Deep Learning for Computer Vision

Deep Generative Models: Image Applications

Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Homework

Question: Why is MI Gap and not MI used as a metric for disentanglement?

Homework

Question: Why is MI Gap and not MI used as a metric for disentanglement?

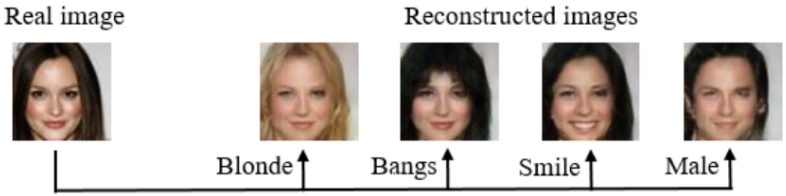
- MI Gap penalizes unaligned latent variables (contains information about more than one generative factor) $\rightarrow$ undesirable for disentanglement
- If one latent variable reliably models a generative factor, not required for other latent variables to be informative about this factor
- Read Chen et al, Isolating Sources of Disentanglement in Variational Autoencoders, NeurIPS 2018 for more information!

GANs for Image Editing

- Simple image edits such as grayscaling, brightness adjustment or contrast don't require an understanding of the image

GANs for Image Editing

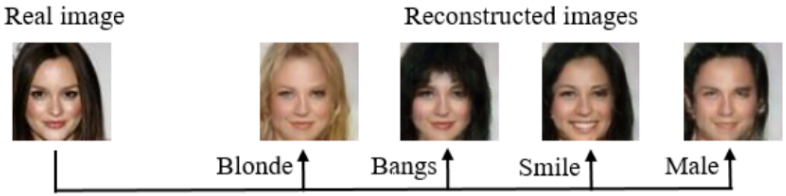
- Simple image edits such as grayscaling, brightness adjustment or contrast don't require an understanding of the image



- However, challenging operations such as changing attributes of a face require sound understanding of the input image. Can GANs help?

GANs for Image Editing

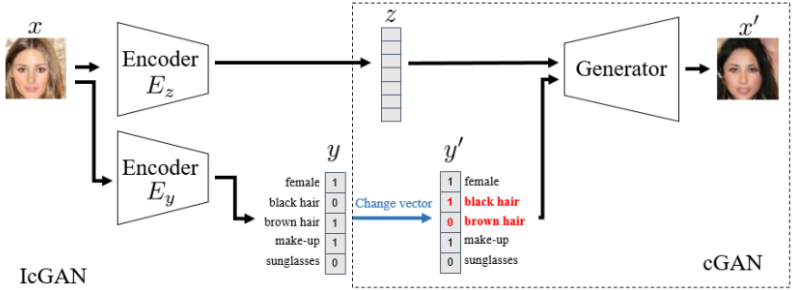
- Simple image edits such as grayscaling, brightness adjustment or contrast don't require an understanding of the image



- However, challenging operations such as changing attributes of a face require sound understanding of the input image. Can GANs help?
- But a GAN on its own does not have the mechanism to map a real image to its latent representation

Image Editing with Invertible Conditional GANs (IcGAN)[1]

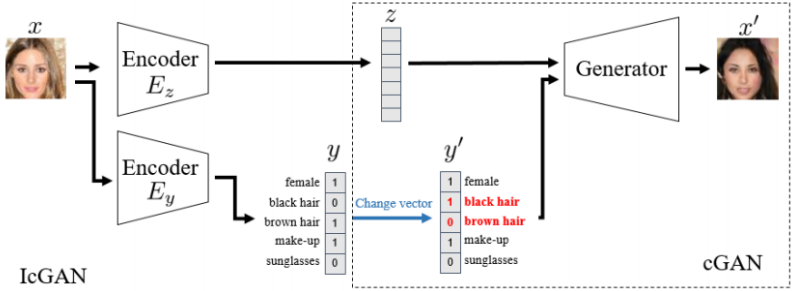
- Combines a conditional GAN with an encoder which can encode an image to its latent representation



[1] Perarnau et al, Invertible Conditional GANs for Image Editing, NeurIPS-W 2016

Image Editing with Invertible Conditional GANs (IcGAN)[1]

- Combines a conditional GAN with an encoder which can encode an image to its latent representation

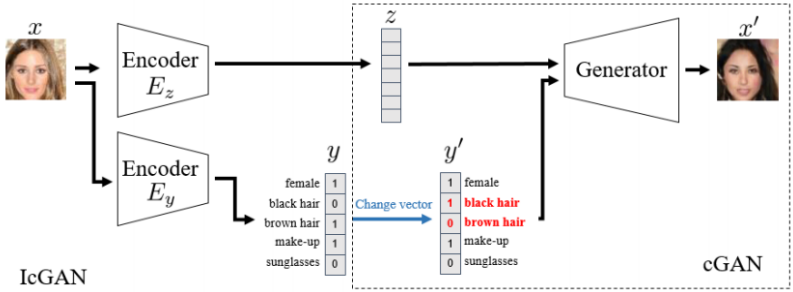


- Can be used for face editing as follows: (1) Pass image through encoder; (2) Change attribute vector y; (3) Pass through generator

[1] Perarnau et al, Invertible Conditional GANs for Image Editing, NeurIPS-W 2016

Invertible Conditional GANs

- Generator G samples an image x' from a latent representation z and conditional information y i.e., $G(z, y) = x'$



- Encoder performs the inverse i.e., $E(x) = (z, y)$

Training IcGAN

- First, a conditional GAN is trained to optimize the following objective:

$$\min_g \max_d v(\theta_g, \theta_d) = E_{x,y\sim p_{data}}[\log D(x,y)] + E_{z\sim p_z, x\sim p_x}[\log(1 - D(G(z,y'),y'))]$$

Training IcGAN

- First, a conditional GAN is trained to optimize the following objective:

$$\min_g \max_d v(\theta_g, \theta_d) = E_{x,y\sim p_{data}}[\log D(x,y)] + E_{z\sim p_z, x\sim p_x}[\log(1 - D(G(z,y'),y'))]$$

- Encoder has two parts: E_z which maps an image to its latent representation, and E_y which maps an image to its conditional information (attributes)

Training IcGAN

- First, a conditional GAN is trained to optimize the following objective:

$$\min_g \max_d v(\theta_g, \theta_d) = E_{x,y\sim p_{data}}[\log D(x,y)] + E_{z\sim p_z.x\sim p_x}[\log(1 - D(G(z,y'),y'))]$$

- Encoder has two parts: E_z which maps an image to its latent representation, and E_y which maps an image to its conditional information (attributes)
- To train E_z, generator is used to generate a dataset of images. Pairs of images and their latent representations (x', z) are used to train E_z using reconstruction loss:

$$\mathcal{L}_{ez} = E_{z\sim p_z.y'\sim p_y}\|z - E_z(G(z,y'))\|_2^2$$

Training IcGAN

- First, a conditional GAN is trained to optimize the following objective:

$$\min_g \max_d v(\theta_g, \theta_d) = E_{x,y\sim p_{data}}[\log D(x,y)] + E_{z\sim p_z.x\sim p_x}[\log(1 - D(G(z,y'),y'))]$$

- Encoder has two parts: E_z which maps an image to its latent representation, and E_y which maps an image to its conditional information (attributes)
- To train E_z, generator is used to generate a dataset of images. Pairs of images and their latent representations (x', z) are used to train E_z using reconstruction loss:

$$\mathcal{L}_{ez} = E_{z\sim p_z.y'\sim p_y}\|z - E_z(G(z,y'))\|_2^2$$

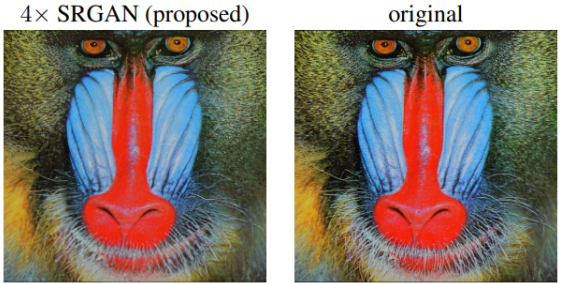
- E_y is trained using real image and attribute pairs:

$$\mathcal{L}_{ey} = E_{x,y\sim p_{data}}\|y - E_y(x)\|_2^2$$

IcGAN: Results on CelebA dataset

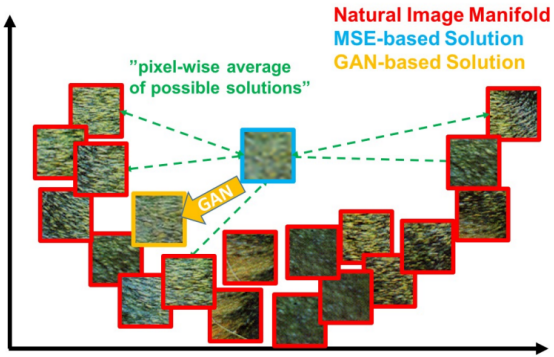
GANs for Super-Resolution[2]

- **Image Super-Resolution:** Task of obtaining a High-Resolution (HR) image from a Low Resolution (LR) image, a challenging task
- **SRGAN (Super-Resolution GAN)** aims to generate photo-realistic images with 4× upscaling factors



[2]Ledig et al, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, CVPR 2017

SR-GAN



- **Super-Resolution CNNs** minimize Mean Square Error (MSE) $\rightarrow$ tend to overly smoothen output images, due to pixel-wise averaging of outcomes
- **SR-GAN**, because of its adversarial objective, drives outputs to natural image manifold $\rightarrow$ results in high quality super-resolved images

SR-GAN: Content and Adversarial Losses

- Uses a pretrained VGG-19 network to compute **content loss** as MSE between VGG-19 feature representations of generator ($G_{\theta_G}(I^{LR})$) and true image I^{HR}:

$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

where $\phi_{i,j}$ = feature map obtained by j-th convolution before i-th max pooling layer, $W_{i,j}$ and $H_{i,j}$ are width and height of feature map respectively

SR-GAN: Content and Adversarial Losses

- Uses a pretrained VGG-19 network to compute **content loss** as MSE between VGG-19 feature representations of generator $(G_{\theta_G}(I^{LR}))$ and true image I^{HR}:

$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

  where $\phi_{i,j}$ = feature map obtained by j-th convolution before i-th max pooling layer, $W_{i,j}$ and $H_{i,j}$ are width and height of feature map respectively
- **Adversarial loss** given by:

$$l^{SR}_{Gen} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

SR-GAN: Content and Adversarial Losses

- Uses a pretrained VGG-19 network to compute **content loss** as MSE between VGG-19 feature representations of generator $(G_{\theta_G}(I^{LR}))$ and true image I^{HR}:

$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

  where $\phi_{i,j}$ = feature map obtained by j-th convolution before i-th max pooling layer, $W_{i,j}$ and $H_{i,j}$ are width and height of feature map respectively

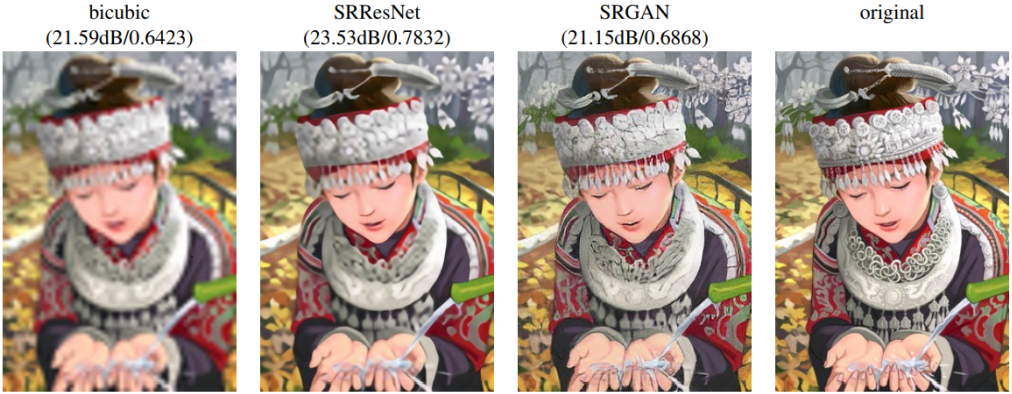
- **Adversarial loss** given by:

$$l^{SR}_{Gen} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

- **Perceptual loss**, a weighted sum of content and adversarial loss:

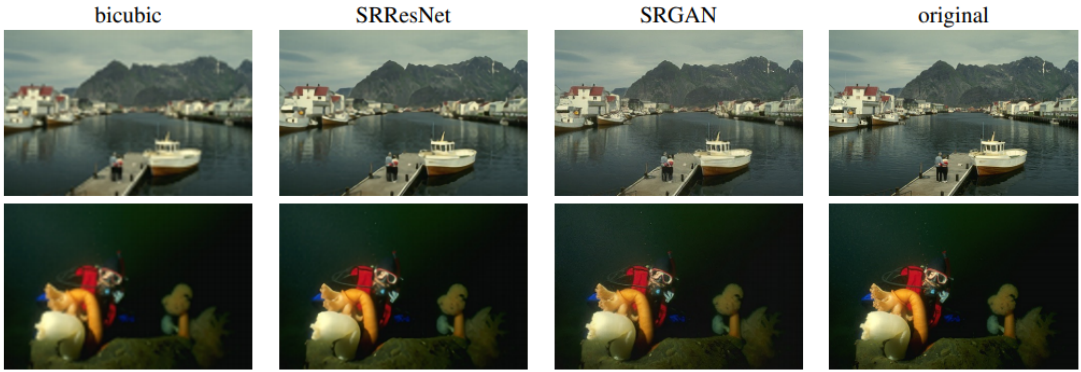
$$l^{SR} = l^{SR}_{\text{X}} + 10^{-3} l^{SR}_{Gen}$$

SR-GAN Results[3]



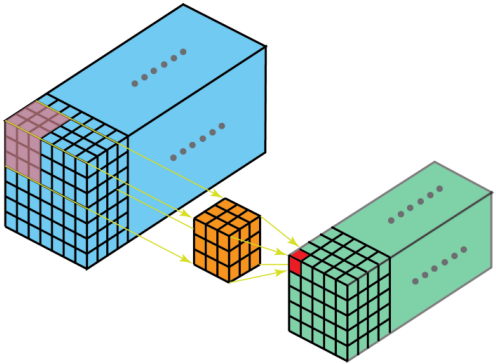
[3]Ledig et al, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, CVPR 2017

SR-GAN More Results[4]



[4]Ledig et al, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, CVPR 2017

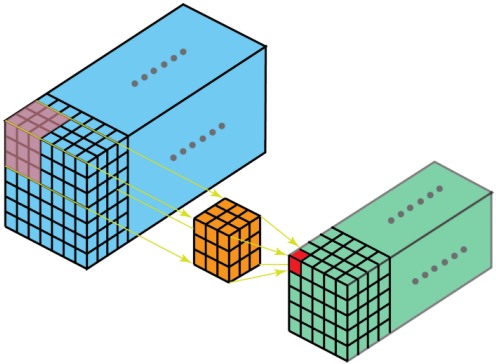
GANs for 3D Object Generation[5]



- We have seen GANs generate photo-realistic 2D images; can GANs generate 3D objects too?

[5] Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS 2016

GANs for 3D Object Generation[5]

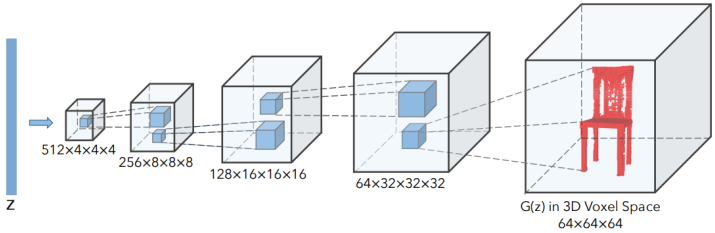


- We have seen GANs generate photo-realistic 2D images; can GANs generate 3D objects too?
- Yes! Recent advances in volumetric (3D) convolutional networks have made it possible to learn 3D object representations

[5] Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS 2016

GANs for 3D Object Generation[6]

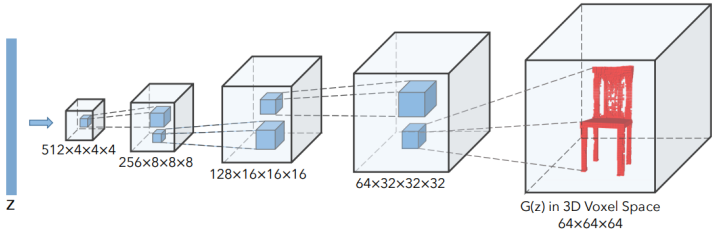
- Generator G maps a 200-length latent vector z to a $64 \times 64 \times 64$ cube, representing an object $G(z)$ in 3D-voxel space



[6]Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS 2016

GANs for 3D Object Generation[6]

- Generator G maps a 200-length latent vector z to a $64 \times 64 \times 64$ cube, representing an object $G(z)$ in 3D-voxel space

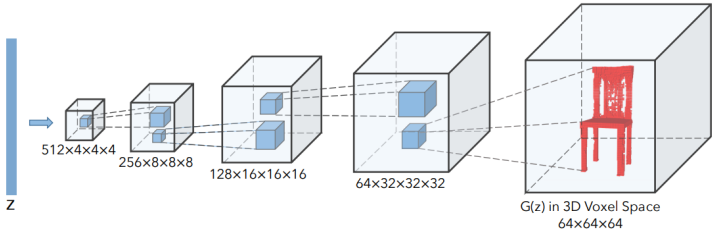


- Discriminator D classifies whether an input 3D object x is real or synthetic

[6]Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS 2016

GANs for 3D Object Generation[6]

- Generator G maps a 200-length latent vector z to a $64 \times 64 \times 64$ cube, representing an object $G(z)$ in 3D-voxel space

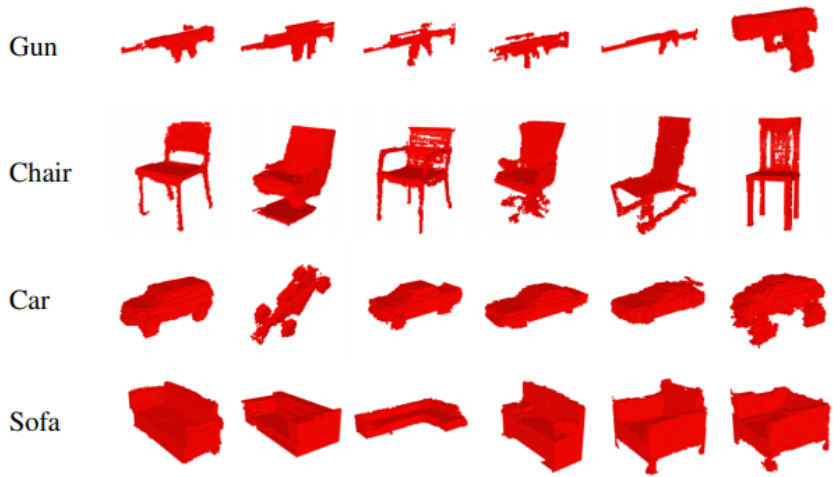


- Discriminator D classifies whether an input 3D object x is real or synthetic
- Adversarial loss in 3D-GAN:

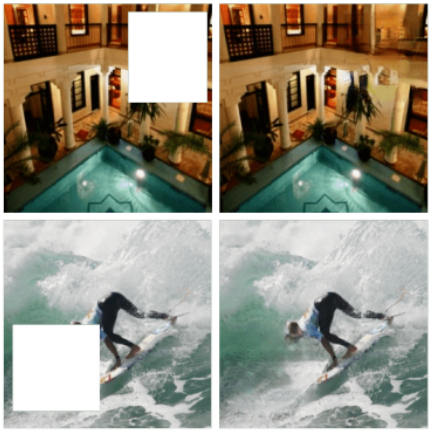
$$L_{3DGAN} = \log D(x) + \log(1 - D(G(z)))$$

[6]Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS 2016

3D-GAN Generated Objects



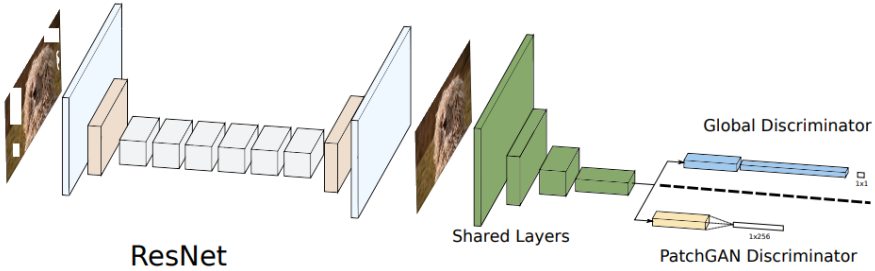
GANs for Image Inpainting[7]



- **Image inpainting** a reconstruction technique used for filling missing parts in an image
- Can a GAN be trained to perform image inpainting?

[7] Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

PG-GAN[8]



- Consists of a ResNet-based generator and a novel discriminator with two heads:
  - **G-GAN discriminator:** reasons globally at image level (decide real vs fake)
  - **Patch GAN discriminator:** reasons locally at patch level (decide real vs fake) - helps capture local texture details

[8] Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

PG-GAN Objective

- **Reconstruction loss:** pixel-wise L1 distance between generated image and ground truth:

$$\mathcal{L}_{rec} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{WHC}\|y - x\|_1$$

- **Adversarial loss:** standard GAN objective:

$$\mathcal{L}_{GAN}(G, D) = E_{x\sim p(x)}[\log D(x)] + E_{y\sim p_G(x')}[\log(1 - D(G(x')))]$$

where $\mathcal{L}_{g\_adv}$ = adversarial loss for global GAN; $\mathcal{L}_{p\_adv}$ = adversarial loss for Patch GAN

- Overall objective:

$$\mathcal{L} = \lambda_1\mathcal{L}_{rec} + \lambda_2\mathcal{L}_{g\_adv} + \lambda_3\mathcal{L}_{p\_adv}$$

PG-GAN Results[9]

[9] Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

PG-GAN More Results[10]

[10] Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

Homework

Readings

- 3D-GANs
- Avinash H, GAN Zoo
- (Optional) Respective papers

References

Guim Perarnau et al. "Invertible Conditional GANs for image editing". In: *ArXiv* abs/1611.06355 (2016).

Jiajun Wu et al. "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling". In: *Advances in Neural Information Processing Systems*. 2016, pp. 82–90.

C. Ledig et al. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 105–114.

Ugur Demir and Gözde B. Ünal. "Patch-Based Image Inpainting with Generative Adversarial Networks". In: *CoRR* abs/1803.07422 (2018). arXiv: 1803.07422.