DCGAN

Generate images with Deep Convolutional GAN

Dương Mạnh Cường
Phạm Nguyễn Mỹ Diễm

Reference:

Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks

https://arxiv.org/abs/1511.06434

Least Squares Generative Adversarial Networks

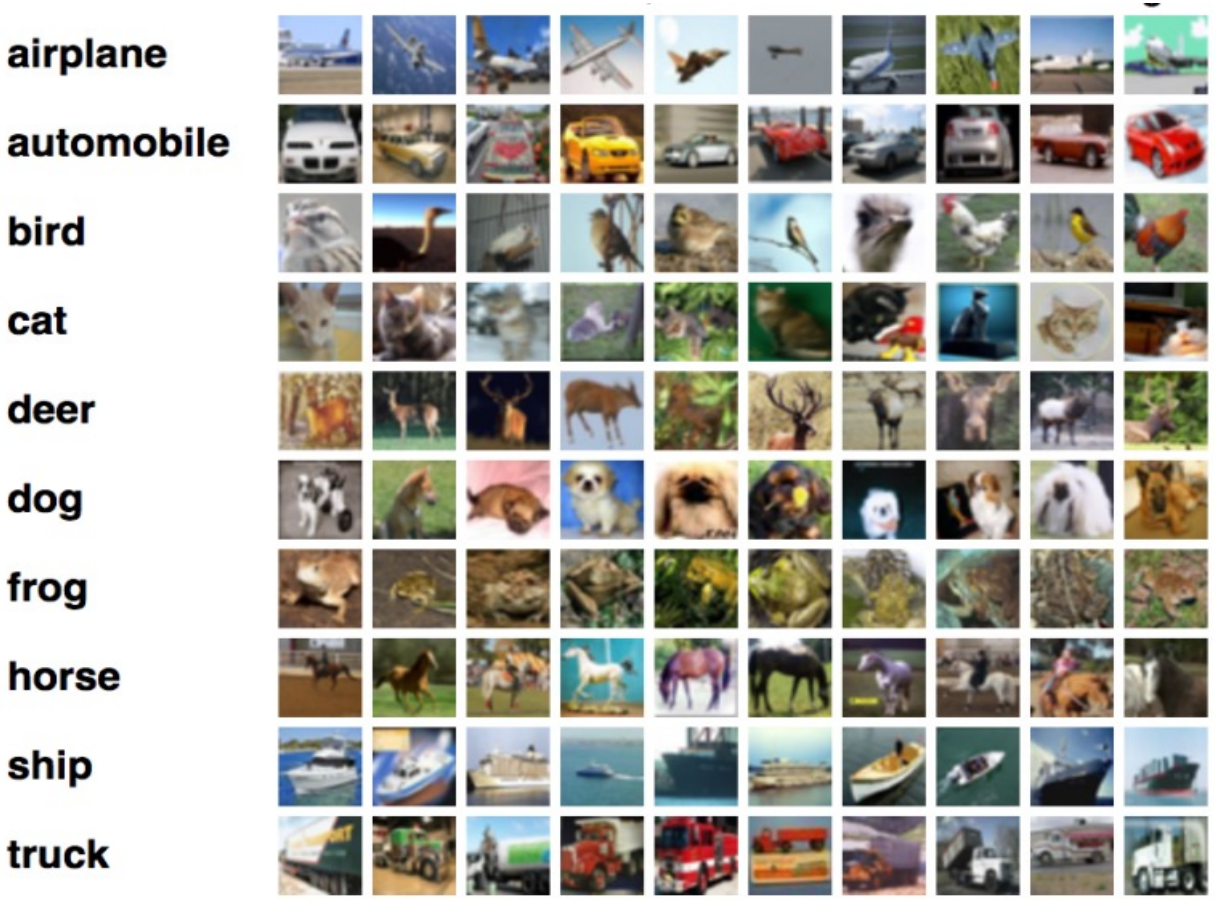
https://arxiv.org/abs/1611.04076

Nội dung

- Giới thiệu bài toán
- Bài báo liên quan
- Mô hình đề xuất
- Thực nghiệm và đánh giá
- Vấn đề gặp phải và giải quyết
- Kết luận và hướng phát triển
- Tài liệu tham khảo

Giới thiệu bài toán

Bài toán: Dùng mạng GAN sinh ra các ảnh giống với dữ liệu trong CIFAR-10 dataset.

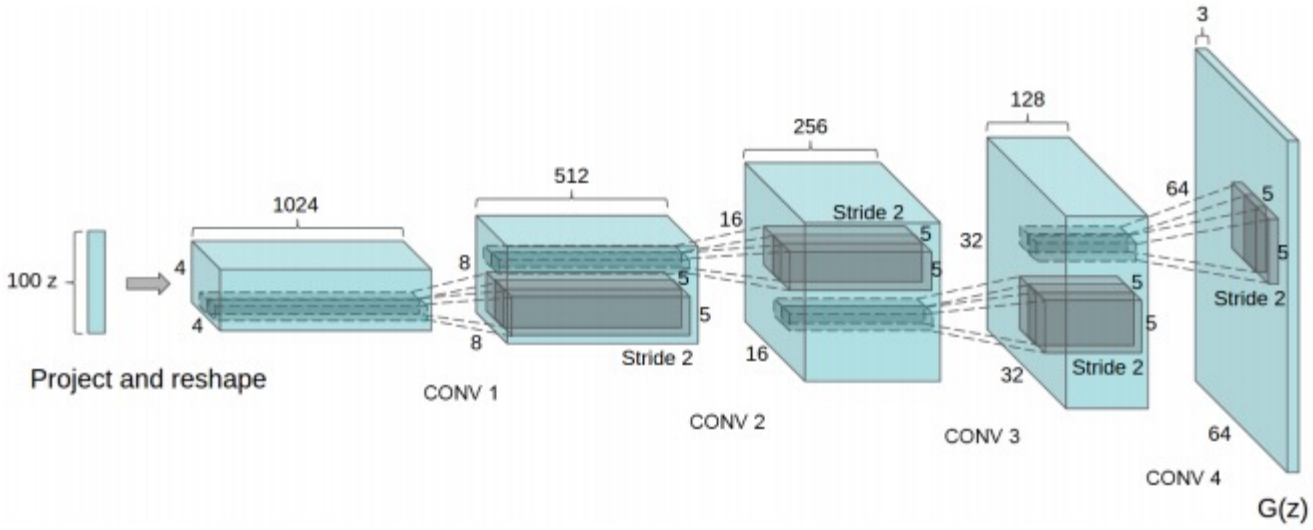


Nguồn: https://www.cs.toronto.edu/~kriz/cifar.html

Bài báo liên quan

Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks

Conference paper at ICLR 2016



Nguồn: https://arxiv.org/pdf/1611.04076v3.pdf

Bài báo liên quan

Least Squares Generative Adversarial Networks

ICCV 2017 · Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, Stephen Paul Smolley

(a) Church outdoor. (b) Dining room. (c) Kitchen. (d) Conference room.

Nguồn: https://arxiv.org/pdf/1511.06434v2.pdf

Generative Adversarial Networks(GAN)

- GAN thuộc nhóm generative model.

Nguồn: https://towardsdatascience.com/an-end-to-end-introduction-to-gans-bf253f1fa52f

- GAN là mạng để sinh dữ liệu mới giống với dữ liệu trong dataset có sẵn và có 2 mạng trong GAN là Generator và Discriminator.

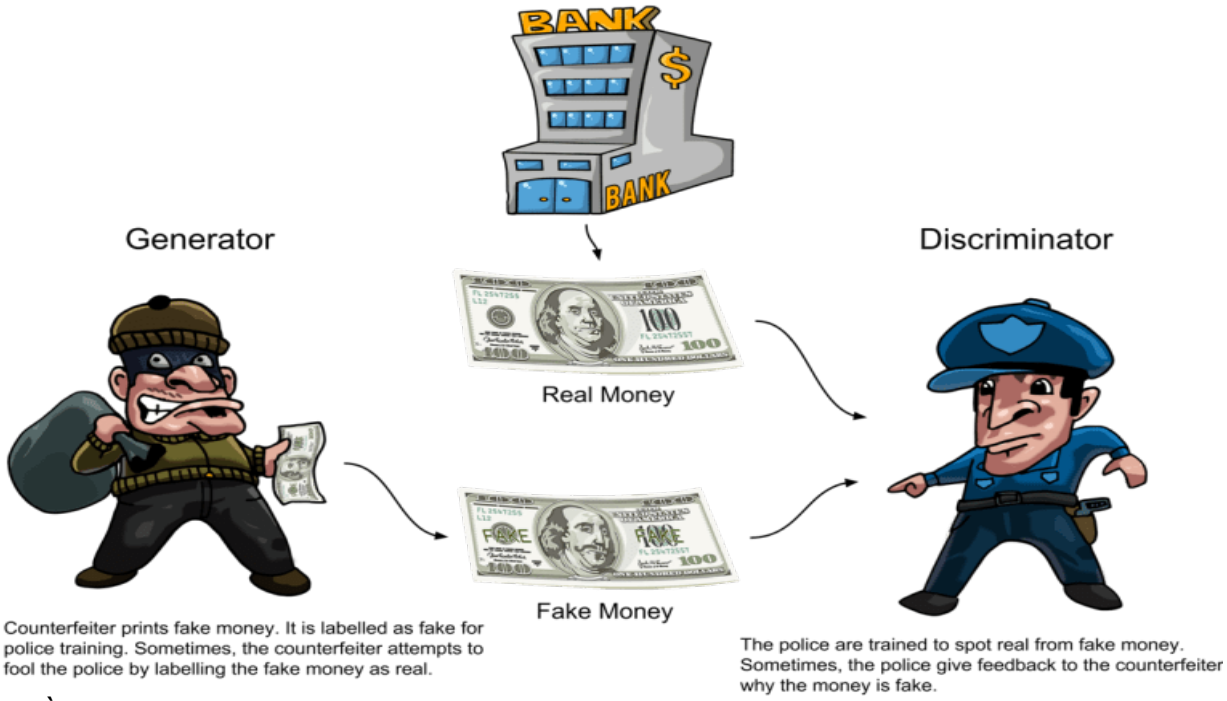
Generative Adversarial Networks(GAN)

- Cấu trúc mạng GAN:

  GAN cấu tạo gồm 2 mạng là Generator và Discriminator.

  (1) Generator sinh ra các dữ liệu giống như thật.

  (2) Discriminator cố gắng phân biệt đâu là dữ liệu được sinh ra từ Generator và đâu là dữ liệu thật có.



Nguồn: https://dzone.com/articles/working-principles-of-generative-adversarial-netwo

Generative Adversarial Networks(GAN)

Ý tưởng của GAN bắt nguồn từ **zero-sum non-cooperative game**

'*A strategy profile is a Nash equilibrium if no player can do better by unilaterally changing his or her strategy.'*

Nguồn: https://www.cantorsparadise.com/the-nash-equilibrium-explained-c9ad7e97633a

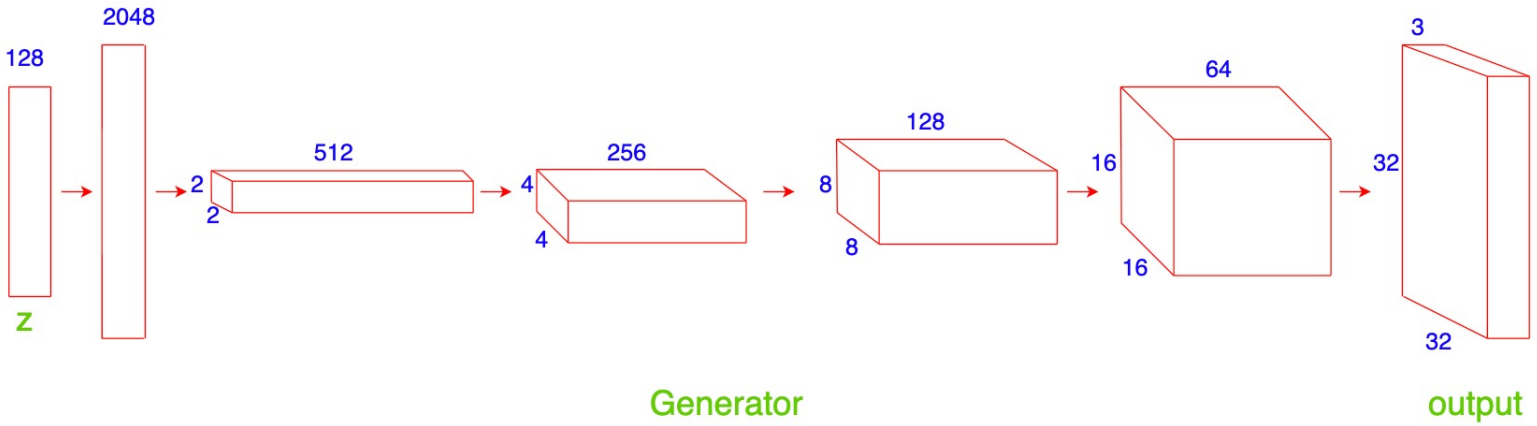
Deep Convolutional GAN (DCGAN)

- Cấu trúc mạng

  Generator và Discriminator được xây dựng bằng mô hình CNN với 2 layers chính là convolutional layer và transposed convolutional layer.

  - Generator

  Mạng Generator nhằm mục đích sinh ảnh fake, input là noise vector kích thước 128 và output là ảnh fake cùng kích thước ảnh thật (32 * 32 * 3).



Mô hình generator của DCGAN

Deep Convolutional GAN (DCGAN)

- Transposed convolution

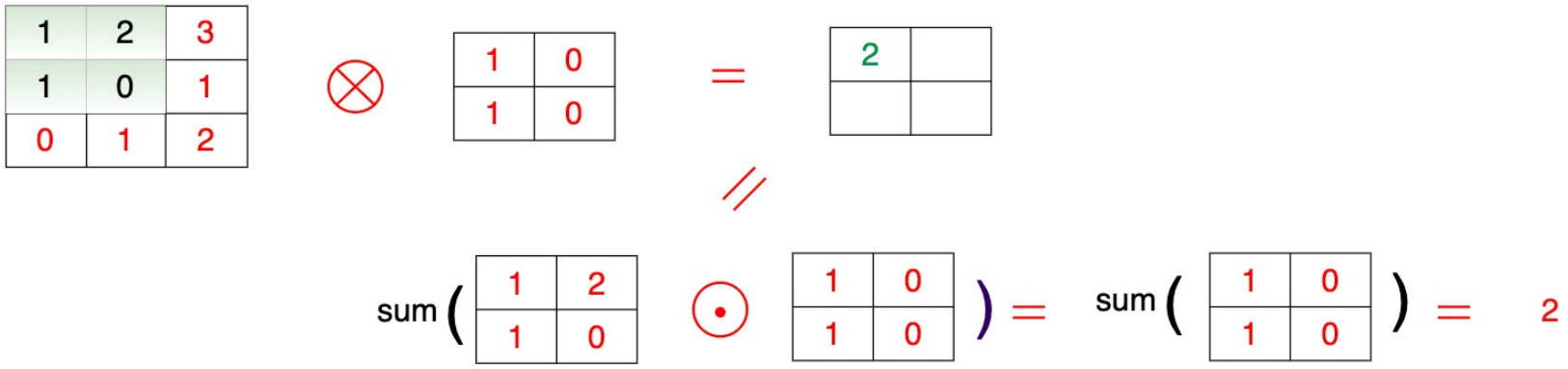
  - Transposed convolution hay deconvolution có thể coi là phép toán ngược của convolution.

  - Nếu như convolution với stride > 1 giúp làm giảm kích thước của ảnh thì transposed convolution với stride > 1 sẽ làm tăng kích thước ảnh. Ví dụ stride = 2 và padding = ‘SAME’ sẽ giúp gấp đôi width, height kích thước của ảnh.

  - Transposed convolution có 2 kiểu định nghĩa.

Deep Convolutional GAN (DCGAN)

- Kiểu 1
  - Ý tưởng đơn giản là transposed convolution là phép tính ngược của convolution.



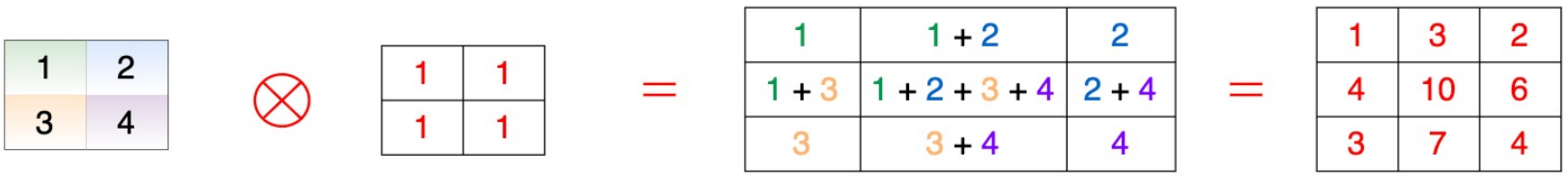
Convolution s=1, p=0

Tham khảo: http://d2l.ai/chapter_computer-vision/transposed-conv.html

Deep Convolutional GAN (DCGAN)

❖ Kiểu 1

➢ Nếu các phần tử ở output viết đè lên nhau thì ta sẽ cộng dồn vào.



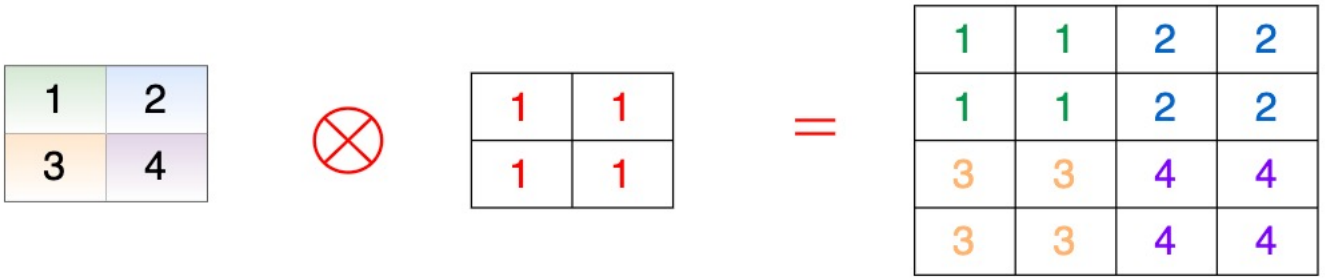
Transposed convolution với s=1, p=0

Tham khảo: http://d2l.ai/chapter_computer-vision/transposed-conv.html

Deep Convolutional GAN (DCGAN)

❖ Kiểu 1

➢ Stride trong transposed convolution được định nghĩa là số bước nhảy khi viết kết quả ra ma trận output.



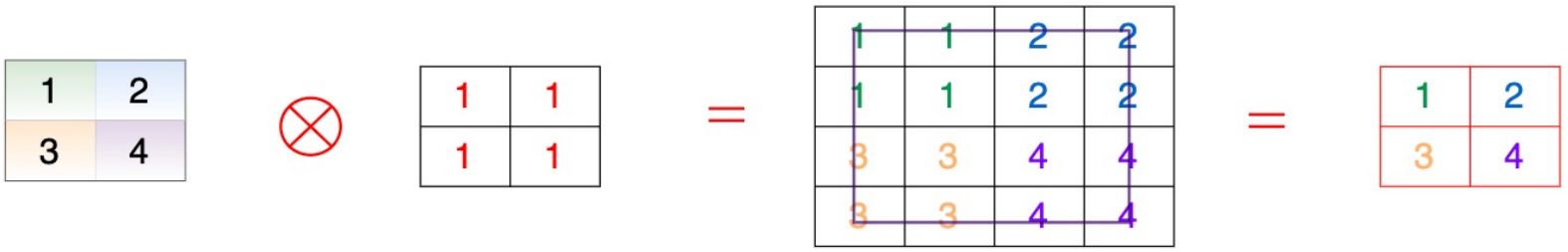
Transposed convolution với s=2, p=0

Tham khảo: http://d2l.ai/chapter_computer-vision/transposed-conv.html

Deep Convolutional GAN (DCGAN)

❖ Kiểu 1

➢ Với padding thì ta tính toán bình thường như với p = 0 sau đó kết quả ta sẽ bỏ p hàng và cột ở 4 cạnh (trên, dưới, trái, phải).



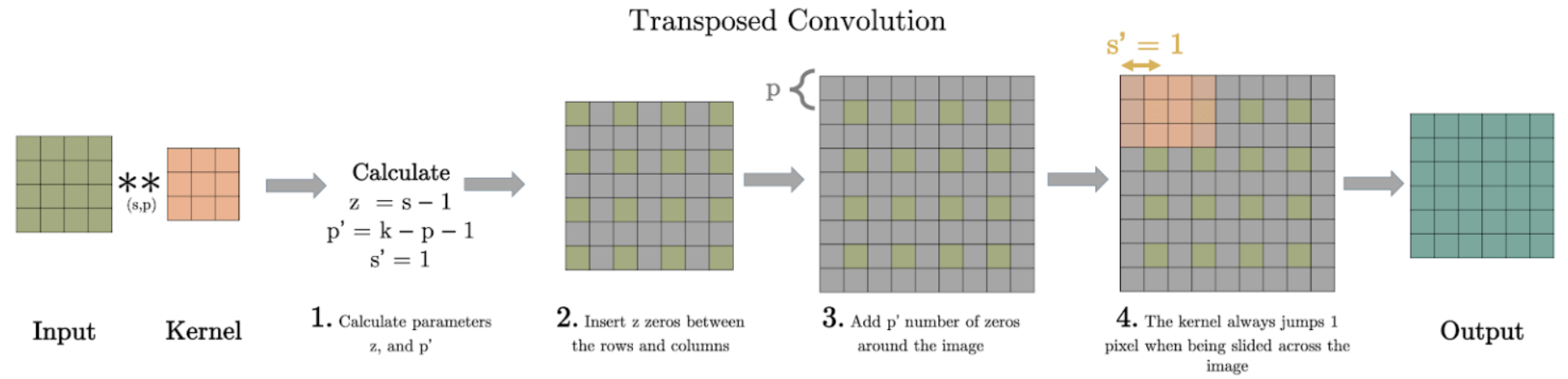
Transposed convolution với s=2, p=1

Tham khảo: http://d2l.ai/chapter_computer-vision/transposed-conv.html

Deep Convolutional GAN (DCGAN)

❖ Kiểu 2

- Kiểu định nghĩa thứ 2 thì phức tạp hơn nhưng lại có vẻ chuẩn và hay gặp hơn. Nguồn ở đây *(https://github.com/vdumoulin/conv_arithmetic).*
- Ý nghĩa của stride và padding ở đây là khi ta thực hiện phép tính convolution trên output sẽ được kích thước giống input.



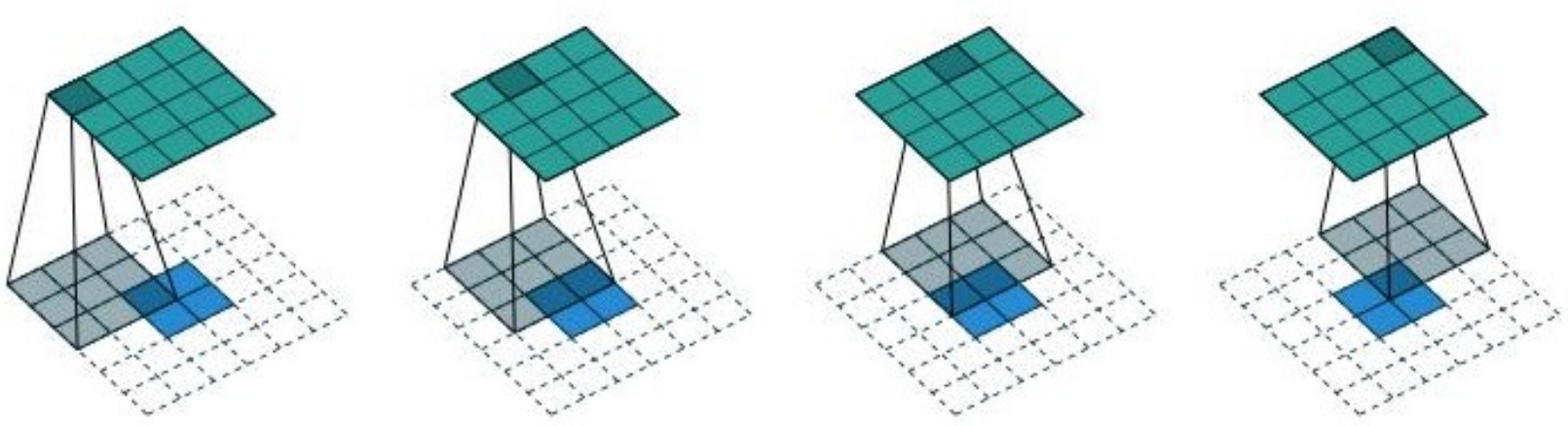
Nguồn: https://www.treyoehmler.com/unet

Các bước thực hiện transposed convolution

Deep Convolutional GAN (DCGAN)

❖ Kiểu 2

- Ta thấy phép tính convolution với input 4*4 , kernel size 3*3 , s = 1, p = 0 thì output kích thước 2*2. Ngược lại phép tính transposed convolution : input là 2*2 , kernel size là 3*3 , s = 1, p = 0 thì output sẽ có kích thước 4*4.

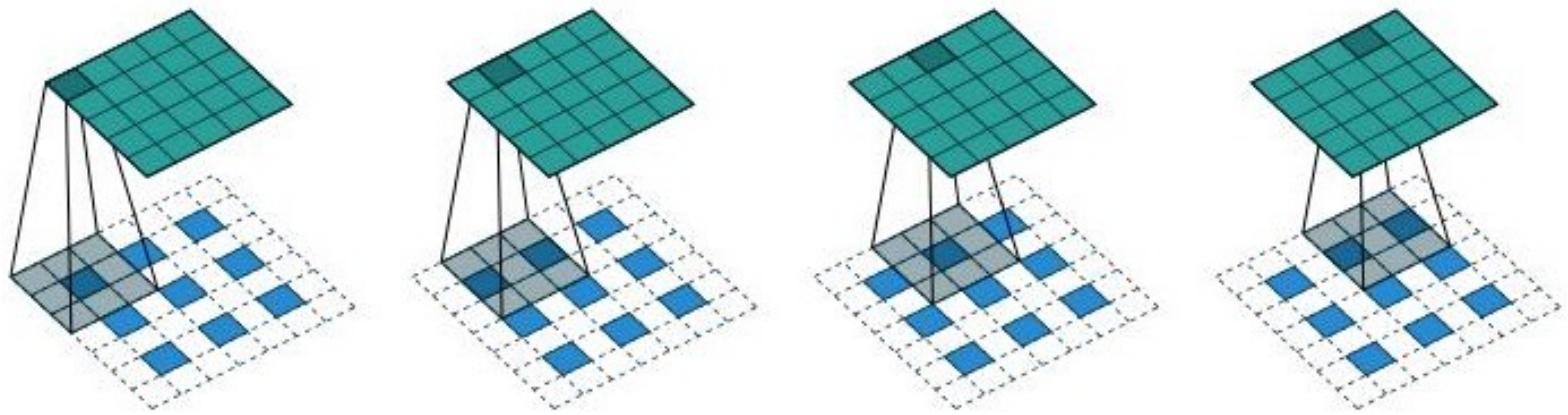


Nguồn: https://www.programmersought.com/article/20773765303/

Transposed convolution s=1, p=0

Deep Convolutional GAN (DCGAN)

- Kiểu 2
  - z = s – 1 = 1 hàng / cột chèn vào giữa, p’ = k – p – 1 = 1 padding input. Màu xanh dương là input, xanh lá cây đậm là output. Nếu check lại thì phép tính convolution với input 5*5, kernel size 3*3, s = 2, p = 1 sẽ được output kích thước 3*3.



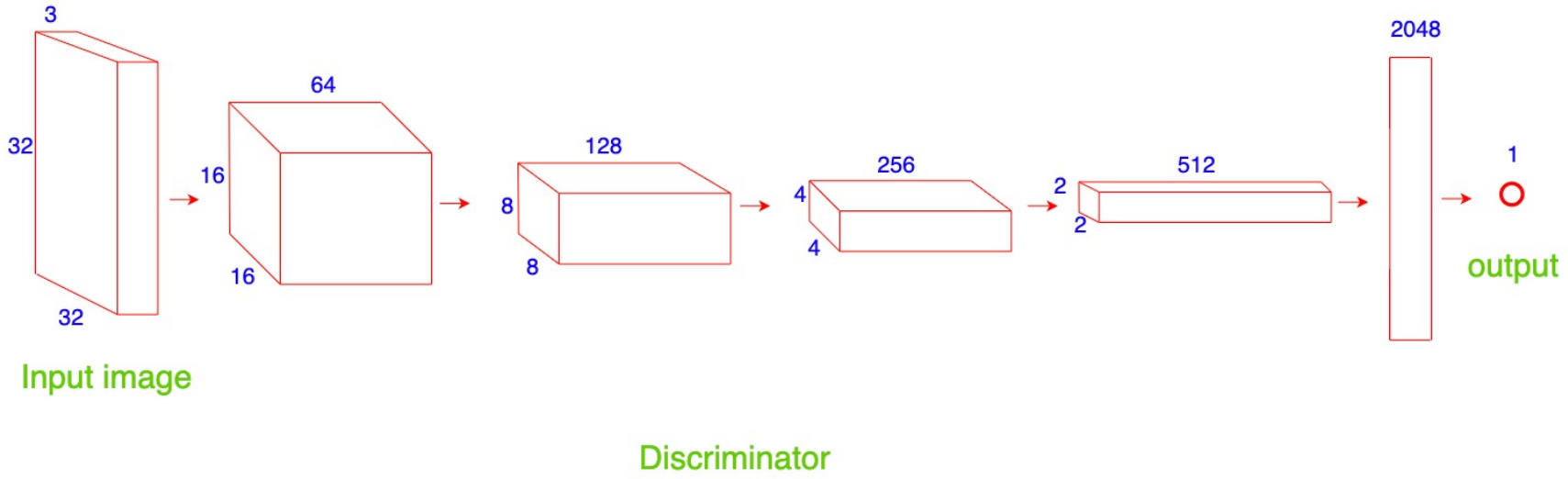
Nguồn: https://www.programmersought.com/article/20773765303/

Transposed convolution s=2, p=1

Deep Convolutional GAN (DCGAN)

- Cấu trúc mạng
  - Discriminator

Mạng Discriminator nhằm mục đích phân biệt ảnh thật từ dataset và ảnh fake do Generator sinh ra, input là ảnh kích thước (32 * 32 *3), output là ảnh thật hay fake (binary classification).



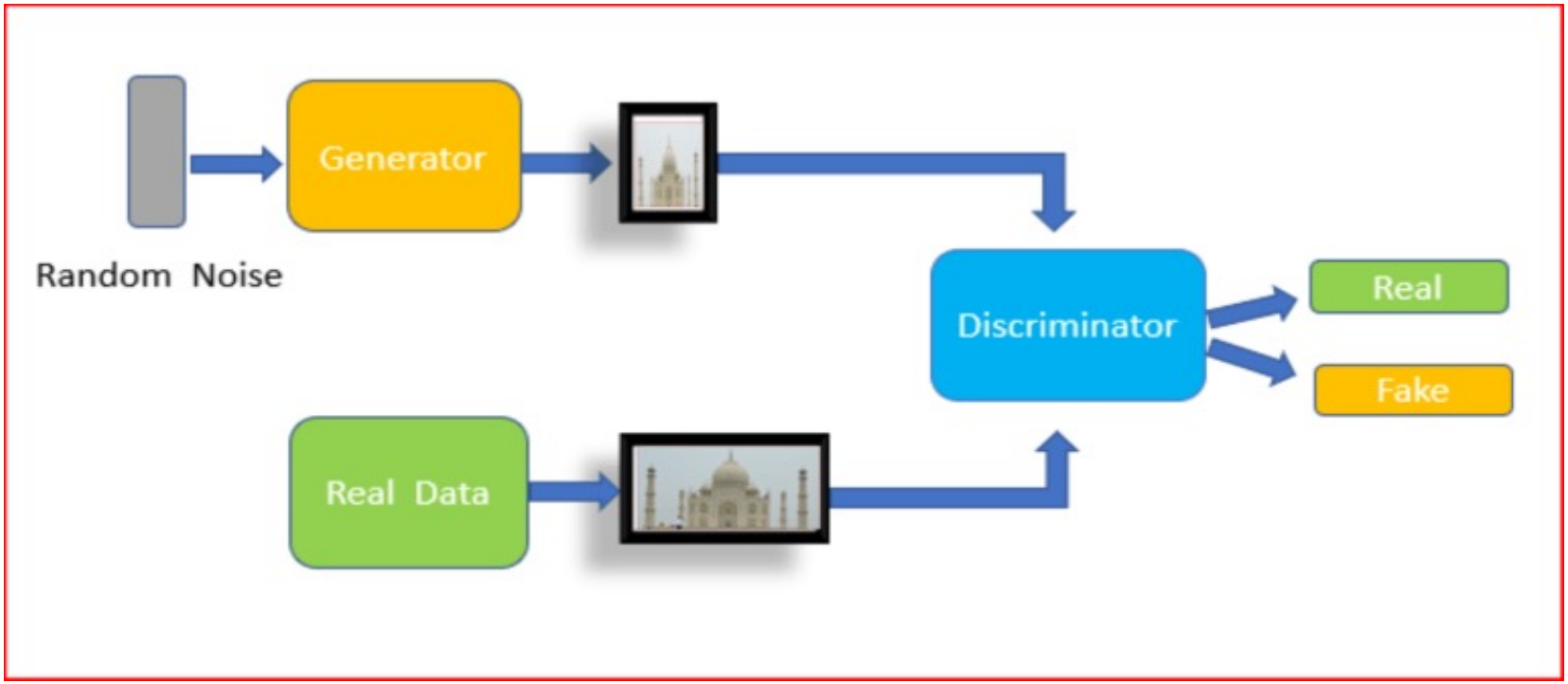
Mô hình discriminator của DCGAN

Deep Convolutional GAN (DCGAN)

- Loss function
  - Kí hiệu z là noise đầu vào của generator, x là dữ liệu thật từ bộ dataset.
  - Kí hiệu mạng Generator là G, mạng Discriminator là D. **G(z)** là ảnh được sinh ta từ Generator. **D(x)** là giá trị dự đoán của Discriminator xem ảnh x là thật hay không, **D(G(z))** là giá trị dự đoán xem ảnh sinh ra từ Generator là ảnh thật hay không.
  - Vì ta có 2 mạng Generator và Discriminator với mục tiêu khác nhau, nên cần thiết kế 2 loss function cho mỗi mạng.
  - **Discriminator** thì cố gắng phân biệt đâu là ảnh thật và đâu là ảnh giả. Vì là bài toán binary classification nên loss function dùng giống với *binary cross-entropy loss* của sigmoid.

Deep Convolutional GAN (DCGAN)

- Loss function
  - Giá trị output của model qua hàm sigmoid sẽ trong (0, 1) nên Discriminator sẽ được train để input ảnh ở dataset thì output gần 1, còn input là ảnh sinh ra từ Generator thì output gần 0, hay $D(x) \rightarrow 1$ còn $D(G(z)) \rightarrow 0$.



Nguồn: https://medium.com/datadriveninvestor/generative-adversarial-network-gan-using-keras-ce1c05cfdfd3

Deep Convolutional GAN (DCGAN)

- Loss function
  - Hay nói cách khác là loss function muốn maximize D(x) và minimize D(G(z)). Ta có minimize D(G(z)) tương đương với maximize (1 – D(G(z))). Do đó loss function của Discriminator được viết lại thành.

$$\max_{D} V(D) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

recognize real images better *recognize generated images better*

Loss Discrimnator. *Nguồn: https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf*

E là kì vọng, hiểu đơn giản là lấy trung bình của tất cả dữ liệu, hay maximize D(x) với x là dữ liệu trong traning set.

Deep Convolutional GAN (DCGAN)

- Loss function
  - Generator sẽ học để đánh lừa Discriminator xem ảnh nó sinh ra là ảnh thật, hay D(G(z)) → 1. Hay loss function muốn maximize D(G(z)), tương đương với minimize (1 – D(G(z))).

$$\min_{D} V(G) = E_{z \sim p_z(z)}[log(1 - D(G(z)))]$$

Optimize G that can fool the discriminator the most.

Loss Generator. *Nguồn: https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf*

Deep Convolutional GAN (DCGAN)

- Loss function
  - Do đó ta có thể viết gộp lại loss của mô hình GAN:

$$\min_G \max_D V(D,G) = E_{x \sim p_{data}(x)}[logD(x)] + E_{z \sim p_z(z)}[log(1 - D(G(z)))]$$

Loss GAN. *Nguồn: https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf*

  - Từ hàm loss của GAN có thể thấy là việc train Generator và Discriminator đối nghịch nhau, trong khi D cố gắng maximize loss thì G cố gắng minimize loss.
  - Quá trình train GAN kết thúc khi model GAN đạt đến trạng thái cân bằng của 2 models, gọi là Nash equilibrium.

Deep Convolutional GAN (DCGAN)

- Loss function
  - Do đó ta có thể viết gộp lại loss của mô hình GAN:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Loss GAN. *Nguồn: https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf*

  - Từ hàm loss của GAN có thể thấy là việc train Generator và Discriminator đối nghịch nhau, trong khi D cố gắng maximize loss thì G cố gắng minimize loss.
  - Quá trình train GAN kết thúc khi model GAN đạt đến trạng thái cân bằng của 2 models, gọi là Nash equilibrium.

Deep Convolutional GAN (DCGAN)

- Tips
  - Đây là một số tips để build model và train DCGAN:
    - Dùng ReLU trong generator trừ output layer.
    - Output layer trong generator dùng tanh (-1, 1) và scale ảnh input về (-1,1) sẽ cho kết quả tốt hơn dùng sigmoid và scale ảnh về (0, 1) hoặc để nguyên ảnh.
    - Sử dụng LeakyReLU trong discriminator.
    - Thay thế max pooling bằng convolution với stride = 2.
    - Sử dụng transposed convolution để upsampling.
    - Sử dụng batch norm từ output layer trong generator và input layer trong discriminator.

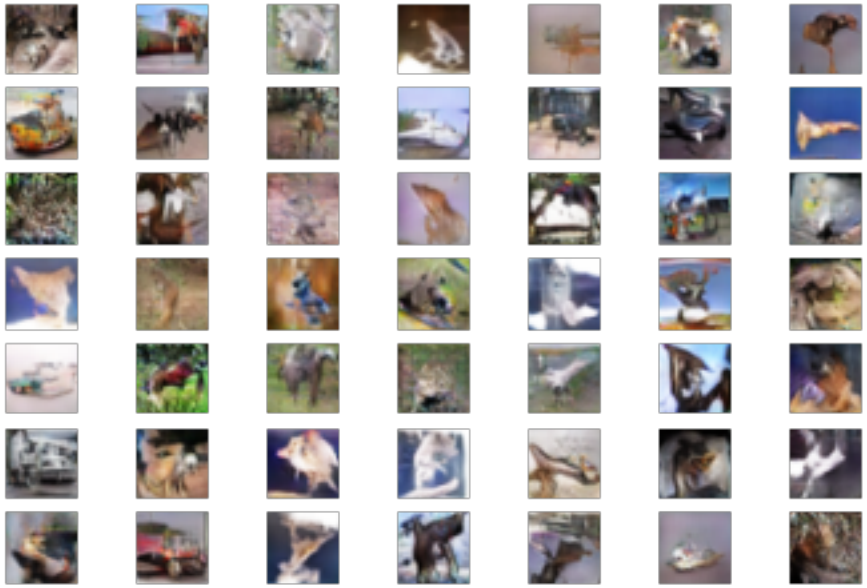
Thực nghiệm và đánh giá

- Ảnh CIFAR-10 được scale về (-1, 1) để cùng scale với ảnh sinh ra bởi generator khi dùng tanh activation.
- Ở những epoch đầu tiên thì generator chỉ sinh ra noise.

Ảnh sinh ra bởi generator sau 10 epochs

Thực nghiệm và đánh giá

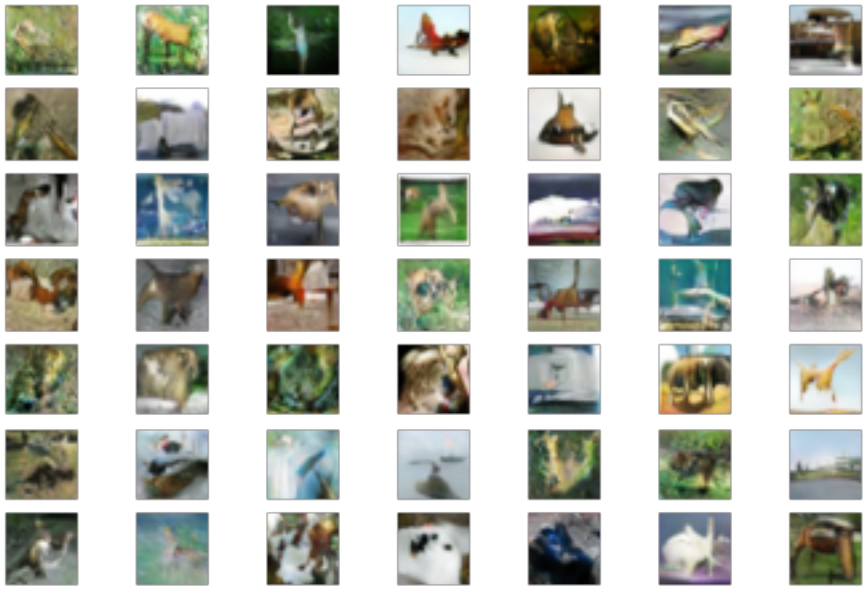
- Tuy nhiên sau 100 epoch thì mạng đã học được thuộc tính của ảnh trong dữ liệu CIFAR-10 và có thể sinh ra được hình con chim, xe tải,..



Ảnh sinh ra bởi generator sau 100 epochs

Thực nghiệm và đánh giá

- Và sau 200 epoch thì mạng đã học được thuộc tính của ảnh trong dữ liệu CIFAR-10 và có thể sinh ra được hình xe tải, con nai, con chim,..



Ảnh sinh ra bởi generator sau 200 epochs

Vấn đề gặp phải và giải quyết

- Vấn đề 1:
  - Khi ta train DCGAN xong rồi dùng generator để sinh ảnh mới giống trong dataset mình không kiểm soát được là ảnh sinh ra giống category nào trong dataset.
  - Ví dụ như dùng DCGAN để sinh các ảnh trong bộ CIFAR-10, thì khi train xong và dùng generator sinh ảnh thì mình không biết được ảnh sinh ra đó thuộc class nào (Airplane, Car, Bird, Cat, Deer, Dog, Frog, Horse, Ship, Truck)
  - Bài toán muốn kiểm soát được generator sinh ra ảnh theo 1 category nhất định. Ví dụ có thể chỉ định generator sinh ra ảnh con **nai** chẳng hạn. Mô hình đấy gọi là Conditional GAN (cGAN).
  - Đây có thể được xem như một bước đột phát của GAN vì trên thực tế có rất nhiều những bức ảnh mà ta sẽ phải định hướng kết quả về hình dạng, format. cGAN cũng tạo ra những đột phá mới về chất lượng hình ảnh và sự ổn định trong quá trình huấn luyện.

Vấn đề gặp phải và giải quyết

- Vấn đề 2:
  - Cấu trúc mạng DCGAN với thành phần là Generator và Discriminator, GAN loss function.
  - Tuy nhiên GAN loss function không tốt, nó bị vanishing gradient khi train generator

→ Hàm LSGAN (Least Squares Generative Adversarial Networks) giải quyết vấn đề trên.

LSGAN

- Vấn đề với GAN loss function:
  - Bổ đề:

$$Xét\ hàm\ f(x) = a * \log(x) + b * \log(1 - x)\ .Tìm\ x\ sao\ cho\ hàm\ f(x)\ cực\ đại.$$

$$Ta\ có:\ f'(x) = 0 = \frac{a}{x} - \frac{b}{1 - x}$$

$$f'(x) = 0 \leftrightarrow \frac{a}{x} = \frac{b}{1 - x} \leftrightarrow x = \frac{a}{a + b}$$

$$Do\ đó\ f(x) cực\ đại\ khi\ x = x^* = \frac{a}{a + b}$$

LSGAN

- Vấn đề với GAN loss function:
  - GAN loss function:

    x là ảnh thật, $p_r(x)$là distribution sinh ra ảnh thật, z là noise, $p_z(z)$ là distribution sinh ra noise, G là generator, D là discriminator. GAN loss function:

$$\min_G \max_D \quad V(D,G) = E_{x\sim p_r(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1 - D(G(z)))]$$
$$= E_{x\sim p_r(x)}[logD(x)] + E_{x\sim p_g(x)}[log(1 - D(x))]$$
$$\min_G \max_D \quad V(D,G) = \int_x (p_r(x)logD(x) + p_g(x)log(1 - D(x)))dx$$

Ta thấy $\min_G \max_D V(D,G)$ có 2 việc tìm D để max V và tìm G để min V.

Trước tiên ta tìm D để $maxV(D,G)$ với G là hằng số.

Trong đó $p_r(x)$ là distribution sinh ra ảnh thật, $p_g(x)$ là distribution sinh ra ảnh fake (chính là generator).

LSGAN

- Vấn đề với GAN loss function:
  - GAN loss function:

Áp dụng bổ đề ta có, V max khi:

$$D(x) = D^*(x) = \frac{p_r(x)}{p_r(x)+p_g(x)}$$

Ta thay D(x) vào V để tìm $\min_G V(D^*, G)$

$$\min_G V(D^*, G) = \int_x \left(p_r(x) log D^*(x) + p_g(x) log(1 - D^*(x))\right) dx$$

$$= \int_x \left(p_r(x)\, log \frac{p_r(x)}{p_r(x)+p_g(x)} + p_g(x) log(\frac{p_g(x)}{p_r(x)+p_g(x)})\right) dx$$

Jensen–Shannon divergence (JS), Kullback–Leibler divergence (KL) dùng để tính độ tương đồng giữa hai distribution, giá trị nhỏ nhất là 0 khi 2 distribution giống nhau.

LSGAN

- Vấn đề với GAN loss function:
  - GAN loss function:

Ta có

$$D_{JS}(p_r||p_g) = \frac{1}{2}D_{KL}\left(p_r||\frac{p_r+p_g}{2}\right) + \frac{1}{2}D_{KL}\left(p_g||\frac{p_r+p_g}{2}\right)$$

$$= \frac{1}{2}\left(\int_x p_r(x)log\left(\frac{2p_r(x)}{p_r(x)+p_g(x)}\right)\right)dx \; + \frac{1}{2}\left(\int_x (p_g(x)log\left(\frac{2p_g(x)}{p_r(x)+p_g(x)}\right)\right)dx$$

$$= \frac{1}{2}\left(log2 + \int_x p_r(x)log\left(\frac{p_r(x)}{p_r(x)+p_g(x)}\right)dx\right) +$$

$$\frac{1}{2}\left(log2 + \int_x (p_g(x)log\left(\frac{p_g(x)}{p_r(x)+p_g(x)}\right)dx\right)$$

$$= \frac{1}{2}(log4 + \min_G V(D^*, G))$$

LSGAN

- Vấn đề với GAN loss function:
  - GAN loss function:

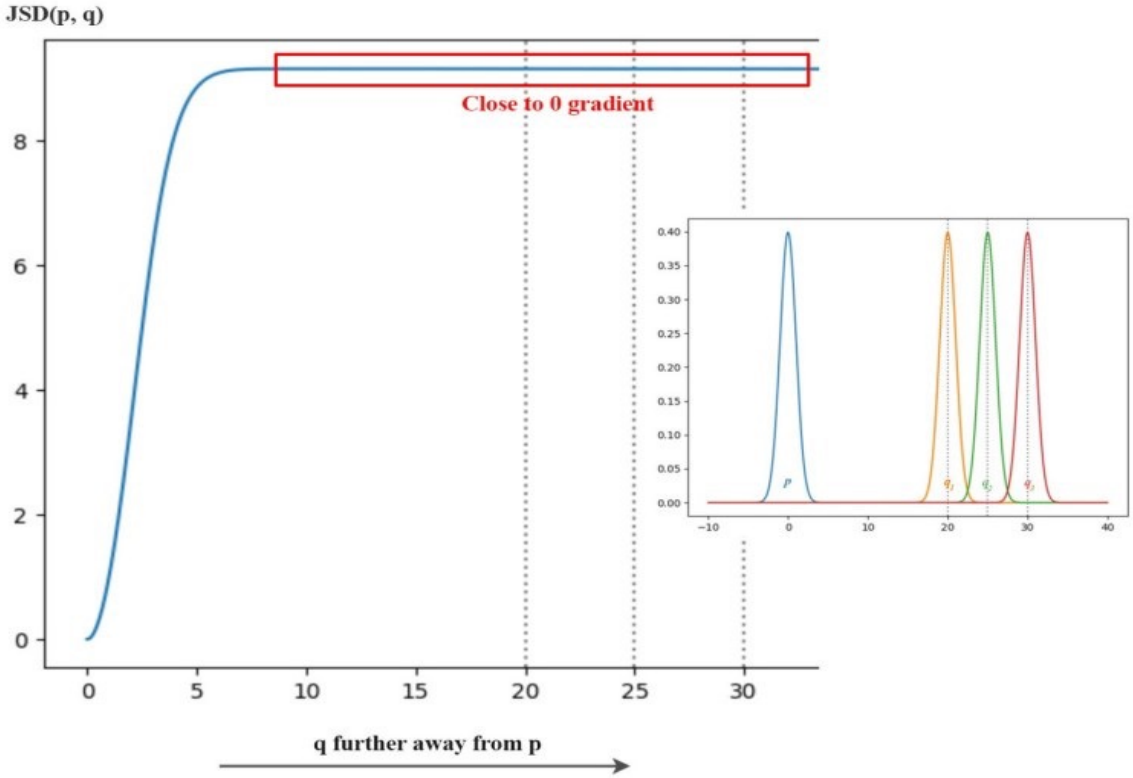
    Do đó:

$$\min_G V(D^*, G)) = 2D_{JS}(p_r || p_g) - log4$$, do đó:

$$\min_G V = -log4 \; khi \; p_r = p_g \; (D_{JS} = 0)$$

LSGAN

- Vấn đề với GAN loss function:
  - GAN loss function:



Trục x là khoảng cách mean giữa distribution p và q, trục y là giá trị JS giữa p và q.

Nguồn: https://medium.com/@jonathan_hui/gan-lsgan-how-to-be-a-good-helper-62ff52dd3578

LSGAN

- Vấn đề với GAN loss function:
  - GAN loss function:

    Nhận xét: Khi p_r, p_g xa nhau (không giống nhau) thì đạo hàm của $\mathrm{D}_{\mathrm{JS}}(p_r \,||\, p_g)$về 0 dẫn tới **vanishing gradient**.

    Khi mới bắt đầu train thì Generator sinh ra ảnh nhiễu nên rất dễ cho discriminator học để phân biệt ảnh thật và giả, tuy nhiên p_r, p_g rất khác dẫn tới vanishing gradient nên việc học của Generator không tốt.

    Do đó loss GAN truyền thống thì Discriminator học rất tốt nhưng thường không tốt cho Generator → Cần hàm loss function tốt hơn để train GAN model.

LSGAN

- Least squares GAN (LSGAN):
  - LSGAN được thiết kế để train generator tốt hơn. LSGAN được định nghĩa như sau:

$$\min_D V_{LSGAN}(D) = \frac{1}{2}E_{x\sim p_{data}(x)}[(D(x) - b)^2] + \frac{1}{2}E_{z\sim p_z(z)}[(D(G(x)) - a)^2] \quad (1)$$

$$\min_G V_{LSGAN}(G) = \frac{1}{2}E_{z\sim p_z(z)}[(D(G(z)) - c)^2] \quad (2)$$

Trong đó:

a và b là target label của ảnh sinh ra từ generator (G(z)) và dataset (x) khi train discriminator.

c là target label của G(z) khi train Generator.

LSGAN

- Least squares GAN (LSGAN):

Do $\min_G V_{LSGAN}(G)$ không phụ thuộc vào D nên ta có thể viết (2) lại thành:

$$\min_G V_{LSGAN}(G) = \frac{1}{2}E_{z\sim p_{data}(x)}[(D(x) - c)^2] + \frac{1}{2}E_{z\sim p_z(z)}[(D(G(x)) - c)^2] \quad (3)$$

Từ (1) ta thấy $\min_D V_{LSGAN}(D)$ nhỏ nhất khi: $D(x) = D^*(x) = \frac{bp_{data}(x)+ap_g(x)}{p_{data}(x)+p_g(x)}$ (4)

Thay (4) vào (3) với b – c = 1 và b – a = 2 ta được:

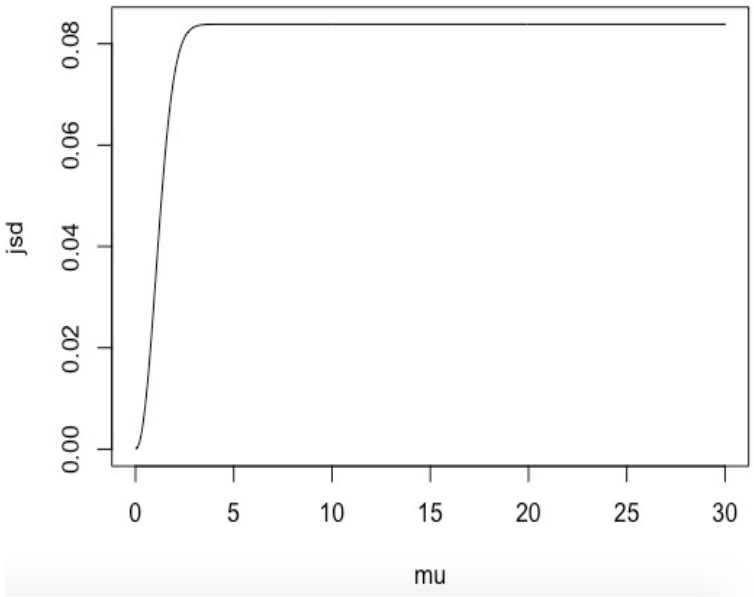
$$\min_G V_{LSGAN}(G) = \int_x \frac{(2p_g(x) - (p_d(x) + p_g(x))^2}{p_d(x) + p_g(x)} dx = X^2_{Pearson}(p_d + p_g || 2p_g)$$

Trong đó: $X^2_{Pearson}$ là **Pearson X^2 divergence**.

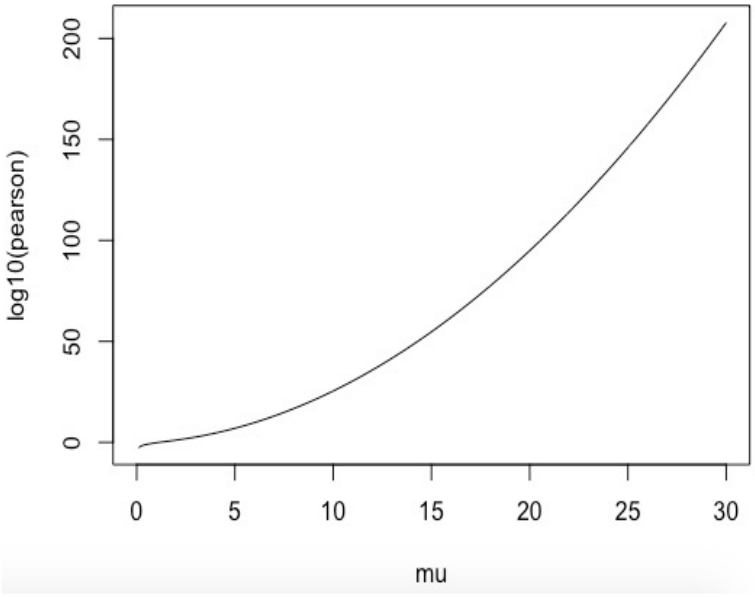
(*Tham khảo: https://en.wikipedia.org/wiki/F-divergence*)

LSGAN

- Least squares GAN (LSGAN):



Jensen–Shannon divergence



Pearson chi-square divergence

So sánh giữa JSD và $\text{Pearson } \chi^2$ divergence.

Nhận thấy là khi p_r, p_g xa nhau thì đạo hàm của $D_{JS}(p_d \,||\, p_g)$ về 0 dẫn tới vanishing gradient, tuy nhiên $\text{Pearson } \chi^2$ divergence thì vẫn có đạo hàm và tránh được vanishing gradient.

LSGAN

- Least squares GAN (LSGAN):

Để thõa mãn b – c = 1 và b – a = 2, ta chọn b = 1, c = 0, a = -1; do đó LSGAN được viết lại thành:

$$\min_D \; V_{LSGAN}(D) = \frac{1}{2}E_{x\sim p_{data}(x)}[\,(D(x)\,-1)^2] + \frac{1}{2}E_{z\sim p_z(z)}[(D(G(x))+1)^2]$$

$$\min_G \; V_{LSGAN}(G) = \frac{1}{2}E_{z\sim p_z(z)}[(D(G(z)))^2]$$

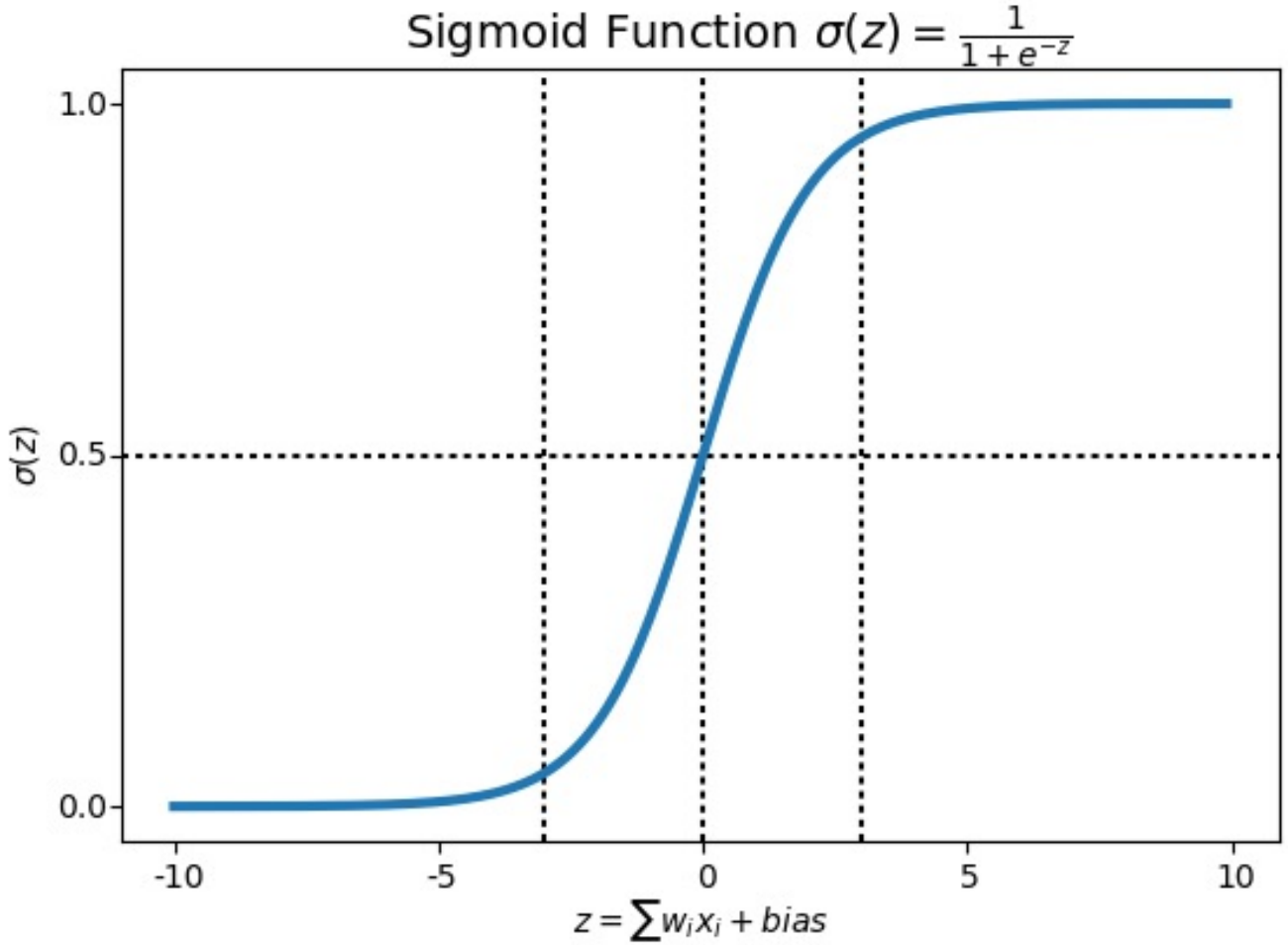
Ngoài ra có thể chọn b= c để ảnh fake sinh ra giống ảnh thật, mình có thể dùng ý tưởng hàm sigmoid ra giá trị từ (0, 1) sẽ để ảnh fake về 0 và ảnh real về 1, nhưng dùng MSE để tối ưu, chọn b = c = 1, a = 0. Ta có hàm LSGAN thay thế:

$$\min_D \; V_{LSGAN}(D) = \frac{1}{2}E_{x\sim p_{data}(x)}[\,(D(x)\,-1)^2] + \frac{1}{2}E_{z\sim p_z(z)}[(D(G(x)))^2]$$

$$\min_G \; V_{LSGAN}(G) = \frac{1}{2}E_{z\sim p_z(z)}\big[(D\big(G(z)\big)-1)^2\big]$$

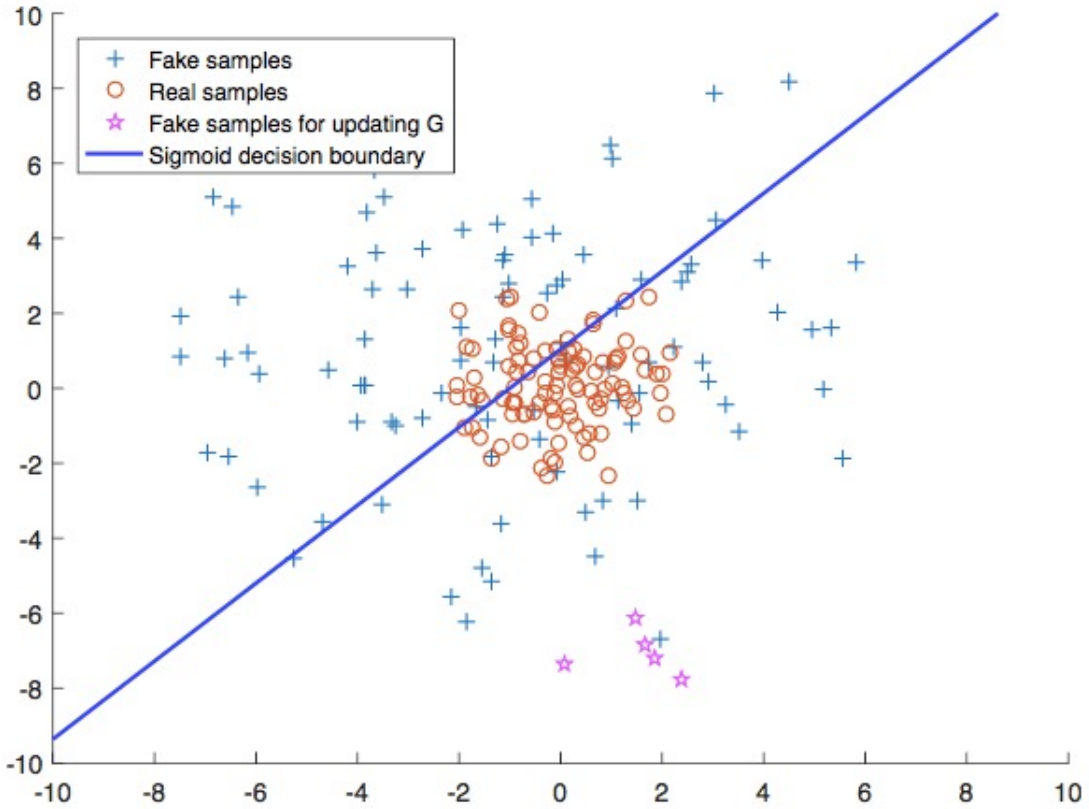
LSGAN

- Trực quan hóa:



LSGAN

- Trực quan hóa:

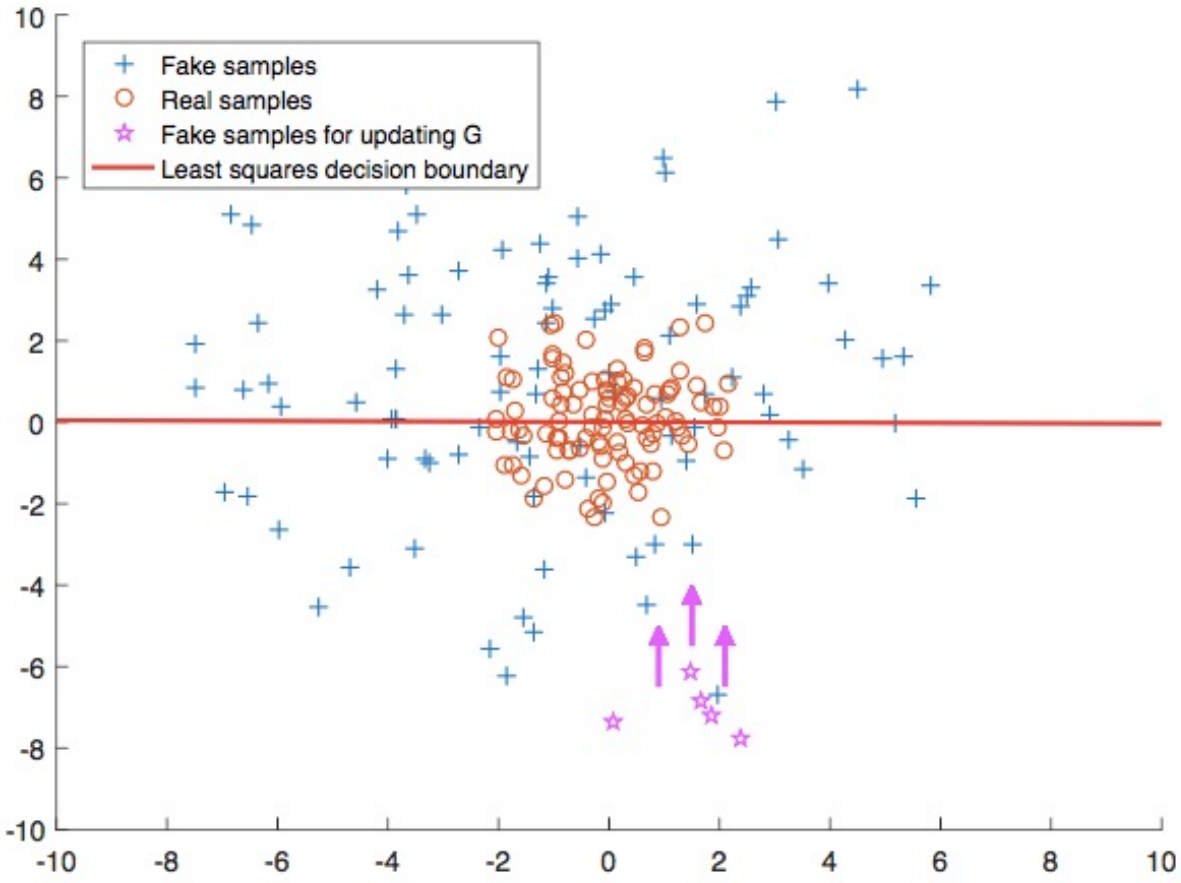


Đường phân chia Discriminator

Nguồn: https://arxiv.org/pdf/1611.04076v3.pdf

LSGAN

- Trực quan hóa:



LSGAN

Vấn đề gặp phải và giải quyết

- Vấn đề 3: GAN evaluation
  - Mình dùng DCGAN train model xong rồi dùng Generator để sinh ảnh. Tuy nhiên mình chưa có cách nào để đánh giá xem chất lượng ảnh sinh ra đã tốt chưa, chủ yếu chỉ nhìn vào mắt thường để đánh giá.
  - Cách đấy không khách quan cũng như không áp dụng một cách hệ thống trên dữ liệu lớn, nên cần phương pháp để đánh giá chất lượng ảnh GAN - DCGAN sinh ra chất lượng tốt hay không.
  - Một số cách để đánh giá chất lượng ảnh GAN (DCGAN) sinh ra: Inception Score (IS) và Fréchet Inception Distance (FID).

Kết luận và hướng phát triển

- DCGAN (deep convolutional GAN) là mô hình GAN áp dụng trong các tác vụ của xử lý ảnh.
- Nhược điểm của DCGAN là chúng ta không thể kiểm soát được bức ảnh được sinh ra thuộc class nào mà nó được tạo ra hoàn toàn ngẫu nhiên.
- cGAN sẽ giúp chúng ta sinh ra được ảnh thuộc một class cụ thể theo ý muốn dựa trên một thông tin được bổ sung vào mô hình là nhãn *y*. *y* được coi như điều kiện để sinh ảnh nên mô hình mới có tên gọi là conditional GAN.
- GAN loss function không tốt, bị vanishing gradient khi train generator thì LSGAN đã giải quyết được vấn đề.
- LSGAN hoạt động ổn định hơn GAN thông thường trong quá trình học. Qua đó, chúng em sẽ vận dụng model LSGAN trên những bộ dữ liệu khác như ImageNet,…

Tài liệu tham khảo

- [Introduction - Google Developers]
  (https://developers.google.com/machine-learning/gan)
- [Generative Adversarial Networks Wiki]
  (https://en.wikipedia.org/wiki/Generative_adversarial_network)
- [Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks]
  (https://paperswithcode.com/paper/unsupervised-representation-learning-with-1)
- [Least Squares Generative Adversarial Networks]
  (https://paperswithcode.com/paper/least-squares-generative-adversarial-networks)
- [Bài 45 - Conditional GAN (cGAN)]
  (https://phamdinhkhanh.github.io/2020/08/09/ConditionalGAN.html)

Tài liệu tham khảo

- [Book: Deep Learning cơ bản]
  (https://drive.google.com/file/d/1INjzISABdoc7SRq8tg-xkCRRZRABPCKi/view)
- [f-divergence wiki]
  (https://en.wikipedia.org/wiki/F-divergence)
- [GANs in Action: Deep learning with Generative Adversarial Networks 1st Edition]
  (https://www.amazon.com/GANs-Action-learning-Generative-Adversarial/dp/1617295566)
- [How to Train a GAN? Tips and tricks to make GANs work]
  (https://github.com/soumith/ganhacks)

Demo