Conditional GANs

Source
Outline

• Introduction
• Generation conditioned on class
  • Self-attention GAN
  • BigGAN
• Generation conditioned on image
  • Paired image-to-image translation: pix2pix
  • Unpaired image-to-image translation: CycleGAN
• Recent trends
Conditional generation

• Suppose we want to condition the generation of samples on discrete side information (label) $y$.
  • How do we add $y$ to the basic GAN framework?
Conditional generation

• Suppose we want to condition the generation of samples on discrete side information (label) $y$
  • How do we add $y$ to the basic GAN framework?

![Diagram of GAN with conditional generation]
Conditional generation

- Example: simple network for generating 28 x 28 MNIST digits


Figure source: F. Fleuret
Conditional generation

- Example: simple network for generating 28 x 28 MNIST digits


"The precise architecture of the discriminator is not critical as long as it has sufficient power; we have found that maxout units are typically well suited to the task."

Figure source: F. Fleuret
Conditional generation

- Example: simple network for generating 28 x 28 MNIST digits

Conditional generation

- Another example: text-to-image synthesis

S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, H. Lee, Generative adversarial text to image synthesis, ICML 2016
Conditional generation

- Another example: text-to-image synthesis

Outline

• Introduction

• Generation conditioned on class
  • Self-attention GAN
  • BigGAN
Self-attention GAN

- Adaptive receptive fields to capture non-local structure

Self-attention GAN

- Adaptive receptive fields to capture non-local structure (based on Wang et al., 2018)

$$s_{ij} = f(x_i)^T g(x_j)$$

$$\beta_{j,i} = \frac{\exp(s_{ij})}{\sum_i \exp(s_{ij})}$$

How much to attend to location $i$ while synthesizing feature at location $j$

$$o_j = v \left( \sum_i \beta_{j,i} h(x_i) \right)$$
Self-attention GAN: Implementation details

• Hinge loss formulation:

\[
L_D = -\mathbb{E}_{(x,y) \sim p_{data}} \left[ \min(0, D(x, y) - 1) \right] \\
- \mathbb{E}_{z \sim p_z, y \sim p_{data}} \left[ \min(0, -D(G(z, y), y) - 1) \right]
\]

\[
L_G = -\mathbb{E}_{z \sim p_z, y \sim p_{data}} D(G(z, y), y)
\]
Self-attention GAN: Implementation details

- Hinge loss formulation
- Conditioning the discriminator: *projection* ([Miyato & Koyama, 2018](https://arxiv.org/abs/1810.00961))
- Conditioning the generator: *conditional batch norm*
Self-attention GAN: Implementation details

- Hinge loss formulation
- Conditioning the discriminator: projection (Miyato & Koyama, 2018)
- Conditioning the generator: conditional batch norm
- Spectral normalization for generator and discriminator (Miyato et al., 2018) – divide weight matrices by largest singular value (estimated)
- Different learning rates for generator and discriminator (TTUR or two-timescale update rule – Heusel et al., 2017)
Self-attention GAN: Results

- 128 x 128 ImageNet

**goldfish**

**indigo bunting**

**redshank**

**Saint Bernard**
Self-attention GAN: Results

- Attention map visualization
BigGAN

• Scale up SA-GAN to generate ImageNet images up to 512 x 512 resolution

A. Brock, J. Donahue, K. Simonyan, Large scale GAN training for high fidelity natural image synthesis, ICLR 2019
BigGAN: Implementation details

• 8x larger batch size, 50% more channels (2x more parameters) than baseline SA-GAN

• Hierarchical latent space: feed (transformations of) $z$ vector into multiple layers of the generator
BigGAN: Implementation details

- 8x larger batch size, 50% more channels (2x more parameters) than baseline SA-GAN
- Hierarchical latent space: feed (transformations of) $z$ vector into multiple layers of the generator
- Truncation trick: at test time, resample the components of the $z$ vector whose magnitude falls above a certain threshold
- Trade off diversity for image quality

“The effects of increasing truncation. From left to right, the threshold is set to 2, 1, 0.5, 0.04.”
BigGAN: Implementation details

- 8x larger batch size, 50% more channels (2x more parameters) than baseline SA-GAN
- Hierarchical latent space: feed (transformations of) $z$ vector into multiple layers of the generator
- Truncation trick: at test time, resample the components of the $z$ vector whose magnitude falls above a certain threshold
- Lots of other tricks (initialization, training, etc.)
- Training observed to be unstable, but good results are achieved “just before collapse”
- Evidence that discriminator memorizes the training data, but the generator doesn’t
BigGAN: Implementation details

https://xkcd.com/1838/
BigGAN: Results

- Samples at 256 x 256 resolution:
BigGAN: Results

- Samples at 512 x 512 resolution:
BigGAN: Results

- Interpolation between $c$ with $z$ held constant:
BigGAN: Results

- Interpolation between $c, z$ pairs:
BigGAN: Results

- Difficult classes:
Conditional GANs: Outline

• Introduction
• Generation conditioned on class
  • Self-attention GAN
  • BigGAN
• Generation conditioned on image
  • Paired image-to-image translation: pix2pix
  • Unpaired image-to-image translation: CycleGAN
Image-to-image translation

Image-to-image translation

- Produce modified image $y$ conditioned on input image $x$ (note change of notation)
  - Generator receives $x$ as input
  - Discriminator receives an $x, y$ pair and has to decide whether it is real or fake
Image-to-image translation

- Generator architecture: U-Net

- Note: no $z$ used as input, transformation is basically deterministic
Image-to-image translation

- Generator architecture: U-Net

Encode: convolution $\rightarrow$ BatchNorm $\rightarrow$ ReLU

Decode: transposed convolution $\rightarrow$ BatchNorm $\rightarrow$ ReLU
Image-to-image translation

- Generator architecture: U-Net

Effect of adding skip connections to the generator
Image-to-image translation

- Generator loss: GAN loss plus L1 reconstruction penalty

\[ G^* = \arg \min_G \max_D \mathcal{L}_{GAN}(G, D) + \lambda \sum_i \| y_i - G(x_i) \|_1 \]

Generated output
\( G(x_i) \) should be close to ground truth target \( y_i \)
Image-to-image translation

- Generator loss: GAN loss plus L1 reconstruction penalty

\[ G^* = \arg \min_G \max_D \mathcal{L}_{GAN}(G, D) + \lambda \sum_i \|y_i - G(x_i)\|_1 \]
Image-to-image translation

• Discriminator
  • Given input image $x$ and second image $y$, decide whether $y$ is a ground truth target or produced by the generator
Image-to-image translation

- **Discriminator**
  - Given input image $x$ and second image $y$, decide whether $y$ is a ground truth target or produced by the generator.
  - “PatchGAN” architecture: output is a 30 x 30 map where each value (0 to 1) represents the quality of the corresponding section of the output image, these values are averaged to obtain final discriminator loss.
  - Implemented as FCN, effective patch size can be increased by increasing the depth.

![Diagram](image-source-url)
Image-to-image translation

- Discriminator
  - Given input image $x$ and second image $y$, decide whether $y$ is a ground truth target or produced by the generator
  - “PatchGAN” architecture: output is a $30 \times 30$ map where each value (0 to 1) represents the quality of the corresponding section of the output image, these values are averaged to obtain final discriminator loss
  - Implemented as FCN, effective patch size can be increased by increasing the depth

Effect of discriminator patch size on generator output
Image-to-image translation: Results

- Translating between maps and aerial photos
Image-to-image translation: Results

- Translating between maps and aerial photos
- Human study:

<table>
<thead>
<tr>
<th>Loss</th>
<th>Photo $\rightarrow$ Map % Turkers labeled real</th>
<th>Map $\rightarrow$ Photo % Turkers labeled real</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>2.8% ± 1.0%</td>
<td>0.8% ± 0.3%</td>
</tr>
<tr>
<td>L1+cGAN</td>
<td>6.1% ± 1.3%</td>
<td>18.9% ± 2.5%</td>
</tr>
</tbody>
</table>
Image-to-image translation: Results

- Semantic labels to scenes
Image-to-image translation: Results

- Semantic labels to scenes
- Evaluation: FCN score
  - The higher the quality of the output, the better the FCN should do at recovering the original semantic labels

<table>
<thead>
<tr>
<th>Loss</th>
<th>Per-pixel acc.</th>
<th>Per-class acc.</th>
<th>Class IOU</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>0.42</td>
<td>0.15</td>
<td>0.11</td>
</tr>
<tr>
<td>GAN</td>
<td>0.22</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cGAN</td>
<td>0.57</td>
<td>0.22</td>
<td>0.16</td>
</tr>
<tr>
<td>L1+GAN</td>
<td>0.64</td>
<td>0.20</td>
<td>0.15</td>
</tr>
<tr>
<td>L1+cGAN</td>
<td><strong>0.66</strong></td>
<td><strong>0.23</strong></td>
<td><strong>0.17</strong></td>
</tr>
<tr>
<td>Ground truth</td>
<td>0.80</td>
<td>0.26</td>
<td>0.21</td>
</tr>
</tbody>
</table>
Image-to-image translation: Results

- Scenes to semantic labels

<table>
<thead>
<tr>
<th>Input</th>
<th>Ground truth</th>
<th>L1</th>
<th>cGAN</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image9.png" alt="Image" /></td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image13.png" alt="Image" /></td>
<td><img src="image14.png" alt="Image" /></td>
<td><img src="image15.png" alt="Image" /></td>
<td><img src="image16.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Image-to-image translation: Results

• Scenes to semantic labels
• Accuracy is worse than that of regular FCNs or generator with L1 loss

<table>
<thead>
<tr>
<th>Loss</th>
<th>Per-pixel acc.</th>
<th>Per-class acc.</th>
<th>Class IOU</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>0.86</td>
<td>0.42</td>
<td>0.35</td>
</tr>
<tr>
<td>cGAN</td>
<td>0.74</td>
<td>0.28</td>
<td>0.22</td>
</tr>
<tr>
<td>L1+cGAN</td>
<td>0.83</td>
<td>0.36</td>
<td>0.29</td>
</tr>
</tbody>
</table>
Image-to-image translation: Results

- Semantic labels to facades
Image-to-image translation: Results

• Day to night
Image-to-image translation: Results

- Edges to photos
Image-to-image translation: Results

- pix2pix demo
Image-to-image translation: Limitations

- Visual quality could be improved
- Requires $x, y$ pairs for training
- Does not model conditional distribution $P(y|x)$, returns a single mode instead
Unpaired image-to-image translation

• Given two unordered image collections $X$ and $Y$, learn to “translate” an image from one into the other and vice versa.

Unpaired image-to-image translation

- Given two unordered image collections $X$ and $Y$, learn to “translate” an image from one into the other and vice versa.

CycleGAN

- Given: domains $X$ and $Y$
- Train two generators $F$ and $G$ and two discriminators $D_X$ and $D_Y$
  - $G$ translates from $X$ to $Y$, $F$ translates from $Y$ to $X$
  - $D_X$ recognizes images from $X$, $D_Y$ from $Y$
  - *Cycle consistency:* we want $F(G(x)) \approx x$ and $G(F(y)) \approx y$
CycleGAN: Architecture

- Generators (based on Johnson et al., 2016):

- Discriminators: PatchGAN on 70 x 70 patches
CycleGAN: Loss

• Requirements:
  • $G$ translates from $X$ to $Y$, $F$ translates from $Y$ to $X$
  • $D_X$ recognizes images from $X$, $D_Y$ from $Y$
  • We want $F(G(x)) \approx x$ and $G(F(y)) \approx y$

• CycleGAN discriminator loss: LSGAN

$$
\mathcal{L}_{GAN}(D_Y) = \mathbb{E}_{y \sim p_{data}(y)}[(D_Y(y) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)}[D_Y(G(x))^2]
$$

$$
\mathcal{L}_{GAN}(D_X) = \mathbb{E}_{x \sim p_{data}(x)}[(D_X(x) - 1)^2] + \mathbb{E}_{y \sim p_{data}(y)}[D_X(F(y))^2]
$$

• CycleGAN generator loss:

$$
\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[D_Y(G(x) - 1)^2] + \mathbb{E}_{y \sim p_{data}(y)}[D_X(F(y) - 1)^2]
$$

$$
+ \mathbb{E}_{x \sim p_{data}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{data}(y)}[\|G(F(y)) - y\|_1]
$$
CycleGAN

- Illustration of cycle consistency:
CycleGAN: Results

- Translation between maps and aerial photos
CycleGAN: Results

- Other pix2pix tasks
CycleGAN: Results

- Scene to labels and labels to scene
  - Worse performance than pix2pix due to lack of paired training data

<table>
<thead>
<tr>
<th>Loss</th>
<th>Per-pixel acc.</th>
<th>Per-class acc.</th>
<th>Class IOU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoGAN [32]</td>
<td>0.40</td>
<td>0.10</td>
<td>0.06</td>
</tr>
<tr>
<td>BiGAN/ALI [9, 7]</td>
<td>0.19</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>SimGAN [46]</td>
<td>0.20</td>
<td>0.10</td>
<td>0.04</td>
</tr>
<tr>
<td>Feature loss + GAN</td>
<td>0.06</td>
<td>0.04</td>
<td>0.01</td>
</tr>
<tr>
<td>CycleGAN (ours)</td>
<td><strong>0.52</strong></td>
<td><strong>0.17</strong></td>
<td><strong>0.11</strong></td>
</tr>
<tr>
<td>pix2pix [22]</td>
<td>0.71</td>
<td>0.25</td>
<td>0.18</td>
</tr>
</tbody>
</table>

Table 2: FCN-scores for different methods, evaluated on Cityscapes labels→photo.

<table>
<thead>
<tr>
<th>Loss</th>
<th>Per-pixel acc.</th>
<th>Per-class acc.</th>
<th>Class IOU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoGAN [32]</td>
<td>0.45</td>
<td>0.11</td>
<td>0.08</td>
</tr>
<tr>
<td>BiGAN/ALI [9, 7]</td>
<td>0.41</td>
<td>0.13</td>
<td>0.07</td>
</tr>
<tr>
<td>SimGAN [46]</td>
<td>0.47</td>
<td>0.11</td>
<td>0.07</td>
</tr>
<tr>
<td>Feature loss + GAN</td>
<td>0.50</td>
<td>0.10</td>
<td>0.06</td>
</tr>
<tr>
<td>CycleGAN (ours)</td>
<td><strong>0.58</strong></td>
<td><strong>0.22</strong></td>
<td><strong>0.16</strong></td>
</tr>
<tr>
<td>pix2pix [22]</td>
<td>0.85</td>
<td>0.40</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Table 3: Classification performance of photo→labels for different methods on cityscapes.
CycleGAN: Results

- Tasks for which paired data is unavailable
CycleGAN: Results

• Style transfer
CycleGAN: Failure cases

- apple → orange
- zebra → horse
- winter → summer
- dog → cat
- cat → dog
- Monet → photo
- photo → Ukiyo-e
- photo → Van Gogh
- iPhone photo → DSLR photo
CycleGAN: Failure cases

*Input* 

*Output*

horse $\rightarrow$ zebra
CycleGAN: Limitations

• Cannot handle shape changes (e.g., dog to cat)
• Can get confused on images outside of the training domains (e.g., horse with rider)
• Cannot close the gap with paired translation methods
• Does not account for the fact that one transformation direction may be more challenging than the other
Outline

• Introduction
• Generation conditioned on class
  • Self-attention GAN
  • BigGAN
• Generation conditioned on image
  • Paired image-to-image translation: pix2pix
  • Unpaired image-to-image translation: CycleGAN
• Some recent highlights
Multimodal image-to-image translation

Toward Multimodal Image-to-Image Translation, NIPS 2017
High-resolution, high-quality pix2pix

T.-C. Wang et al., *High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs*, CVPR 2018
High-resolution, high-quality pix2pix

- Two-scale generator architecture (up to 2048 x 1024 resolution)

First train *global generator* network (G1) on lower-res images

Then append higher-res *enhancer network* (G2) blocks and train G1 and G2 jointly

T.-C. Wang et al., *High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs*, CVPR 2018
High-resolution, high-quality pix2pix

- Two-scale generator architecture (up to 2048 x 1024 resolution)
- Three-scale discriminator architecture (full res, 2x and 4x downsampled)
- Incorporate feature matching loss into discriminator
Human generation conditioned on pose

Figure 3: (Top) Training: Our model uses a pose detector $P$ to create pose stick figures from video frames of the target subject. We learn the mapping $G$ alongside an adversarial discriminator $D$ which attempts to distinguish between the “real” correspondences $(x_t, x_{t+1}), (y_t, y_{t+1})$ and the “fake” sequence $(x_t, x_{t+1}), (G(x_t), G(x_{t+1}))$. (Bottom) Transfer: We use a pose detector $P$ to obtain pose joints for the source person that are transformed by our normalization process $Norm$ into joints for the target person for which pose stick figures are created. Then we apply the trained mapping $G$.

Human generation conditioned on pose

[Images of dance pose transitions]

https://carolineec.github.io/everybody_dance_now/

DeepFakes (coming up at the end of the course…)