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

Generator

Image

Generator

Image

Generator

Generator

Generator
Conditional GAN

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

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

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

Text: “train”

Target of NN output

A blurry image!
Conditional GAN

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

- **c**: train
- Prior distribution \( z \)

It is a distribution
Approximate the distribution of real data

Text: “train”
Conditional GAN

Prior distribution $z$  $\rightarrow$ $\rightarrow$ $\rightarrow$ Image  $x = G(c,z)$

$c$: train $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

$x$ is realistic or not $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

$x$ is realistic or not + $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$
c and $x$ are matched or not

$D$ (type 1) $\rightarrow$ scalar $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

Positive example: $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

Negative example: $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

$D$ (type 2) $\rightarrow$ scalar $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

Positive example: $(\text{train, } \rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$

Negative example: $(\text{train, } \rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ (cat, $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$ $\rightarrow$)
Conditional GAN (CGAN model)

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

Positive samples:
- real image + right texts

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

[Reed et al. ICML 2016]
this small bird has a pink breast and crown, and black primaries and secondaries.

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

this magnificent fellow is almost all black with a red crest, and white cheek patch.

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.png" alt="Images of white flowers with a yellow stamen" /></td>
</tr>
<tr>
<td>the center is yellow surrounded by wavy dark purple petals</td>
<td><img src="image2.png" alt="Images of purple flowers with wavy petals" /></td>
</tr>
<tr>
<td>this flower has lots of small round pink petals</td>
<td><img src="image3.png" alt="Images of pink flowers with small petals" /></td>
</tr>
</tbody>
</table>
StackGAN: similar idea with LapGan

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

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
<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><img src="image1.png" alt="Image of a pitcher throwing a ball" /> <img src="image2.png" alt="Image of a pitcher throwing a ball" /> <img src="image3.png" alt="Image of a pitcher throwing a ball" /> <img src="image4.png" alt="Image of a pitcher throwing a ball" /> <img src="image5.png" alt="Image of a pitcher throwing a ball" /></td>
</tr>
<tr>
<td>a group of people on skis stand in the snow</td>
<td><img src="image6.png" alt="Image of people on skis" /> <img src="image7.png" alt="Image of people on skis" /> <img src="image8.png" alt="Image of people on skis" /> <img src="image9.png" alt="Image of people on skis" /> <img src="image10.png" alt="Image of people on skis" /></td>
</tr>
<tr>
<td>a man in a wet suit riding a surfboard on a wave</td>
<td><img src="image11.png" alt="Image of a man surfing" /> <img src="image12.png" alt="Image of a man surfing" /> <img src="image13.png" alt="Image of a man surfing" /> <img src="image14.png" alt="Image of a man surfing" /> <img src="image15.png" alt="Image of a man 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

[Isola et al. CVPR 2017]
Image-to-image pix2pix

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

[Isola et al. CVPR 2017]
Label2Image

Input

Ground truth

L1

cGAN

L1 + cGAN

[Isola et al. CVPR 2017]
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

Multi-scale Discriminators

Robust Objective

Residual blocks

Residual blocks

real
synthesized

real

real

synthesized
Results

• Qualitative comparisons
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

Conditional Normalization Layers

Proven effective for recent generative adversarial networks such as StyleGAN
Improving Segmentation2Image strategy

SPADE block: spatially-adaptive denormalization

\[ \gamma_{c,y,x}(m) \frac{h_{n,c,y,x} - \mu_{c}^{i}}{\sigma_{c}^{i}} + \beta_{c,y,x}(m) \]

\[ \mu_{c}^{i} = \frac{1}{NH^{i}W^{i}} \sum_{n,y,x} h_{n,c,y,x}^{i} \]

\[ \sigma_{c}^{i} = \sqrt{\frac{1}{NH^{i}W^{i}} \sum_{n,y,x} (h_{n,c,y,x}^{i})^{2} - (\mu_{c}^{i})^{2}} \]
SPADE Generator
SPADE Generator

Better preserve semantic information against common normalization layers
SPADE 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
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 × 227 × 3 image
- **Output:** encoder context features (6 × 6 × 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

![Examples of masks applied to images](image)
Training: Loss - Overview

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

\[
\min_F \mathcal{L} = \lambda_{rec} \mathcal{L}_{rec} + \lambda_{adv} \mathcal{L}_{adv}
\]

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

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

- **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

\[ \mathcal{L}_{rec} = (x^c/W, y^c/H, (x^c + w^c)/W, (y^c + h^c)/H) \]

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

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

\[ \mathcal{L}_{tot}(\theta_g, \theta_d) = \mathcal{L}_{rec}(\theta_g) + \lambda_{compl} \mathcal{L}_{compl}^{vgg}(\theta_g) + \lambda_{adv} \mathcal{L}_{adv}(\theta_g, \theta_d) + \lambda_{H_nS} \mathcal{L}_{coord}^{H_nS}(\theta_g, \theta_d) \]
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 \textit{without paired data}
Cycle GAN

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

Domain X

become similar to domain Y

Not what we want

ignore input

$G_{X \rightarrow Y}$

Input image belongs to domain Y or not

$D_Y$ scalar
Cycle GAN

Domain X

as close as possible

Domain Y

$G_{X \rightarrow Y}$

$L_{Y \rightarrow X}$

$L_{Y \rightarrow X}$

Lack of information for reconstruction

Input image belongs to domain Y or not

scalar
Cycle GAN

![Image of Cycle GAN diagram]

- $G_{X \rightarrow Y}$: Generator mapping from domain X to domain Y
- $G_{Y \rightarrow X}$: Generator mapping from domain Y to domain X
- $D_X$: Discriminator determining whether a scalar belongs to domain X or not
- $D_Y$: Discriminator determining whether a scalar belongs to domain Y or not

The goal is to make the mappings as close as possible:

- $G_{X \rightarrow Y}$: $D_Y(G_{X \rightarrow Y})$ as close as possible
- $G_{Y \rightarrow X}$: $D_X(G_{Y \rightarrow X})$ as close as possible

This cycle constraint ensures that the mappings do not distort the original domain characteristics.

![Image of Cycle GAN diagram]
Results -- Cycle GAN

photo \rightarrow \text{van Gogh}

Domain X

Domain Y
Appendix
GANs for Video, 3D, etc.
Video GAN


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

\[\text{Wu et al. NIPS 2016}\]