ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

-oOo-

BÁO CÁO

MÔN: Lập trình Python cho Máy học - CS116.M11

Đề tài: Tìm hiểu Generative Adversarial Networks (GANs)

Giảng viên hướng dẫn: TS. Nguyễn Vinh Tiệp

Sinh viên thực hiện:	**Mã số sinh viên**
Trần Thanh Nguyên	19520192
Nguyễn Chí Cường	19521299

TP.HCM – 12/2021

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

MỤC LỤC

Contents

TÓM TẮT 4
1. CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN 5
1.1. Giới thiệu quá trình hình thành 5
1.2. Mục tiêu đề tài 5
1.3. Thông tin nhóm 6
1.4. Bảng phân công công việc 6
2. CHƯƠNG 2: LÝ THUYẾT 7
2.1. Generative Adversarial Networks (GANs) là gì? 7
2.2. Discriminative network 8
2.3. Generative network 8
2.4. Loss Function 9
2.5. Một vài kiến trúc GAN 10
2.5.1. CycleGAN 10
2.5.2. DiscoGAN 11
3. CHƯƠNG 3: THỰC TIỄN 12
4. CHƯƠNG 5: KẾT LUẬN 13
4.1. Kết luận 13
4.1.1. Ưu điểm 14
4.1.2. Nhược điểm 14
TÀI LIỆU THAM KHẢO 14

TÓM TẮT

Các model trước đây cũng dùng đến CNN đều làm được những điều rất tốt như phân biệt ngay lập tức được hàng trăm, hàng ngàn bước ảnh chó và mèo hoặc thẩm chí nhiều thứ khác. Nhưng điểm chung ở những bài toán trên là phải có gán nhãn (label), việc gán nhãn này làm tốn rất nhiều thời gian và công sức. Thế nên, Generative Adversarial Networks đã được sinh ra với kỳ vọng để tạo ra những model có độ chính xác cao mà không cần nhiều đến sự hoạt động của con người trong việc huấn luyện. (Unsupervised Learning). Trong bài báo cáo này sẽ tìm hiểu rõ về GANs là cái gì, GANs hoạt động như thế nào, tại sao lại dùng đến GANs.

1. CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN

1.1. Giới thiệu quá trình hình thành

- Ý tưởng đặt hai thuật toán chống lại nhau bắt nguồn từ Arthur Samuel, một nhà nghiên cứu nổi tiếng trong lĩnh vực khoa học máy tính, người được ghi nhận là người đã phổ biến thuật ngữ "machine learning". Khi ở IBM, ông đã nghĩ ra một trò chơi cờ caro - the Samuel Checkers-playing Program - là một trong những trò chơi đầu tiên tự học thành công, một phần bằng cách ước tính cơ hội chiến thắng của mỗi bên ở một vị trí nhất định.
- Nhưng nếu Samuel là ông của GAN, Ian Goodfellow, cựu nhà khoa học nghiên cứu Google Brain và giám đốc máy học tại Nhóm các dự án đặc biệt của Apple, có thể là cha của nó. Trong một bài báo nghiên cứu năm 2014 có tiêu đề đơn giản là " generative adversarial net ", Goodfellow và các đồng nghiệp mô tả việc triển khai hoạt động đầu tiên của một mô hình chung dựa trên các mạng đối địch.
- Goodfellow thường tuyên bố rằng anh ta lấy cảm hứng từ "noise-contrastive estimation", một cách học phân phối dữ liệu bằng cách so sánh nó với phân phối nhiễu xác định (tức là, một hàm toán học đại diện cho dữ liệu bị hỏng hoặc bị bóp méo). Noise-contrastive estimation sử dụng các hàm tổn thất giống như GAN - nói cách khác, cùng một phép đo hiệu suất liên quan đến khả năng dự đoán kết quả mong đợi của mô hình.

1.2. Mục tiêu đề tài

- Tìm hiểu về generative adversarial networks(Gans)
- Thực hành, thử nghiệm demo các vấn đề gặp phải khi ứng dụng

1.3. Thông tin nhóm

- Danh sách thành viên:

Bảng 1: Danh sách các thành viên trong nhóm

STT	MSSV	Họ và tên
1	19520192	Trần Thanh Nguyên
2	19521299	Nguyễn Chí Cường

1.4. Bảng phân công công việc

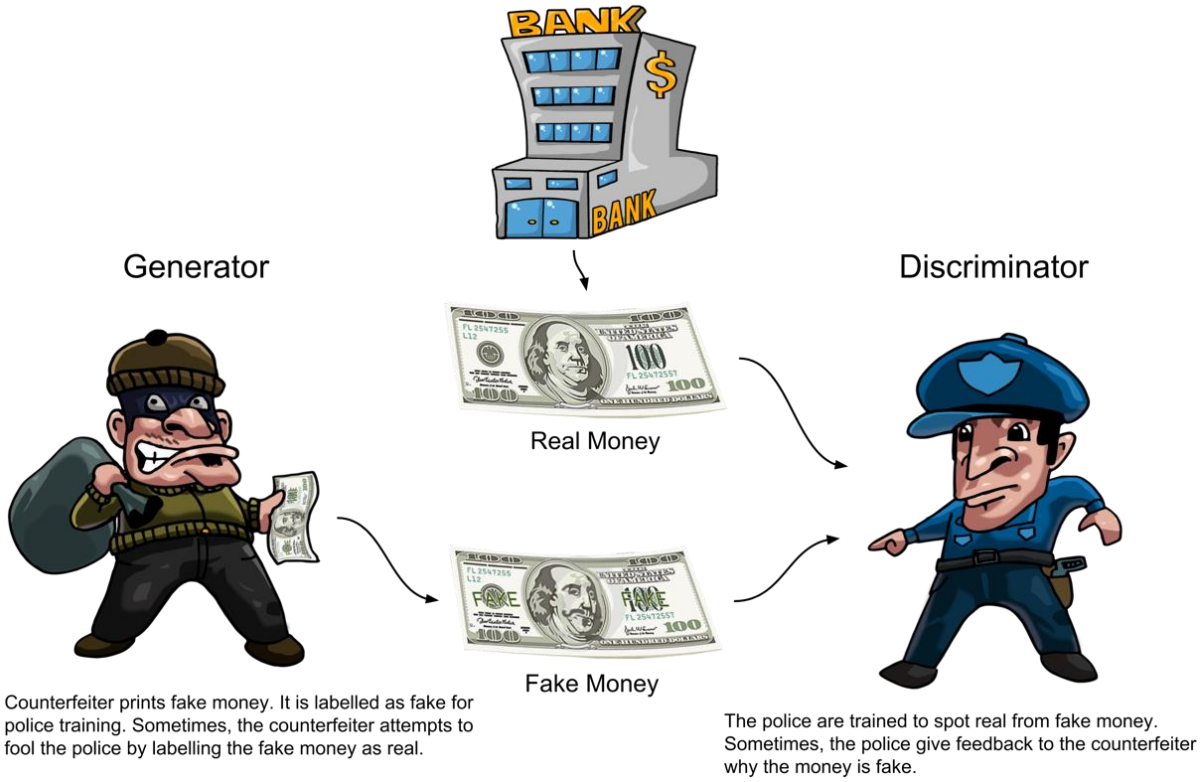
Bảng 2 Bảng phân công công việc

Công việc	Người thực hiện
Tìm hiểu khái quát đề tài	Nguyên
Làm file báo cáo	Cả nhóm
Lên lịch thực hiện	Cường
Tìm hiểu ứng dụng của Gans	Cường
Phân tích tìm hiểu lý thuyết toán	Nguyên
Chỉnh sửa báo cáo	Cả nhóm

2. CHƯƠNG 2: LÝ THUYẾT

2.1. Generative Adversarial Networks (GANs) là gì?

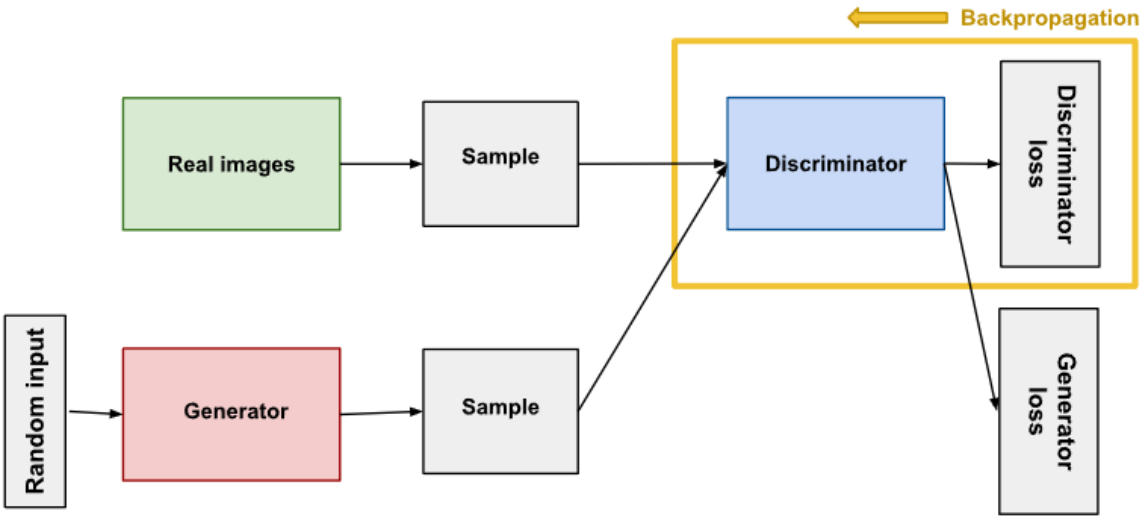
Generative Adversarial Networks Hướng tới việc sinh ra dữ liệu mới sau khi học như là tạo ra một khuôn mặt người, chữ viết, bản nhạc, hoặc những thứ khác như thế. Thế Generative Adversarial Networks là cái gì? Generative Adversarial Networks goi tắt là (GANs) - Mạng Chống đối Tạo sinh - hình thành trên ý tưởng về sự cạnh tranh của hai mạng neural network: - Discriminative network (mạng phân biệt) - Generative network (mạng sinh) Quan hệ giữa Discriminative network và Generative network?



Discriminative được ví như là một cảnh sát, còn Generative như là một tội phạm làm tiền giả. Tội phạm làm tiền giả thì sẽ học cách làm tiền giả sao cho cảnh sát không thể phân biệt được đó là tiền giả. Cảnh sát thì sẽ học cách phân biệt được đâu là tiền giả còn đâu là tiền thật, và báo cho tội phạm làm tiền giả biết là tiền giả của nó còn non lắm cần phải cải thiện thêm. Làm việc như trên như thế sẽ làm cho tội phạm làm tiền giả giống tiền thật hơn và cảnh sát cũng sẽ phân biệt được tiền giả tốt hơn.

2.2. Discriminative network

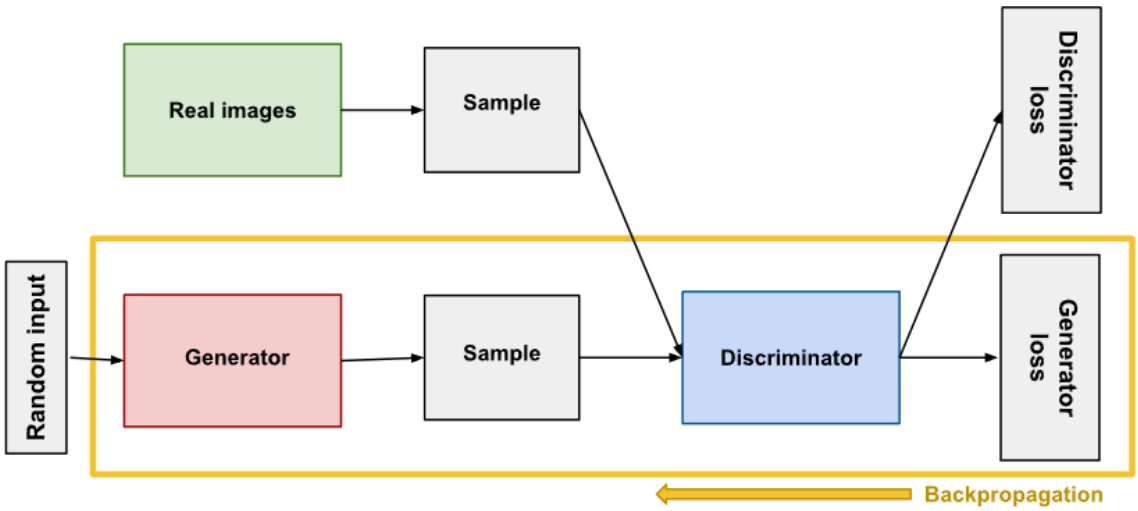
Discriminative đơn giản là một mô hình phân lớp (classifier) giống với bài toán Logistic Regression. Mô hình này sẽ học phân biệt đâu là data thực (real data) và đâu là data được tạo bởi Generator (fake data).



Discriminator được huấn luyện từ dữ liếu đến từ hai nguồn: - Real data: Dữ liệu thực tế được lập trình viên đưa vào. Discriminator sẽ xem đây là nhãn positive trong suốt quá trình huấn luyên (training). - Fake data: Dữ liệu được tạo bởi Generator. Discriminator sẽ xem đây là nhãn negative trong suốt quá trình huấn luyện (training).

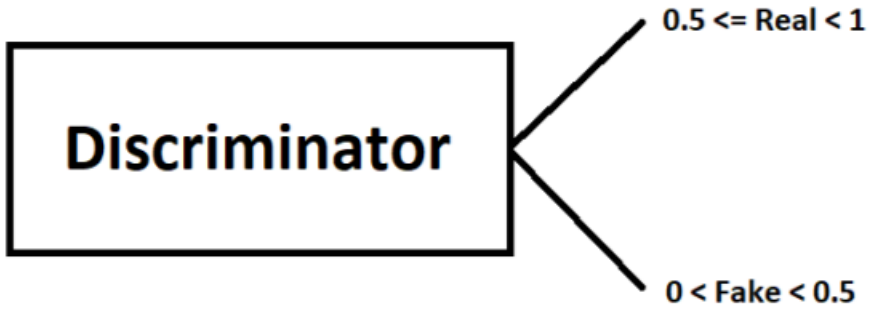
2.3. Generative network

Generator sẽ tạo dữ liệu từ phản hồi của Discriminator. Và từ đó học làm sao để cho Discriminator không phân biệt được dữ liệu của nó tạo ra là giả. Khi huấn luyện Generator yêu cầu tích hợp chặt chẽ với Discriminator. Phần đào tạo Generator: - Random input (random noise). - Generator từ random input tạo ra dữ liệu. - Discriminator sẽ phân loại dữ liệu được tạo. - Generator Loss



2.4. Loss Function

Cũng như thế GANs có hai loss function: một là cho Generator huấn luyện còn một là cho Discriminator huấn luyện. Làm rõ về hai Loss Function. Ta ký hiệu: z là đầu vào noise random, x là dữ liệu từ datasets. Ký hiệu mạng Generator là G, Discriminator là D và từ đó ta thì G(z) là dữ liệu sinh ra từ mạng Generator và D(x) là giá trị dự đoán của mạng Discriminator từ dữ liệu từ bộ datasets. Vậy D(G(z)) là giá trị dự đoán của mạng Discriminator về dữ liệu sinh ra từ Generator. Discriminator phân biệt thật giả giống như bài toán phân lớp Logistic Regression, vì thế nên Loss Function nó cũng dùng binary cross-entropy loss của bài sigmoid (binary cross-entropy dùng cho các bài toán binary classification).



Thế nên giá trị output của Discriminator nằm trong khoảng (0, 1). Discriminator sẽ train với dữ liệu trong datasets tiến tới gần bằng 1 nhất, với dữ liệu từ Generator thì sẽ gần bằng 0 nhất. Tức là ta muốn làm Loss Function của D(x) cao nhất có thể (maximize) và D(G(z)) thấp nhất có thể (minimize). Ta có minimize của

D(G(z)) tương đương với maximize (1 – D(G(z))), Loss function của Discriminator sẽ là:

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

Trong đó: $E_{x \sim p_{data}}[logD(x)]$ là thể hiện được khả năng nhận diện được data thật thốt hơ. còn $E_{z \sim p_z(z)}[log(1 - D(G(z)))]$ cho thấy khả năng nhận diện data từ Generator tốt hơn.

Còn về Generator thì chỉ có một E kỳ vọng để thể hiện khả năng lừa được Discrimnator tốt hơn.

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

Và ta gộp lại như sau.

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

2.5. Một vài kiến trúc GAN

2.5.1. CycleGAN

CycleGAN là một kiến trúc GAN rất phổ biến, chủ yếu được sử dụng để học chuyển đổi giữa các hình ảnh có kiểu dáng khác nhau.

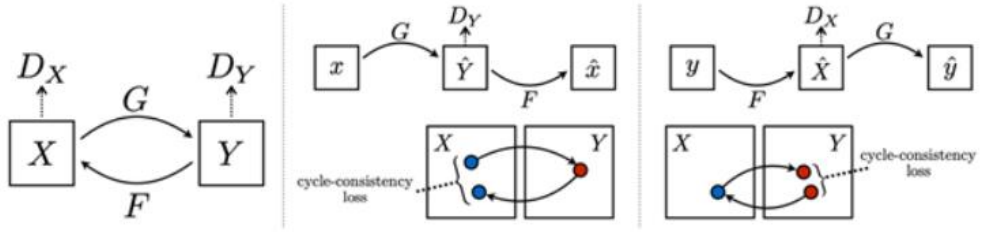
Ví dụ: chuyển đổi hình ảnh mùa đông và mùa hè, chuyển đổi hình ảnh giữa báo đen và báo đốm, chuyển đổi hình ảnh ngựa thường và ngựa vằn,…

FaceApp là một ứng dụng khá nổi tiếng xử dụng CycleGAN, nơi khuôn mặt chúng ta sẽ được biến đổi theo nhóm tuổi, độ tuổi khác nhau.

CycleGAN là một mở rộng của kiến trúc GAN cổ điển bao gồm 2 Generator và 2 Discriminator. Generator đầu tiên gọi là G, nhận đầu vào là ảnh từ domain X và convert nó sang domain Y . Generator còn lại gọi là Y, có nhiệm vụ convert ảnh từ domain Y sang X. Mỗi mạng Generator có 1 Discriminator tương ứng với nó

DYD_YDY: phân biệt ảnh lấy từ domain Y và ảnh được translate G(x).

DXD_XDX: phân biệt ảnh lấy từ domain X và ảnh được translate F(y).



Trong quá trình huấn luyện, generator G cố gắng tối thiểu hóa hàm adversarial loss bằng cách translate ra ảnh G(x) (với x là ảnh lấy từ domain X) sao cho giống với ảnh từ domain Y nhất, ngược lại Discriminator DYD_YDY cố gắng cực đại hàm adversarial loss bằng cách phân biệt ảnh G(x) và ảnh thật y từ domain

2.5.2. DiscoGAN

DiscoGAN mới trở nên phổ biến trong thời gian gần đây nhờ vào khả năng học các mối liên hệ giữa các tên miền với dữ liệu không giám sát.

Nếu con người chúng ta nhìn vào hình trên dễ dàng phát hiện mối liên hệ giữa 2 miền là về bản chất của màu sắc. Tuy nhiên để máy có thể tìm ra mối liên hệ giữa 2 miền chưa được ghép nối là vô cùng khó khăn.

DiscoGAN và CycleGAN khá giống nhau về mặt thiết kế mạng. Một bên xử dụng phép chuyển đổi từ miền X sang miền Y, còn một bên thì học theo một ánh xạ ngược lại. Cả hai đều sử dụng reconstruction loss như một phép đo mức độ tái tạo sau hai lần chuyển đổi giữa các miền.

3. CHƯƠNG 3: THỰC TIỄN

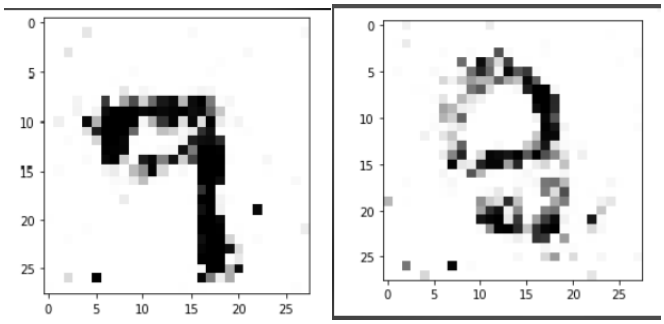
Dùng data MNIT HandWritten Digit. Train model tạo hình chữ số viết tay. Sẽ resize ảnh về (28, 28, 1) Build mạng Generator:

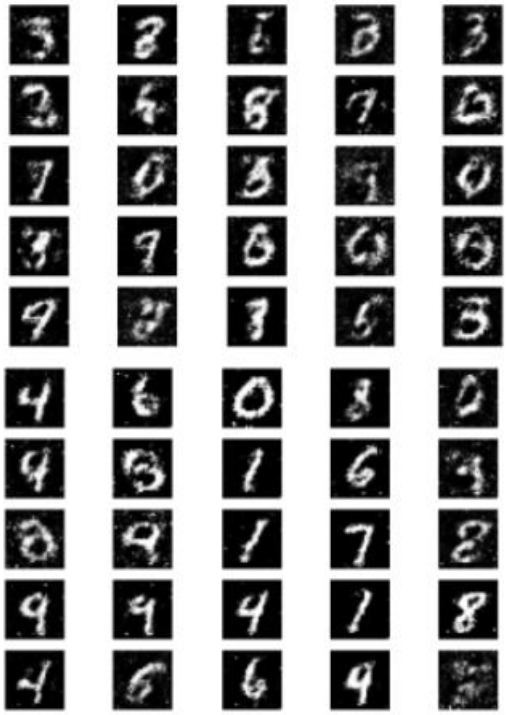
```
def build_generator():

    noise_shape = (100,)

    model = Sequential()

    model.add(Dense(256, input_shape=noise_shape))
    model.add(LeakyReLU(alpha=0.2))
    model.add(BatchNormalization(momentum=0.8))
    model.add(Dense(512))
    model.add(LeakyReLU(alpha=0.2))
    model.add(BatchNormalization(momentum=0.8))
    model.add(Dense(1024))
    model.add(LeakyReLU(alpha=0.2))
    model.add(BatchNormalization(momentum=0.8))

    model.add(Dense(np.prod(img_shape), activation='tanh'))
    model.add(Reshape(img_shape))

    model.summary()

    noise = Input(shape=noise_shape)
    img = model(noise)

    return Model(noise, img)
```

Bulid mạng Discrimnator

```
def build_discriminator():
    model = Sequential()

    model.add(Flatten(input_shape=img_shape))
    model.add(Dense(512))
    model.add(LeakyReLU(alpha=0.2))
    model.add(Dense(256))
    model.add(LeakyReLU(alpha=0.2))
    model.add(Dense(1, activation='sigmoid'))
    model.summary()

    img = Input(shape=img_shape)
    validity = model(img)

    return Model(img, validity)
```

Một vài hình ảnh các dữ liệu được tạo ra bởi Generator trong quá trình train.





4. CHƯƠNG 5: KẾT LUẬN

4.1. Kết luận

Tiềm hiểu rõ về Generative Adversarial Networks(GANs) được cấu thành từ 2 mạng Generator và Discriminator.

GANs giải quyết vấn đề về các hoạt động huấn luyện mà không cần can thiệp nhiều đến từ còn người trong việc huấn luyện cụ thể rõ như gán nhãn cho dữ liệu.

Hiểu vai trò của từng mạng Generator và Discriminator trong GANs. Hai mạng này được xem giống như là đối trọng của nhau, và cũng như cùng nhau phát triển. Discriminator báo Generator dữ liệu làm giả của nó không được tốt, Discriminator thì sẽ học được khả năng phân biệt dữ liệu thật và giả nhờ dữ liệu được cung cấp từ Generator.

Hiểu được Ưu và nhược điểm của GANs.

Tìm hiểu sơ hàm Loss Function của model GANs

Biết được một vài kiến trúc GANs phổ biến

4.1.1. Ưu điểm

GAN tạo ra dữ liệu giống với dữ liệu gốc. Nếu cung cấp cho GAN một hình ảnh thì nó sẽ tạo ra một phiên bản mới của hình ảnh trông giống với hình ảnh gốc. Tương tự, nó có thể tạo ra các phiên bản khác nhau của văn bản, video, âm thanh.

GAN đi sâu vào chi tiết dữ liệu và có thể dễ dàng diễn giải thành các phiên bản khác nhau, vì vậy nó rất hữu ích trong việc thực hiện công việc học máy.

Bằng cách sử dụng GAN và máy học, có thể dễ dàng nhận ra cây cối, đường phố, người đi xe đạp, người và ô tô đang đỗ và cũng có thể tính toán khoảng cách giữa các đối tượng khác nhau.

4.1.2. Nhược điểm

Khó đào tạo hơn: Cần cung cấp liên tục các loại dữ liệu khác nhau để kiểm tra xem nó có hoạt động chính xác hay không.

Việc tạo kết quả từ văn bản hoặc giọng nói rất phức tạp

TÀI LIỆU THAM KHẢO

Sách tham khảo:

1.GANs in Action: Deep Learning with Generative Adversarial Networks

Tài liệu tham khảo:

2. https://nttuan8.com/

3. https://www.researchgate.net/publication/334484229_OnkoGan_Bangla_Handwritten_Digit_Generation_with_Deep_Convolutional_Generative_Adversarial_Networks