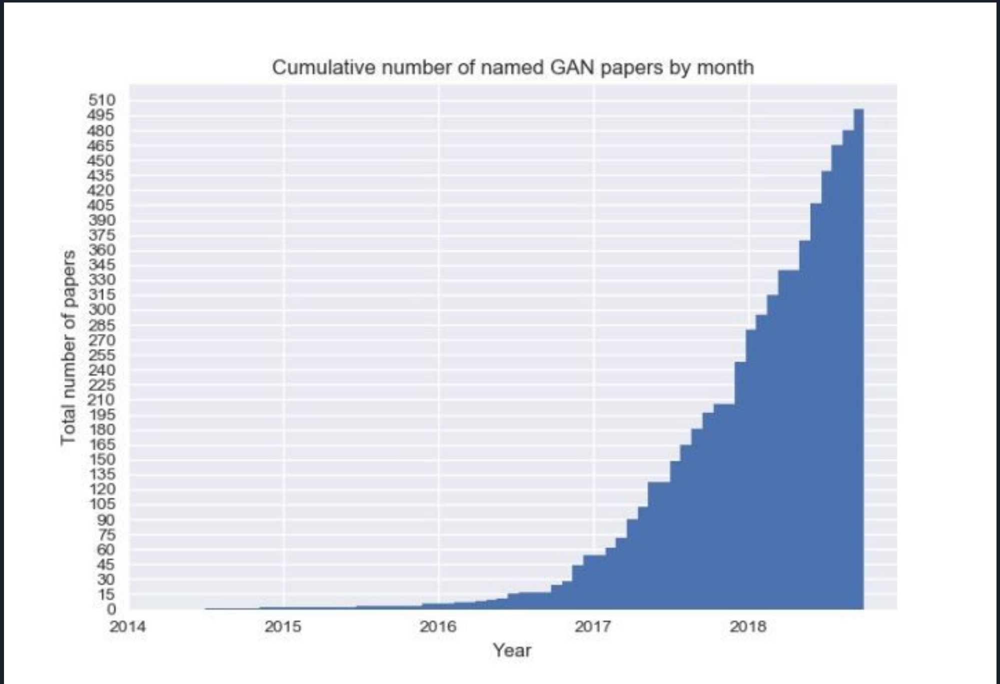
Generative Adversarial Networks

Supervisor: Abdul QAYYUM

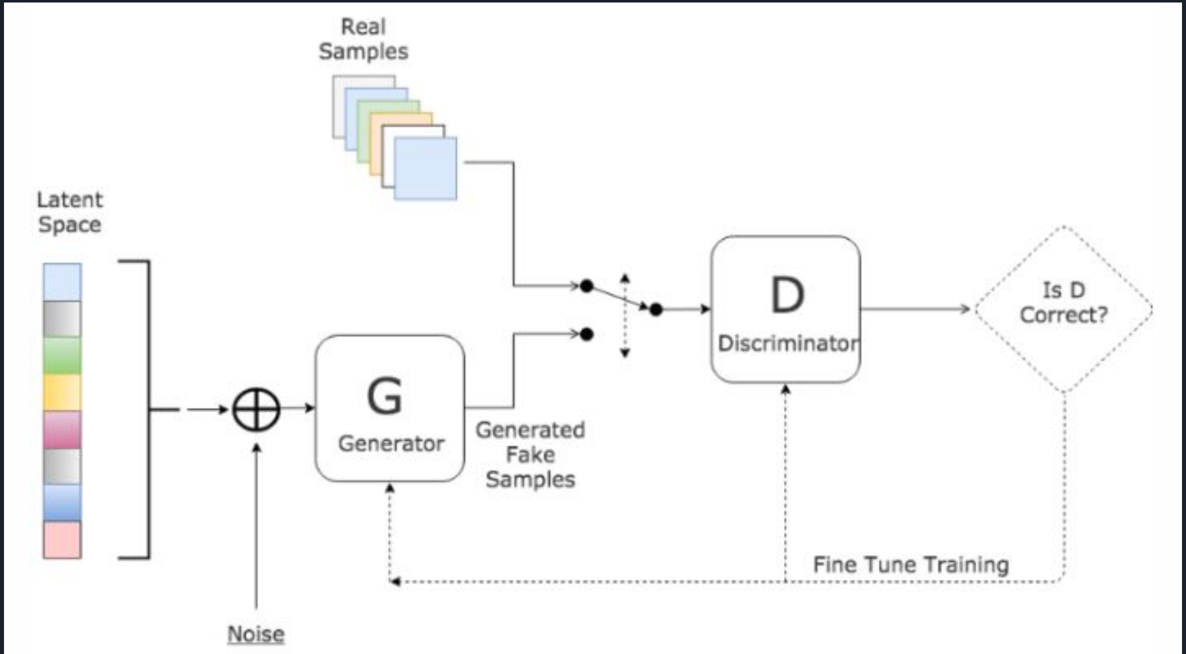
Presenters: Cheng CHEN, Muhammad Arsalan KHAWAJA, Mahmoud Badran.

Contents

Introduction



Introduction : basic principles

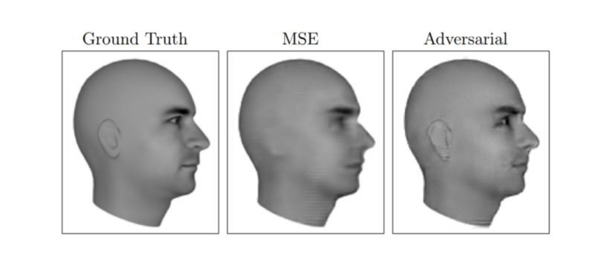


Introduction : types

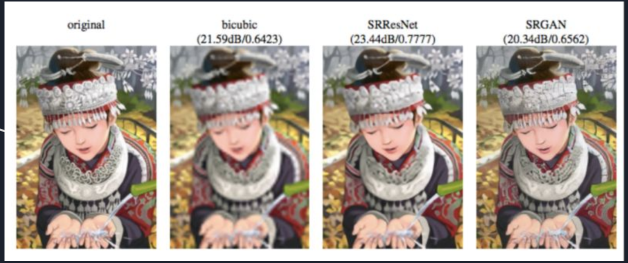
- ❏ Deep Convolutional GANs (DCGANs)
- ❏ Conditional GANs (cGANs)
- ❏ StackGAN
- ❏ InfoGANs
- ❏ Wasserstein GANs(WGAN)
- ❏ Disco GANS
- ❏ etc...

Introduction : applications

- ➢ Predicting the next frame in a video :



- ➢ Increasing Resolution of an image :



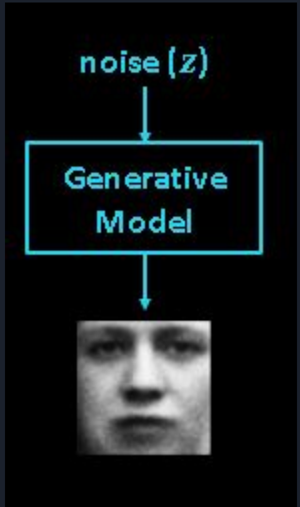
- ➢ Text-to-Image Generation

- ➢ Image to Image Translation :

Lets Explore GAN’s

Generative Model

- How to make it generate different samples each time it is run?
  - input to model is noise
- Generative model as a neural network
  - computes $x = G(z|\theta)$
  - differentiable
  - does not have to be invertible
  - z typically has very high dimensionality (higher than x)

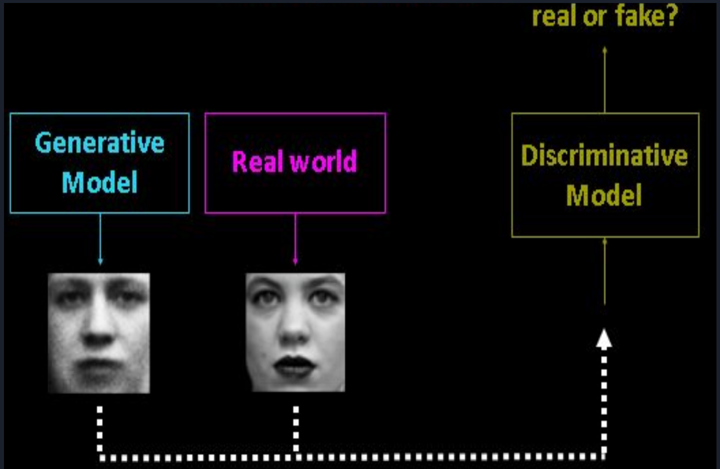


Discriminative Model

- Think of it as a critic
    - A good critic can tell real from fake
- Discriminative model is a neural net
- It is differentiable
- It computes $D(x)$, with value 1 if real, 0 if fake

Training Methodology: Basic Concept

- G tries to fool D
- D tries not to be fooled
- Models are trained simultaneously
- As G gets better, D has a more challenging task
- As D gets better, G has a more challenging task
- Ultimately, we don't care about the D
- Its role is to force G to work harder



Loss Function

- GAN can have **two** loss functions: one for generator training and one for discriminator training.
- In the paper that introduced GANs, the generator tries to minimize the function while the discriminator tries to maximize it, thats why the loss function is called **Minimax**

Loss Function for Discriminator

- Loss function for D
  - maximize the likelihood that model says ‘real’ to samples from the world and ‘fake’ to generated samples

$$\mathcal{L}_D = -\frac{1}{2}\mathbb{E}_{x\sim\text{world}} \ln D(x) - \frac{1}{2}\mathbb{E}_z \ln\left(1 - D(G(z))\right)$$

Loss Function for Generator

- What should the loss function be for G?
- $\mathcal{L}_G = -\mathcal{L}_D$
- But because first term doesn't matter for G

$$\mathcal{L}_D = \frac{1}{2}\mathbb{E}_z \ln\left(1 - D(G(z))\right)$$

Training

Because a GAN contains two separately trained networks, its training algorithm must address two complications:

- GANs must juggle two different kinds of training (generator and discriminator).
- GAN convergence is hard to identify.

Alternating Training

GAN training proceeds in alternating periods:

- The discriminator trains for one or more epochs.
- The generator trains for one or more epochs.
- Repeat steps 1 and 2 to continue to train the generator and discriminator networks.

Demonstration / Implementation of GAN

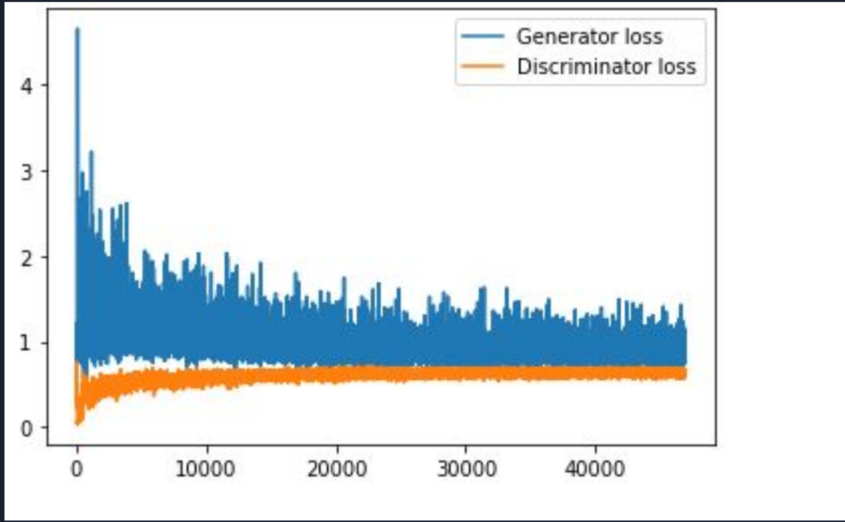
G Architecture

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 6272)              633472
_________________________________________________________________
leaky_re_lu (LeakyReLU)      (None, 6272)              0
_________________________________________________________________
reshape (Reshape)            (None, 7, 7, 128)         0
_________________________________________________________________
conv2d_transpose (Conv2DTran (None, 14, 14, 128)       262272
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 14, 14, 128)       0
_________________________________________________________________
conv2d_transpose_1 (Conv2DTr (None, 28, 28, 128)       262272
_________________________________________________________________
leaky_re_lu_2 (LeakyReLU)    (None, 28, 28, 128)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 28, 28, 1)         6273
=================================================================
Total params: 1,164,289
Trainable params: 1,164,289
Non-trainable params: 0
```

D Architecture

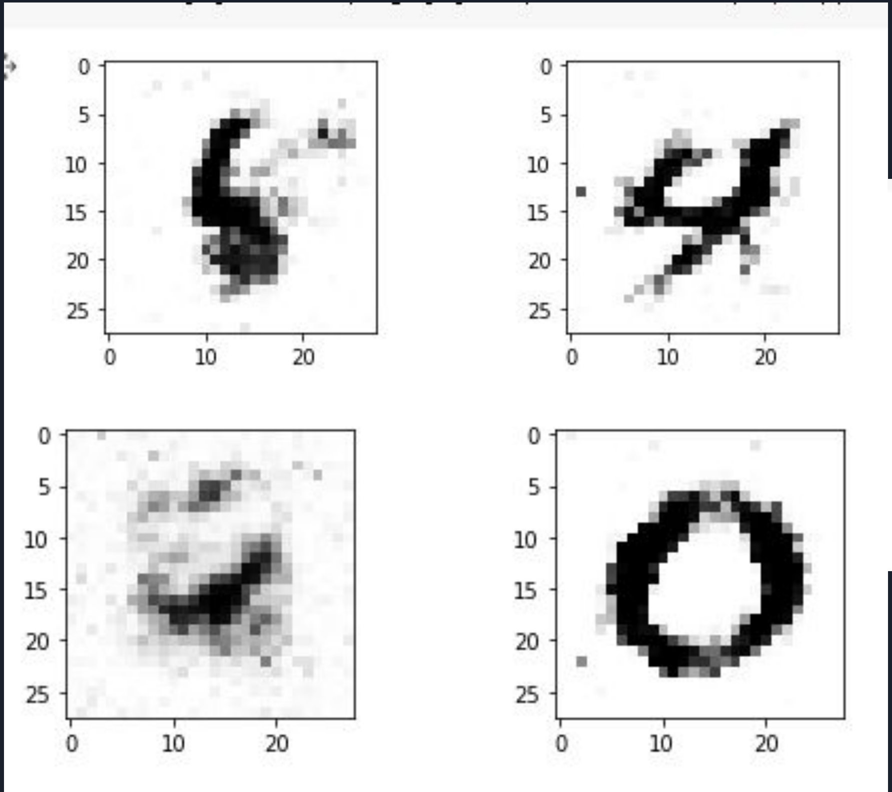
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 14, 14, 64)        640
_________________________________________________________________
leaky_re_lu (LeakyReLU)      (None, 14, 14, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 7, 7, 64)          36928
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 7, 7, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 3136)              0
_________________________________________________________________
dense (Dense)                (None, 1)                 3137
=================================================================
Total params: 40,705
Trainable params: 40,705
Non-trainable params: 0
```

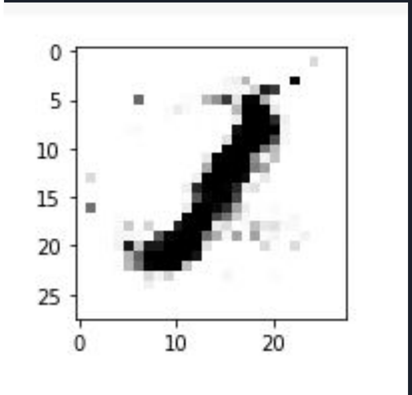
Loss



Generator loss
Discriminator loss
4
3
2
1
0
0
10000
20000
30000
40000

Generated images (MNIST)





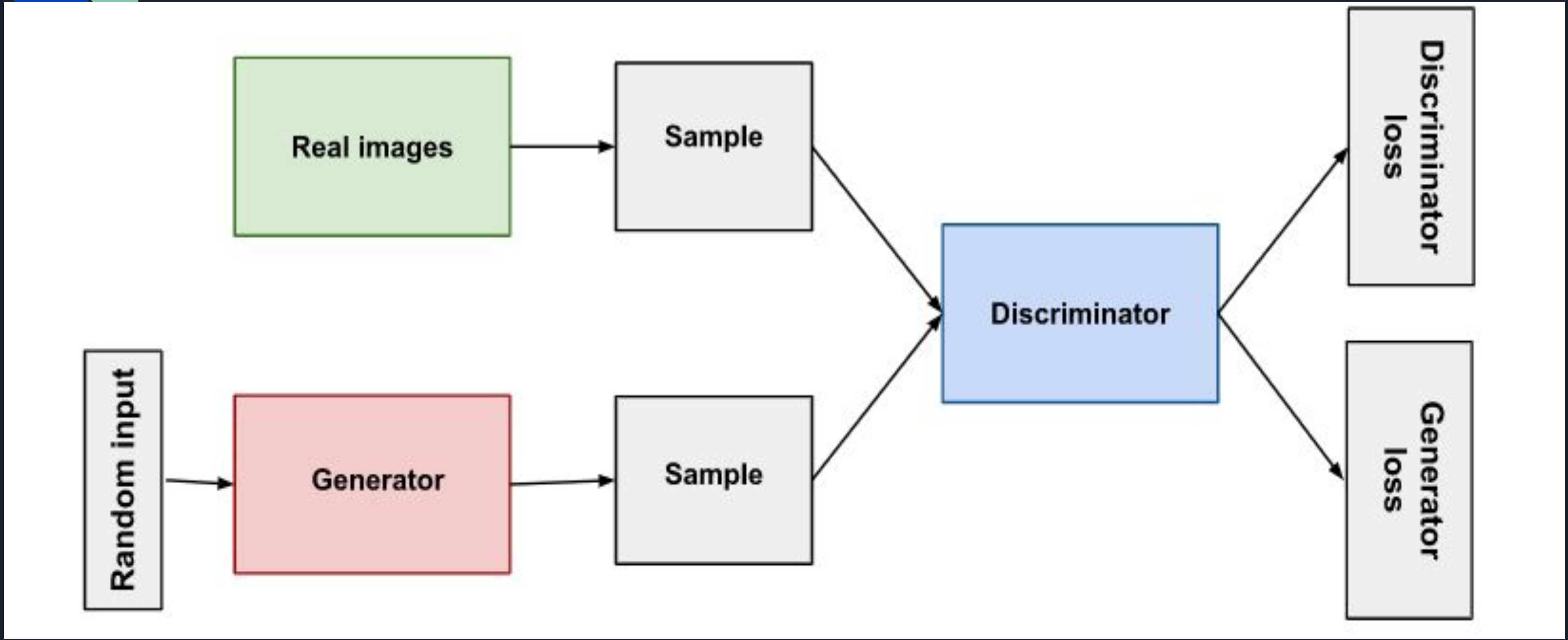
Conclusion

- ❑ Mode collapse: the generator produces limited varieties of samples,
- ❑ Diminished gradient: the discriminator gets too successful that the gradients vanish and the generator learns nothing,
- ❑ Non-convergence: the model parameters oscillate, destabilize and never converge,

Conclusion

- Unbalance between the generator and discriminator causes overfitting.
- Highly sensitive to hyperparameters.

Executive Summary of GAN’s



Thank you~