Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures
4. Editing
5. Conditional GANs
Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures
4. Editing
5. Conditional GANs
   1. Principle
Motivation

Prior distribution $z$

c: train

Text

this white and yellow flower have thin white petals and a round yellow stamen

Image

Image

Image
Conditional GAN

- **Text to image** by traditional supervised learning

\[ c^1: \text{a dog is running} \quad \hat{x}^1: \]

\[ c^2: \text{a bird is flying} \quad \hat{x}^2: \]

\( x \):

\[ x \]:

\( c^2 \):

\[ \text{a bird is flying} \]

Text: “train”

Target of NN output

A blurry image!

\( \hat{x}^1: \text{as close as possible} \)
Conditional GAN

\[ x = G(c, z) \]

- \( c: \) train
- \( G \): generator
- \( z \): prior distribution
- \( \text{Image} \):

It is a distribution
Approximate the distribution of real data

Text: “train”
Conditional GAN

Prior distribution $z$  \[ \xrightarrow{} \] $G$  \[ \xrightarrow{} \] Image \[ x = G(c,z) \]

$x$ is realistic or not  \[ \xrightarrow{} \] $D$ (type 1) \[ \xrightarrow{} \] scalar

Positive example:  \[ \text{train} \]
Negative example:  \[ \text{Image} \]

$x$ is realistic or not + $c$ and $x$ are matched or not  \[ \xrightarrow{} \] $D$ (type 2) \[ \xrightarrow{} \] scalar

Positive example:  \[ (\text{train}, \text{train}) \]
Negative example:  \[ (\text{train}, \text{Image}) \] (\text{cat}, ...)
Conditional GAN (CGAN model)

\[
\min_G \max_D \left( \mathbb{E}_{x,y \sim p_{\text{data}}} [\log D(x, y)] + \mathbb{E}_{y \sim p_y, z \sim p_z} [\log(1 - D(G(z, y), y))] \right)
\]
Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures
4. Editing
5. Conditional GANs
   1. Principle
   2. Text2Image
Text2Image: architecture example

Positive samples:
- real image + right texts

Negative samples:
- fake image + right texts
- Real image + wrong texts

[Reed et al. ICML 2016]
Text2Image results

this small bird has a pink breast and crown, and black primaries and secondaries. this magnificent fellow is almost all black with a red crest, and white cheek patch.

the flower has petals that are bright pinkish purple with white stigma

this white and yellow flower have thin white petals and a round yellow stamen

[Reed et al. ICML 2016]
<table>
<thead>
<tr>
<th>Caption</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>this flower has white petals and a yellow stamen</td>
<td><img src="image1.jpg" alt="Image" /></td>
</tr>
<tr>
<td>the center is yellow surrounded by wavy dark purple petals</td>
<td><img src="image2.jpg" alt="Image" /></td>
</tr>
<tr>
<td>this flower has lots of small round pink petals</td>
<td><img src="image3.jpg" alt="Image" /></td>
</tr>
</tbody>
</table>
Text2Image: architecture example (2)

StackGAN: similar idea with LapGan to generate higher resolution images

Figure 2. The architecture of the proposed StackGAN. The Stage-I generator draws a low-resolution image by sketching rough shape and basic colors of the object from the given text and painting the background from a random noise vector. Conditioned on Stage-I results, the Stage-II generator corrects defects and adds compelling details into Stage-I results, yielding a more realistic high-resolution image. 

[Zhang et al. 2016]
StackGAN results

This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face

This bird is white with some black on its head and wings, and has a long orange beak

This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments

(a) Stage-I images

(b) Stage-II images

[Zhang et al. 2016]
<table>
<thead>
<tr>
<th>Caption</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>a pitcher is about to throw the ball to the batter</td>
<td>![Images of baseball players]</td>
</tr>
<tr>
<td>a group of people on skis stand in the snow</td>
<td>![Images of skiers in snow]</td>
</tr>
<tr>
<td>a man in a wet suit riding a surfboard on a wave</td>
<td>![Images of people surfing]</td>
</tr>
</tbody>
</table>
Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures
4. Editing
5. Conditional GANs
   1. Principle
   2. Text2Image
   3. Image2Image
Image-to-Image Translation \textit{pix2pix}

- Conditioned on an image of different modality
- No need to specify the loss function

\[ \text{Isola et al. CVPR 2017} \]
Image-to-image pix2pix

\[ x = G(z | c) \]
Image-to-image pix2pix

- Traditional supervised approach

Testing:

It is blurry because it is the average of several images.
Image-to-image

- Conditional GAN

Testing:

- Input
- Close
- GAN
- GAN + close
- GT

Supplementary loss (close)
Positive examples

Real or fake pair?

D

G tries to synthesize fake images that fool D

D tries to identify the fakes

Negative examples

Real or fake pair?

D

G
Edges2Image

[Isola et al. CVPR 2017]
Pix2pixHD [CVPR 2018]

High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs
Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, Bryan Catanzaro
Pix2pixHD [CVPR 2018]

**Coarse-to-fine Generator**

- Residual blocks

**Multi-scale Discriminators**

- $D_1$
- $D_2$
- $D_3$

**Robust Objective**

- $D_i$
- Real
- Match
- Real
- Synthesized
- Synthesized
Results

• Qualitative comparisons

Semantic Map

pix2pix

CRN

Ours
Improving Segmentation2Image strategy

[SPADE: Semantic Image Synthesis with Spatially-Adaptive Normalization, CVPR19]
Improving Segmentation2Image strategy

Previous approach:

Directly feed the semantic layout as input to the deep network, which is processed through stacks of convolution, normalization, and nonlinearity layers.

However, this is suboptimal as the normalization layers tend to “wash away” semantic information in input semantic segmentation masks.
Improving Segmentation2Image strategy

Proven effective for recent generative adversarial networks such as StyleGAN

Can we do the same for conditional GAN? **Conditional Normalization Layers?**
Improving Segmentation2Image strategy

Recall: Adaptive instance normalization

\[ \text{AdaIN}(x, y) = \sigma(y) \left( \frac{x - \mu(x)}{\sigma(x)} \right) + \mu(y) \]

SPADE block = spatially-adaptive denormalization:
Same idea but per class c over each channel i (N=batch size)

\[
\begin{align*}
\gamma_{c, y, x}^i = & \frac{1}{NH^i W^i} \sum_{n, y, x} h_{n, c, y, x}^i - \mu_{c}^i \\
\beta_{c, y, x}^i = & \frac{1}{NH^i W^i} \sum_{n, y, x} (h_{n, c, y, x}^i)^2 - (\mu_{c}^i)^2 \\
\end{align*}
\]

SPADE paper = [Semantic Image Synthesis with Spatially-Adaptive Normalization CVPR 2019]
SPADE Generator
SPADE Generator

Better preserve semantic information against common normalization layers
<table>
<thead>
<tr>
<th>Label</th>
<th>Ground Truth</th>
<th>Multi-modal results</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Label Image" /></td>
<td><img src="image2" alt="Ground Truth Image" /></td>
<td><img src="image3" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image4" alt="Label Image" /></td>
<td><img src="image5" alt="Ground Truth Image" /></td>
<td><img src="image6" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image7" alt="Label Image" /></td>
<td><img src="image8" alt="Ground Truth Image" /></td>
<td><img src="image9" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image10" alt="Label Image" /></td>
<td><img src="image11" alt="Ground Truth Image" /></td>
<td><img src="image12" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image13" alt="Label Image" /></td>
<td><img src="image14" alt="Ground Truth Image" /></td>
<td><img src="image15" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image16" alt="Label Image" /></td>
<td><img src="image17" alt="Ground Truth Image" /></td>
<td><img src="image18" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image19" alt="Label Image" /></td>
<td><img src="image20" alt="Ground Truth Image" /></td>
<td><img src="image21" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image22" alt="Label Image" /></td>
<td><img src="image23" alt="Ground Truth Image" /></td>
<td><img src="image24" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image25" alt="Label Image" /></td>
<td><img src="image26" alt="Ground Truth Image" /></td>
<td><img src="image27" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image28" alt="Label Image" /></td>
<td><img src="image29" alt="Ground Truth Image" /></td>
<td><img src="image30" alt="Multi-modal Results Image" /></td>
</tr>
<tr>
<td><img src="image31" alt="Label Image" /></td>
<td><img src="image32" alt="Ground Truth Image" /></td>
<td><img src="image33" alt="Multi-modal Results Image" /></td>
</tr>
</tbody>
</table>

**SPADE results**
SPADE with real image:
[OASIS iclr 2021] (follow-up paper of SPADE) with real image:

```
3D Noise & Label Map
Generator

Real Image
Fake Image

Segmentation-Based Discriminator

Real Image
Label Map
Real Image

Encoder
Noise
Generator
Fake Image

N Real/1 Fake Class
```
[OASIS iclr 2021] (follow-up paper of SPADE) with real image:
Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures
4. Editing
5. Conditional GANs
   1. Principle
   2. Text2Image
   3. Image2Image
   4. Inpainting and general missing data encoder
Inpainting task

- Complete the missing part
Inpainting as unsupervised learning with GAN loss

Reconstruct missing pixels by decoding using context

Loss defined on the predicted patch and the real one (known at training time)
First proposition -- Architecture

- Architecture: Encoder/Fully connected/Decoder

- DC-GAN for inpainting task
- **Input:** $227 \times 227 \times 3$ image
- **Output:** encoder context features $(6 \times 6 \times 256)$
Channel-wise fully-connected layer

- **Input / output:** $6 \times 6 \times 256$ channels
- **First layer:** Channel-wise fully-connected
  (each $6 \times 6$ input connected to the corresponding $6 \times 6$ output)
- **Second layer:** Stride 1 convolution to mix channels

Decoder

- **Architecture:** Same as DC-GAN: 5 up-convolutional layers
  ("deconv" + ReLU)
- **Input:** decoder context features $6 \times 6 \times 256$
- **Output:** $227 \times 227 \times 3$ image
Training: Masking the images

- **How to define the mask?**
  - Center region of the image
  - Random regions (chosen solution)
  - Random segmentation mask from VOC (said to be equivalent to random regions)

- **Formal definition:** Defined by a mask $\hat{M} \in \{0, 1\}^{227 \times 227}$ with 1 if the pixel should be masked
Training: Loss - Overview

- Trained completely from scratch to fill-up the masked areas
- **Problem**: multiple plausible solutions
- **Solution**: combining 2 losses:
  - $\mathcal{L}_{\text{rec}}$ **L2 reconstruction loss**: learn the structure of the missing region (average multiple modes in prediction)
  - $\mathcal{L}_{\text{adv}}$ **Adversarial loss**: make it look real (pick a mode from the distribution)

$$
\min_F \mathcal{L} = \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}}
$$

$$
\mathcal{L}_{\text{rec}}(x) = \left\| \hat{M} \odot \left( x - F((1 - \hat{M}) \odot x) \right) \right\|_2
$$

$$
\mathcal{L}_{\text{adv}} = \max_D \mathbb{E}_{x \in \mathcal{X}} \left[ \log(D(x)) + \log \left(1 - D(F((1 - \hat{M}) \odot x))\right) \right]
$$

- Rq: The encoder-decoder is the generator, D is a CNN
Results

Dataset: StreetView Paris and ImageNet
Semantic inpainting - Qualitative results
Generalizing inpainting: missing data encoder
Adding perceptual loss, BB regression loss

\[ r^c = (x^c/W, y^c/H, (x^c + w^c)/W, (y^c + h^c)/H) \]

\[ \mathcal{L}_{\text{disc}}^{HnS}(\theta_d) = \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} \| r^c_i - \hat{r}^c_i(\theta_g, \theta_d) \| \]

\[ \mathcal{L}_{\text{gen}}^{HnS}(\theta_g) = \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} \| q^c_i - \hat{q}^c_i(\theta_g, \theta_d) \| \]

\[ \mathcal{L}_{\text{tot}}(\theta_g, \theta_d) = \mathcal{L}_{\text{rec}}(\theta_g) + \lambda_{\text{compl}} \mathcal{L}_{\text{compl}}^{vvg}(\theta_g) + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}}(\theta_g, \theta_d) + \lambda_{HnS} \mathcal{L}_{\text{coord}}^{HnS}(\theta_g, \theta_d) \]
Results
Results
Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures
4. Editing
5. Conditional GANs
   1. Principle
   2. Text2Image
   3. Image2Image
   4. Inpainting and general missing data encoder
5. Learning unpaired Transformation
Unpaired Transformation
- Cycle GAN, Disco GAN

Transform an object from one domain to another without paired data

Domain X

Domain Y
Cycle GAN

https://arxiv.org/abs/1703.10593
https://junyanz.github.io/CycleGAN/

Domain X

Ignore input

\[ G_{X \rightarrow Y} \]

Become similar to domain Y

Not what we want

\[ D_Y \rightarrow \text{scalar} \]

Input image belongs to domain Y or not

Domain Y
Cycle GAN

as close as possible

Lack of information for reconstruction

Input image belongs to domain Y or not
Cycle GAN

\[ G_{X\rightarrow Y} \rightarrow G_{Y\rightarrow X} \]

as close as possible

\[ D_X \rightarrow D_Y \]

scalar: belongs to domain X or not

\[ G_{Y\rightarrow X} \rightarrow G_{X\rightarrow Y} \]

as close as possible

scalar: belongs to domain Y or not
Results -- Cycle GAN

photo -> van Gogh

Domain X

Domain Y

Monet ↔ Photos

Zebras ↔ Horses

Summer ↔ Winter
GANs: works in progress

A lot of things to better understand, to use, adapt, ...
Appendix

GANs for Video, 3D, etc.
Video GAN


[Vondrick et al. NIPS 2016]
Shape modeling using 3D Generative Adversarial Network

Wu et al. NIPS 2016