limitations of IS:

- If you’re learning to **generate something not present in the classifier’s training data** - then you may always get low IS despite generating high quality images since that image doesn’t get classified as a distinct class.

- If you’re generating images with a **different set of labels from the classifier training set** (say, you’re training the GAN to generate different varieties of poodles, or just elephants and ants) it can score lowly.
Measuring GAN Performance: Inception Score (IS)

The limitations of IS:

- If you’re learning to **generate something not present in the classifier’s training data** - then you may always get low IS despite generating high quality images since that image doesn’t get classified as a distinct class.

- If you’re generating images with a **different set of labels from the classifier training set** (say, you’re training the GAN to generate different varieties of poodles, or just elephants and ants) it can score lowly.

- **If the classifier network cannot detect features relevant to your concept of image quality** (e.g. there is evidence that CNNs rely heavily on local image textures for classification, and coarse shapes do not matter so much), then poor quality images may still get high scores. For example, you might generate people with two heads, but not get penalized for it.
Measuring GAN Performance: Inception Score (IS)

The limitations of IS:

- If you’re learning to **generate something not present in the classifier’s training data** - then you may always get low IS despite generating high quality images since that image doesn’t get classified as a distinct class.

- If you’re generating images with a **different set of labels from the classifier training set** (say, you’re training the GAN to generate different varieties of poodles, or just elephants and ants) it can score lowly.

- **If the classifier network cannot detect features relevant to your concept of image quality** (e.g. there is evidence that CNNs rely heavily on local image textures for classification, and coarse shapes do not matter so much), then poor quality images may still get high scores. For example, you might generate people with two heads, but not get penalized for it.

- If your **generator generates only one image per classifier image class**, repeating each image many times, it can score highly (i.e. there is no measure of intra-class diversity).
The limitations of IS:

- If you’re learning to **generate something not present in the classifier’s training data** - then you may always get low IS despite generating high quality images since that image doesn’t get classified as a distinct class.

- If you’re generating images with a **different set of labels from the classifier training set** (say, you’re training the GAN to generate different varieties of poodles, or just elephants and ants) it can score lowly.

- **If the classifier network cannot detect features relevant to your concept of image quality** (e.g. there is evidence that CNNs rely heavily on local image textures for classification, and coarse shapes do not matter so much), then poor quality images may still get high scores. For example, you might generate people with two heads, but not get penalized for it.

- If your **generator generates only one image per classifier image class**, repeating each image many times, it can score highly (i.e. there is no measure of intra-class diversity).

- If your **generator memorizes the training data and replicates it**, it can score highly.
Measuring GAN Performance: Fréchet Inception Distance (FID)

Fréchet proposed the following distance between distributions $f$ and $g$ of random variables $X$ and $Y$, respectively:

$$d^2(f, g) = \min_{X,Y} \mathbb{E} \left[ |X - Y|^2 \right]$$

Unlike the earlier inception score (IS), which evaluates only the distribution of generated images, the FID compares the distribution of generated images with the distribution of real images that were used to train the generator.
Measuring GAN Performance: Fréchet Inception Distance (FID)

Fréchet proposed the following distance between distributions $f$ and $g$ of random variables $X$ and $Y$, respectively:

$$d^2(f, g) = \min_{X,Y} \mathbb{E} \left[ |X - Y|^2 \right]$$

For two Normal distributions:

$$d^2(f, g) = (\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

Unlike the earlier inception score (IS), which evaluates only the distribution of generated images, the FID compares the distribution of generated images with the distribution of real images that were used to train the generator
Measuring GAN Performance: Fréchet Inception Distance (FID)

Fréchet proposed the following distance between distributions $f$ and $g$ of random variables $X$ and $Y$, respectively:

$$d^2(f, g) = \min_{X, Y} \mathbb{E} \left[ |X - Y|^2 \right]$$

For two Normal distributions:

$$d^2(f, g) = (\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

For two multivariate (vectors) Normal distributions:

$$d^2(f, g) = (\mu_X - \mu_Y)^2 + \text{tr} \left[ \Sigma_X + \Sigma_Y - 2 \left( \Sigma_X \Sigma_Y \right)^{1/2} \right]$$

Unlike the earlier inception score (IS), which evaluates only the distribution of generated images, the FID compares the distribution of generated images with the distribution of real images that were used to train the generator.
Figure 3: FID is evaluated for upper left: Gaussian noise, upper middle: Gaussian blur, upper right: implanted black rectangles, lower left: swirled images, lower middle: salt and pepper noise, and lower right: CelebA dataset contaminated by ImageNet images. The disturbance level rises from zero and increases to the highest level. The FID captures the disturbance level very well by monotonically increasing.
# Measuring GAN Performance

<table>
<thead>
<tr>
<th>Paper</th>
<th>IS</th>
<th>FID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real CIFAR-10 data (Salimans et al., 2016)</td>
<td>11.24</td>
<td>–</td>
</tr>
<tr>
<td>Unsupervised representation learning with deep convolutional generative adversarial networks (DCGAN) (Radford, Metz, &amp; Chintala, 2015)</td>
<td>6.16</td>
<td>37.1</td>
</tr>
<tr>
<td>Conditional image generation with PixelCNN decoders (van den Oord et al., 2016)</td>
<td>4.60</td>
<td>65.9</td>
</tr>
<tr>
<td>Adversarially learned inference (ALI) (Dumoulin et al., 2016)</td>
<td>5.34</td>
<td>–</td>
</tr>
<tr>
<td>Improved techniques for training GANs (Salimans et al., 2016)</td>
<td>6.86</td>
<td>–</td>
</tr>
<tr>
<td>Improving generative adversarial networks with denoising feature matching (Warde-Farley &amp; Bengio, 2016)</td>
<td>7.72</td>
<td>–</td>
</tr>
<tr>
<td>Learning to generate samples from noise through infusion training (Bordes, Honari, &amp; Vincent, 2017)</td>
<td>4.62</td>
<td>–</td>
</tr>
<tr>
<td>BEGAN: Boundary equilibrium generative adversarial networks (Berthelot, Schumm, &amp; Metz, 2017)</td>
<td>5.62</td>
<td>–</td>
</tr>
<tr>
<td>MMD GAN: Towards deeper understanding of moment matching network (Li, Chang, Cheng, Yang, &amp; Páczos, 2017)</td>
<td>6.17</td>
<td>–</td>
</tr>
<tr>
<td>Improved training of Wasserstein GANs (Gulrajani, Ahmed, Arjovsky, Dumoulin, &amp; Courville, 2017)</td>
<td>7.86</td>
<td>–</td>
</tr>
<tr>
<td>Coulomb GANs: Provably optimal Nash equilibrium via potential fields (Unterthiner et al., 2017)</td>
<td>–</td>
<td>27.3</td>
</tr>
<tr>
<td>GANs trained by a two time-scale update rule converge to a local Nash equilibrium (Heusel, Ramsauer, Unterthiner, Nessler, &amp; Hochreiter, 2017)</td>
<td>–</td>
<td>24.8</td>
</tr>
<tr>
<td>Autoregressive quantile networks for generative modeling (AIQN) (Ostrovski, Dabney, &amp; Munos, 2018)</td>
<td>5.29</td>
<td>49.5</td>
</tr>
<tr>
<td>Learning implicit generative models with the method of learned moments (Ravuri, Mohamed, Rosca, &amp; Vinyals, 2018)</td>
<td>7.90</td>
<td>18.9</td>
</tr>
</tbody>
</table>
• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• **Conditional GAN**

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
  – StyleGAN and StyleGAN2 (will not be covered)
Conditional GANs (CGANs) are an extension of the GANs model. CGANs are allowed to generate images that have certain conditions or attributes.

- The Generator and Discriminator both receive some additional conditioning input information. This could be the class of the current image or some other property.
- For example, if we train a DCGANs to generate new MNIST images, there is no control over which specific digits will be produced by the Generator. CGAN can be conditioned on the specific digit.
Conditional GANs (CGANs)

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))].$$
Conditional GANs (CGANs)

\[ \min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]. \]
Conditional GANs (CGANs)

\[
\begin{align*}
\min_G \max_D V(D, G) &= \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))].
\end{align*}
\]

\[
\begin{align*}
\min_G \max_D V(D, G) &= \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x|y)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|y)))].
\end{align*}
\]

\[y\] could be any kind of auxiliary information, such as class labels or data from other modalities. We can perform the conditioning by feeding \[y\] into the both the discriminator and generator as additional input layer.
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- GAN for cross-domain transfer
  - Pix2Pix networks: image-to-image translation
  - CycleGAN
  - StarGAN
- Energy-Based GAN (EBGAN)
- InfoGAN?
- Progressive GAN and StackGAN
Image-to-image translation

- Labels to Street Scene
  - Input
  - Output

- Labels to Facade
  - Input
  - Output

- BW to Color
  - Input
  - Output

- Aerial to Map
  - Input
  - Output

- Day to Night
  - Input
  - Output

- Edges to Photo
  - Input
  - Output
Pix2Pix networks
Image-To-Image Translation is a process for translating one representation of an image into another representation.

$x$ - input image, $y$ - output image, $z$ - random noise vector
Pix2Pix networks

- Image-To-Image Translation is a process for translating one representation of an image into another representation. 
  \( x \) - input image, \( y \) - output image, \( z \) - random noise vector
- Pix2Pix network is basically a Conditional GANs (cGAN) that learn the mapping from an input image to output an image

\[
\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y} [\log D(x, y)] + \mathbb{E}_{x,z} [\log (1 - D(x, G(x, z)))]
\]
Pix2Pix networks

- Image-To-Image Translation is a process for translating one representation of an image into another representation.
  - input image, y - output image, z - random noise vector

- Pix2Pix network is basically a Conditional GANs (cGAN) that learn the mapping from an input image to output an image

\[
L_{cGAN}(G, D) = \mathbb{E}_{x, y} \left[ \log D(x, y) \right] + \mathbb{E}_{x, z} \left[ \log (1 - D(x, G(x, z))) \right]
\]
Pix2Pix networks

• Image-To-Image Translation is a process for translating one representation of an image into another representation.
  \( x \) - input image, \( y \) - output image, \( z \) - random noise vector

• Pix2Pix network is basically a Conditional GANs (cGAN) that learn the mapping from an input image to output an image

\[
\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]
\]
Pix2Pix networks

• Image-To-Image Translation is a process for translating one representation of an image into another representation.
  \( x \) - input image, \( y \) - output image, \( z \) - random noise vector

• Pix2Pix network is basically a Conditional GANs (cGAN) that learn the mapping from an input image to output an image

\[
\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]
\]

• Additionally, the Generator output should be close to ground truth for each pixel in \( \ell_1 \)

\[
\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]
\]
Pix2Pix networks

- Image-To-Image Translation is a process for translating one representation of an image into another representation.
  - input image, $y$ - output image, $z$ - random noise vector

- Pix2Pix network is basically a Conditional GANs (cGAN) that learn the mapping from an input image to output an image

\[
\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y} \left[ \log D(x, y) \right] + \mathbb{E}_{x,z} \left[ \log (1 - D(x, G(x, z))) \right]
\]

- Additionally, the Generator output should be close to ground truth for each pixel in $\ell_1$

\[
\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z} [\| y - G(x, z) \|_1]
\]
Pix2Pix networks

- Image-To-Image Translation is a process for translating one representation of an image into another representation. 
  \( x \) - input image, \( y \) - output image, \( z \) - random noise vector

- Pix2Pix network is basically a Conditional GANs (cGAN) that learn the mapping from an input image to output an image

\[
\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))] 
\]

- Additionally, the Generator output should be close to ground truth for each pixel in \( \mathcal{L}_1 \)

\[
\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1] 
\]

- Overall:

\[
G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) 
\]
Pix2Pix networks: Generator
Pix2Pix networks: Generator

- The **Generator** is implemented as a **U-Net**. **U-Net** is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.
Pix2Pix networks: Generator

- The **Generator** is implemented as a **U-Net**. **U-Net** is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.
Pix2Pix networks: Generator

- The **Generator** is implemented as a **U-Net**. **U-Net** is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.
- Skip connections between each layer $i$ and layer $n - i$, where $n$ is the total number of layers.
Pix2Pix networks: Generator

- The **Generator** is implemented as a **U-Net**. **U-Net** is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.
- Skip connections between each layer \( i \) and layer \( n - i \), where \( n \) is the total number of layers.
- Each skip connection simply concatenates all channels at layer \( i \) with those at layer \( n - i \).
Pix2Pix networks: Discriminator
Pix2Pix networks: Discriminator

- To avoid blurry results in images - we should improve the high-frequencies in the images.
• To avoid blurry results in images - we should improve the **high-frequencies** in the images.

• It is sufficient to restrict our attention to the structure in local image patches —> **PatchGAN** only penalizes structure at the scale of patches
Pix2Pix networks: Discriminator

- To avoid blurry results in images - we should improve the **high-frequencies** in the images.
- It is sufficient to restrict our attention to the structure in local image patches —> **PatchGAN** only penalizes structure at the scale of patches.
Pix2Pix networks: Discriminator

- To avoid blurry results in images - we should improve the **high-frequencies** in the images.
- It is sufficient to restrict our attention to the structure in local image patches —> **PatchGAN** only penalizes structure at the scale of patches
- This discriminator tries to classify if each $N \times N$ patch in an image is real or fake.
Pix2Pix networks: Discriminator

• To avoid blurry results in images - we should improve the **high-frequencies** in the images.

• It is sufficient to restrict our attention to the structure in local image patches —> **PatchGAN** only penalizes structure at the scale of patches.

• This discriminator tries to classify if each $N \times N$ patch in an image is real or fake.

• This discriminator is applied by a convolution across the image, and then by averaging all responses to provide the ultimate output of D.
Pix2Pix networks: Examples

Different losses induce different quality of results. Each column shows results trained under a different loss.
Pix2Pix networks: Examples

Adding skip connections to an encoder-decoder to create a “U-Net” results in much higher quality results.
Pix2Pix networks: Examples

Example results on Google Maps at 512x512 resolution (model was trained on images at 256 × 256 resolution, and run convolutionally on the larger images at test time). Contrast adjusted for clarity.
Pix2Pix networks: Examples

Colorization results of conditional GANs versus the L2 regression from
Example results of our method on facades labels→photo, compared to ground truth.
Example results of our method on automatically detected edges→handbags, compared to ground truth.
Figure 16: Example results of the edges→photo models applied to human-drawn sketches from [10]. Note that the models were trained on automatically detected edges, but generalize to human drawings.
Today

- Measuring performance of GAN
  - Inception Score (IS)
  - Fréchet Inception Distance (FID)
- Conditional GAN
- **GAN for cross-domain transfer**
  - Pix2Pix networks: image-to-image translation
  - **CycleGAN**
    - StarGAN (will not be covered)
- High quality image generation
  - Progressive GAN
  - StackGAN
  - StyleGAN and StyleGAN2 (will not be covered)
CycleGAN

Monet $\iff$ Photos

Monet $\rightarrow$ photo
CycleGAN

Monet $\leftrightarrow$ Photos

Monet $\rightarrow$ photo

photo $\rightarrow$ Monet
CycleGAN

Monet ↔ Photos

Monet → photo

photo → Monet

Summer ↔ Winter

summer → winter

winter → summer
CycleGAN

... and turning horse into a Zebra
CycleGAN

... and turning horse into a Zebra
In image-to-image translation (pix2pix) the training set was paired.
CycleGAN
CycleGAN

- The model consists of two mapping functions (Generators):
  \[G : X \rightarrow Y\] and \[F : Y \rightarrow X\]
CycleGAN

- The model consists of two mapping functions (Generators): \( G : X \rightarrow Y \) and \( F : Y \rightarrow X \)
- It is associated with two discriminators: \( D_X \) and \( D_Y \) encourages \( G \) to translate \( X \) into outputs indistinguishable from domain \( Y \), and vice versa for \( D_X \) and \( F \).
CycleGAN

- The model consists of two mapping functions (Generators): $G : X \rightarrow Y$ and $F : Y \rightarrow X$
- It is associated with two discriminators: $D_X$ and $D_Y$ encourages $G$ to translate $X$ into outputs indistinguishable from domain $Y$, and vice versa for $D_X$ and $F$.
- Forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$
The model consists of two mapping functions (Generators): $G : X \rightarrow Y$ and $F : Y \rightarrow X$.

It is associated with two discriminators: $D_X$ and $D_Y$ encourages $G$ to translate $X$ into outputs indistinguishable from domain $Y$, and vice versa for $D_X$ and $F$.

Forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$

Backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$
CycleGAN

- The model consists of two mapping functions (Generators): \( G : X \rightarrow Y \) and \( F : Y \rightarrow X \)
- It is associated with two discriminators: \( D_X \) and \( D_Y \) encourages \( G \) to translate \( X \) into outputs indistinguishable from domain \( Y \), and vice versa for \( D_X \) and \( F \).
- Forward cycle-consistency loss: \( x \rightarrow G(x) \rightarrow F(G(x)) \approx x \)
- Backward cycle-consistency loss: \( y \rightarrow F(y) \rightarrow G(F(y)) \approx y \)
Reconstruction cost: This is the Cycle consistency loss which measures the L1-norm reconstruction cost for the real image \( (x \to y \to \text{reconstructed } x) \) and the Monet paintings \( (y \to x \to \text{reconstructed } y) \)

\[
\mathcal{L}_{\text{cyc}}(G,F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]
\]
**CycleGAN**

**Reconstruction cost:** This is the Cycle consistency loss which measures the L1-norm reconstruction cost for the real image ($x \rightarrow y \rightarrow$ reconstructed $x$) and the Monet paintings ($y \rightarrow x \rightarrow$ reconstructed $y$)

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim \mathcal{P}_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim \mathcal{P}_{\text{data}}(y)}[\|G(F(y)) - y\|_1]$$

**Adversary loss**

$$\mathcal{L}_{\text{GAN}}(G, D, X, Y)$$

- For $G$, minimize $\mathbb{E}_{x \sim \mathcal{P}_{\text{data}}(x)}[(D(G(x)) - 1)^2]$
- For $D$, minimize $\mathbb{E}_{y \sim \mathcal{P}_{\text{data}}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim \mathcal{P}_{\text{data}}(x)}[D(G(x))^2]$
CycleGAN

**Reconstruction cost:** This is the Cycle consistency loss which measures the L1-norm reconstruction cost for the real image (x → y → reconstructed x) and the Monet paintings (y → x → reconstructed y)

\[ \mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1] \]

**Adversary loss**

\[ \mathcal{L}_{\text{GAN}}(G, D, X, Y) \]
- For G, minimize \( \mathbb{E}_{x \sim p_{\text{data}}(x)}[(D(G(x)) - 1)^2] \)
- For D, minimize \( \mathbb{E}_{y \sim p_{\text{data}}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[D(G(x))^2] \)

The final objective function:

\[ \mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F), \]
**CycleGAN**

**Reconstruction cost:** This is the Cycle consistency loss which measures the L1-norm reconstruction cost for the real image (x → y → reconstructed x) and the Monet paintings (y → x → reconstructed y)

\[
\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]
\]

**Adversary loss**

\[
\mathcal{L}_{\text{GAN}}(G, D, X, Y)
\]

- For G, minimize \( \mathbb{E}_{x \sim p_{\text{data}}(x)}[(D(G(x)) - 1)^2] \)
- For D, minimize \( \mathbb{E}_{y \sim p_{\text{data}}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[D(G(x))^2] \)

The **final** objective function:

\[
\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F),
\]

*PyTorch implementation: [https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix](https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix)*
CycleGAN: Examples
CycleGAN: Examples

Mapping Monet's paintings to a photographic style
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
  – StyleGAN and StyleGAN2 (will not be covered)
Progressive growing of GANs

Figure 5: $1024 \times 1024$ images generated using the \texttt{CELEBA-HQ} dataset. See Appendix F for a larger set of results, and the accompanying video for latent space interpolations.
Progressive growing of GANs


Figure 6: Visual quality comparison in LSUN BEDROOM; pictures copied from the cited articles.
Progressive growing of GANs

It applies the strategy of divide-and-conquer to make training much feasible. Layers of convolution layers are trained once at a time to build images of $2x$ resolution.
Progressive growing of GANs
Both the generator (G) and discriminator (D) having a low spatial resolution of 4×4 pixels
Progressive growing of GANs
Progressive growing of GANs

Layers are incrementally added to G and D.
Progressive growing of GANs
Progressive growing of GANs

All existing layers remain trainable throughout the process.
Progressive growing of GANs
Progressive growing of GANs

In phase 9 a $1024 \times 1024$ image is generated.
Progressive growing of GANs
Progressive growing of GANs

(a) 16x16
    └── toRGB

(b) 16x16
    └── 2x
        ├── 16x16
        │    └── toRGB
        │         └── 1-α + α
        │                   └── 32x32
        │                                └── toRGB
        │                                           └── 0.5x
        │                                                └── fromRGB
        │                                                    └── 32x32
        │                                                        └── 0.5x
        │                                                                └── fromRGB
        │                                                                                     └── 16x16

(c) 16x16
    └── 2x
        └── 32x32
            └── 0.5x
                └── fromRGB
                    └── 16x16
Progressive growing of GANs

When doubling the resolution of the generator (G) and discriminator (D) - new layers are faded in smoothly.

(a) G: 16x16 to toRGB
(b) D: 16x16 fromRGB 0.5x 32x32 fromRGB 32x32 fromRGB 0.5x 16x16
(c) D: 16x16 fromRGB 2x 32x32 fromRGB 32x32 fromRGB 0.5x 16x16
Progressive growing of GANs
The layers that operate on the higher resolution are treated like a residual block, whose weight $\alpha$ increases linearly from 0 to 1.
Progressive growing of GANs

(a) G: 16x16 -> toRGB -> 16x16

(b) D: 16x16 -> fromRGB -> 32x32 -> 16x16

(c) D: 32x32 -> fromRGB -> 16x16
Progressive growing of GANs

Here 2x and 0.5x refer to doubling and halving the image resolution using nearest neighbor filtering and average pooling, respectively.
Progressive growing of GANs

G

16x16

\rightarrow toRGB

\downarrow

16x16

D

fromRGB

\downarrow

16x16

(a)

16x16

\rightarrow toRGB

\downarrow

32x32

\rightarrow toRGB

\downarrow

32x32

\rightarrow toRGB

\downarrow

16x16

\rightarrow 1-\alpha + \alpha

\rightarrow +

\rightarrow 0.5 \times

\rightarrow fromRGB

\downarrow

16x16

(b)

16x16

\rightarrow 2 \times

\rightarrow 32x32

\rightarrow 1-\alpha + \alpha

\rightarrow +

\rightarrow 0.5 \times

\rightarrow fromRGB

\downarrow

16x16

\rightarrow 16x16

(c)
The `toRGB` represents a layer that projects feature vectors to RGB colors and `fromRGB` does the reverse; both use 1×1 convolutions.
Progressive growing of GANs

G

16x16

\[ \text{toRGB} \]

16x16

D

\[ \text{fromRGB} \]

16x16

(a)

\[ \text{fromRGB} \]

16x16

(b)

16x16

\[ 0.5x \]

16x16

\[ 1 - \alpha \]

\[ \alpha \]

(c)

\[ \text{fromRGB} \]

16x16

\[ 32x32 \]

\[ 0.5x \]
Progressive growing of GANs

When training the discriminator, real images are downscaled to match the current resolution of the network.
Progressive growing of GANs
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)
• Conditional GAN
• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)
• **High quality image generation**
  – Progressive GAN
  – **StackGAN**
  – StyleGAN and StyleGAN2 (will not be covered)
<table>
<thead>
<tr>
<th>Image</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Stage-I" /></td>
<td>This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face.</td>
</tr>
<tr>
<td><img src="image2" alt="Stage-II" /></td>
<td>This bird is white with some black on its head and wings, and has a long orange beak.</td>
</tr>
<tr>
<td><img src="image3" alt="Stage-II" /></td>
<td>This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments.</td>
</tr>
</tbody>
</table>
StackGAN

Stage-I Generator

Conditioning Augmentation

Stage-I Discriminator

This bird is grey with white on its chest and has a very short beak

Stage-II Generator

Conditioning Augmentation

Stage-II Discriminator

This bird is grey with white on its chest and has a very short beak
Today

• Measuring performance of GAN
  – Inception Score (IS)
  – Fréchet Inception Distance (FID)

• Conditional GAN

• GAN for cross-domain transfer
  – Pix2Pix networks: image-to-image translation
  – CycleGAN
  – StarGAN (will not be covered)

• High quality image generation
  – Progressive GAN
  – StackGAN
  – StyleGAN and StyleGAN2 (will not be covered)
Many more GANs

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:

  Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

• Other types of GANs:
  – InfoGAN, Energy-Based GAN
  – StarGAN and DiscoGAN
  – StyleGAN

• Cool applications:
  – Music generation
  – Video generation
  – Image inpainting

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting
  - Different pose generation

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting
  - Different pose generation

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

• Other types of GANs:
  – InfoGAN, Energy-Based GAN
  – StarGAN and DiscoGAN
  – StyleGAN

• Cool applications:
  – Music generation
  – Video generation
  – Image inpainting
  – Different pose generation

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

• Other types of GANs:
  – InfoGAN, Energy-Based GAN
  – StarGAN and DiscoGAN
  – StyleGAN

• Cool applications:
  – Music generation
  – Video generation
  – Image inpainting
  – Different pose generation
  – Super resolution

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool
  - Music
  - Video
  - Image
  - Different pose generation
  - Super resolution

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections

Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]
Many more GANs

• Other types of GANs:
  – InfoGAN, Energy-Based GAN
  – StarGAN and DiscoGAN
  – StyleGAN

• Cool applications:
  – Music generation
  – Video generation
  – Image inpainting
  – Different pose generation
  – Super resolution

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting
  - Different pose generation
  - Super resolution
  - Texture synthesis

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting
  - Different pose generation
  - Super resolution
  - Texture synthesis

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Many more GANs

- Other types of GANs:
  - InfoGAN, Energy-Based GAN
  - StarGAN and DiscoGAN
  - StyleGAN

- Cool applications:
  - Music generation
  - Video generation
  - Image inpainting
  - Different pose generation
  - Super resolution
  - Texture synthesis

Many implementations. For example: https://github.com/hwalsuklee/tensorflow-generative-model-collections
Questions?