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

- **c:** train
- **Prior distribution** \( z \)
- **Text:**
  - this white and yellow flower have thin white petals and a round yellow stamen
- **Image:**
  - Train
  - Flower
  - Handbag
Conditional GAN

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

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

  \[ \hat{x}^1: \]

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

  \[ \hat{x}^2: \]

Text: “train”

Target of NN output

A blurry image!
Conditional GAN

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

It is a distribution
Approximate the distribution of real data

Text: “train”
Conditional GAN

Prior distribution $z$

$x = G(c, z)$

$x$ is realistic or not

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

Positive example: (train, )

Negative example: (train, ) (cat, )
Conditional GAN (CGAN model)

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

• Positive samples:
  • real image + right texts

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

This flower has small, round violet petals with a dark purple center

$\varphi \leftarrow (\varphi(t))$ $\Rightarrow \hat{x} := G(z, \varphi(t))$

This flower has small, round violet petals with a dark purple center

$D(\hat{x}, \varphi(t))$
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="White Flower Images" /></td>
</tr>
<tr>
<td>the center is yellow surrounded by wavy dark purple petals</td>
<td><img src="image2.png" alt="Purple Flower Images" /></td>
</tr>
<tr>
<td>this flower has lots of small round pink petals</td>
<td><img src="image3.png" alt="Pink Flower Images" /></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

(a) Stage-I images

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

(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></td>
</tr>
<tr>
<td>a group of people on skis stand in the snow</td>
<td></td>
</tr>
<tr>
<td>a man in a wet suit riding a surfboard on a wave</td>
<td></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 pix2pix

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

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

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

https://arxiv.org/pdf/1611.07004
Image-to-image pix2pix

- Traditional supervised approach

Testing:

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

• Conditional GAN

Supplementary loss (close)

Testing:

input  close  GAN  GAN + close  GT
Positive examples

Real or fake pair?

\[ D \]

\[ \text{G tries to synthesize fake images that fool D} \]

\[ \text{D tries to identify the fakes} \]

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
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

\[
\gamma_{c,y,x}(m) \frac{h_{n,c,y,x} - \mu_c}{\sigma_c} + \beta_{c,y,x}(m)
\]

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

\[
\sigma_c = \sqrt{\frac{1}{NH^i W^i} \sum_{n,y,x} (h_{n,c,y,x})^2 - (\mu_c)^2}
\]
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:** $227 \times 227 \times 3$ 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)

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

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

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

- **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

\[ r^c = \left( \frac{x^c}{W}, \frac{y^c}{H}, \frac{x^c + w^c}{W}, \frac{y^c + h^c}{H} \right) \]

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

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

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

$D_Y$ → scalar

Input image belongs to domain Y or not

Domain Y
Cycle GAN

\[ G_{X \rightarrow Y} \]

as close as possible

\[ G_{Y \rightarrow X} \]

Lack of information for reconstruction

\[ D_Y \]

Input image belongs to domain Y or not

scalar

Domain X

Domain Y

Input image belongs to domain Y or not
Cycle GAN

**Domain X**

![Image of Domain X]

**Domain Y**

![Image of Domain Y]

---

as close as possible

---

\[ G_{X \rightarrow Y} \]

\[ D_X \]

scalar: belongs to domain X or not

\[ G_{Y \rightarrow X} \]

\[ D_Y \]

scalar: belongs to domain Y or not

---

as close as possible

---
Results -- Cycle GAN

photo → van Gogh

Domain X

Domain Y

Monet ↔ Photos

Zebras ↔ Horses

Summer ↔ Winter

Monet → photo

horse → zebra

photo → Monet

summer → winter

winter → summer
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]