



Introduction to Deep Learning

Amrita Vishwa Vidyapeetham
Amritapuri Campus





Generative Adversarial Network (GAN)

Courtesy: Mitesh Kapra NPTEL, analytics vidhya, fast.ai, coursera: AndrewNG

- Train an artificial author which can write an article and explain data science concepts to a community in a very simplistic manner by learning from past articles on Analytics Vidhya

- You are not able to buy a painting from a famous painter which might be too expensive. Can you create an artificial painter which can paint like any famous artist by learning from his / her past collections?





(GANs), and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion.” – Yann Lecun

106.2661v1 [stat.ML] 10 Jun 2014

Generative Adversarial Nets

Ian J. Goodfellow, Jean Pouget-Abadie*, Mehdi Mirza, Bing Xu, David Warde-Farley,
Sherjil Ozair†, Aaron Courville, Yoshua Bengio‡

Département d’informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

Abstract

We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G . The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D , a unique solution exists, with G recovering the training data distribution and D equal to $\frac{1}{2}$ everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples. Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of the generated samples.

1 Introduction

The promise of deep learning is to discover rich, hierarchical models [2] that represent probability distributions over the kinds of data encountered in artificial intelligence applications, such as natural

Which face is real?



A



B



C



- <https://thispersondoesnotexist.com/>

Deepfakes?-Face synthesis with GANs and Autoencoders

Deepfakes are synthetic media in which **a person in an existing image or video is replaced with someone else's likeness**

facial manipulation using

- 1.Face synthesis (StyleGAN)**
- 2.Face swap**
- 3.Facial attributes and expression**



These people are not real – they were produced by StyleGAN's generator that allows control over different aspects of the image.

Generating realistic-looking images that even a human can't recognize whether it's real or not

Closest Real Images



Supervised vs unsupervised learning

Supervised Learning

Data: (x, y)

x is data, y is label

Goal: Learn function to map
 $x \rightarrow y$

Examples: Classification, regression, object detection, semantic segmentation, etc.

Unsupervised Learning

Data: x

x is data, no labels!

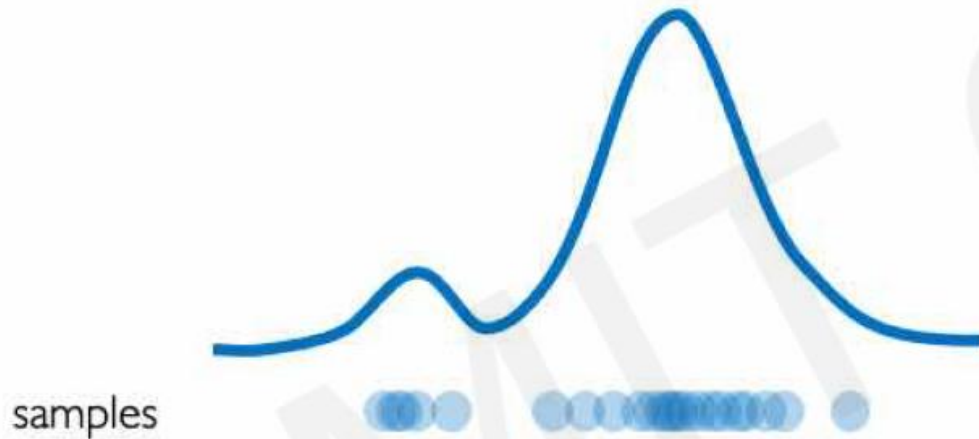
Goal: Learn the *hidden* or *underlying structure* of the data

Examples: Clustering, feature or dimensionality reduction, etc.

Generative modeling

Goal: Take as input training samples from some distribution and learn a model that represents that distribution

Density Estimation



Sample Generation



Input samples

Training data $\sim P_{data}(x)$



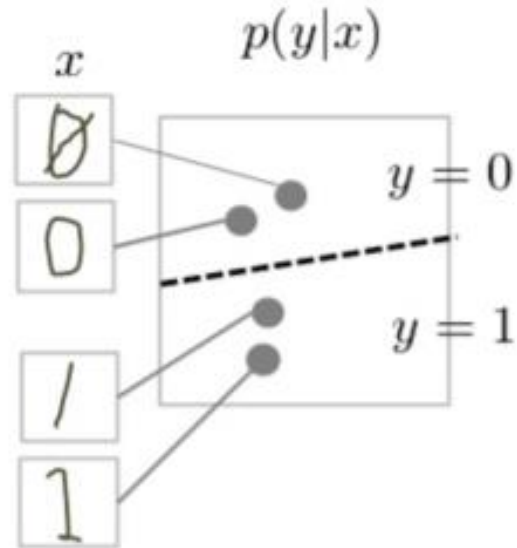
Generated samples

Generated $\sim P_{model}(x)$

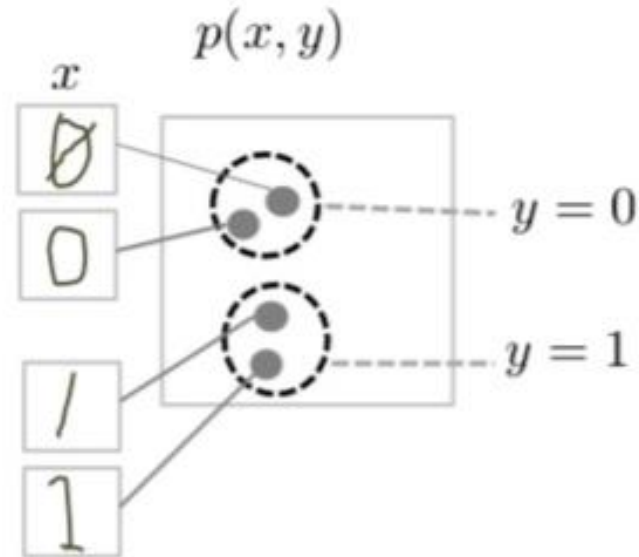
How can we learn $P_{model}(x)$ similar to $P_{data}(x)$?

GANs are just one kind of generative model. More formally, given a set of data instances X and a set of labels Y : Generative models **capture the joint probability $p(X, Y)$, or just $p(X)$ if there are no labels**. Discriminative models capture the conditional probability $p(Y | X)$.

- Discriminative Model



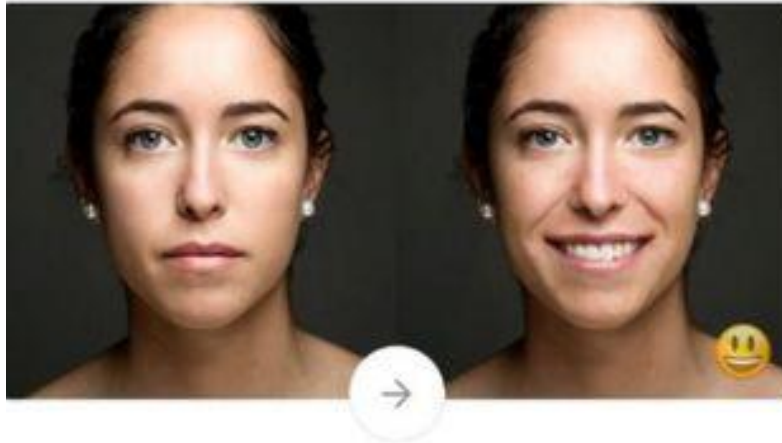
- Generative Model



FaceApp

FaceApp. The viral Russian app, FaceApp which automatically generates highly realistic transformations of your face

Make them smile



Meet your future self



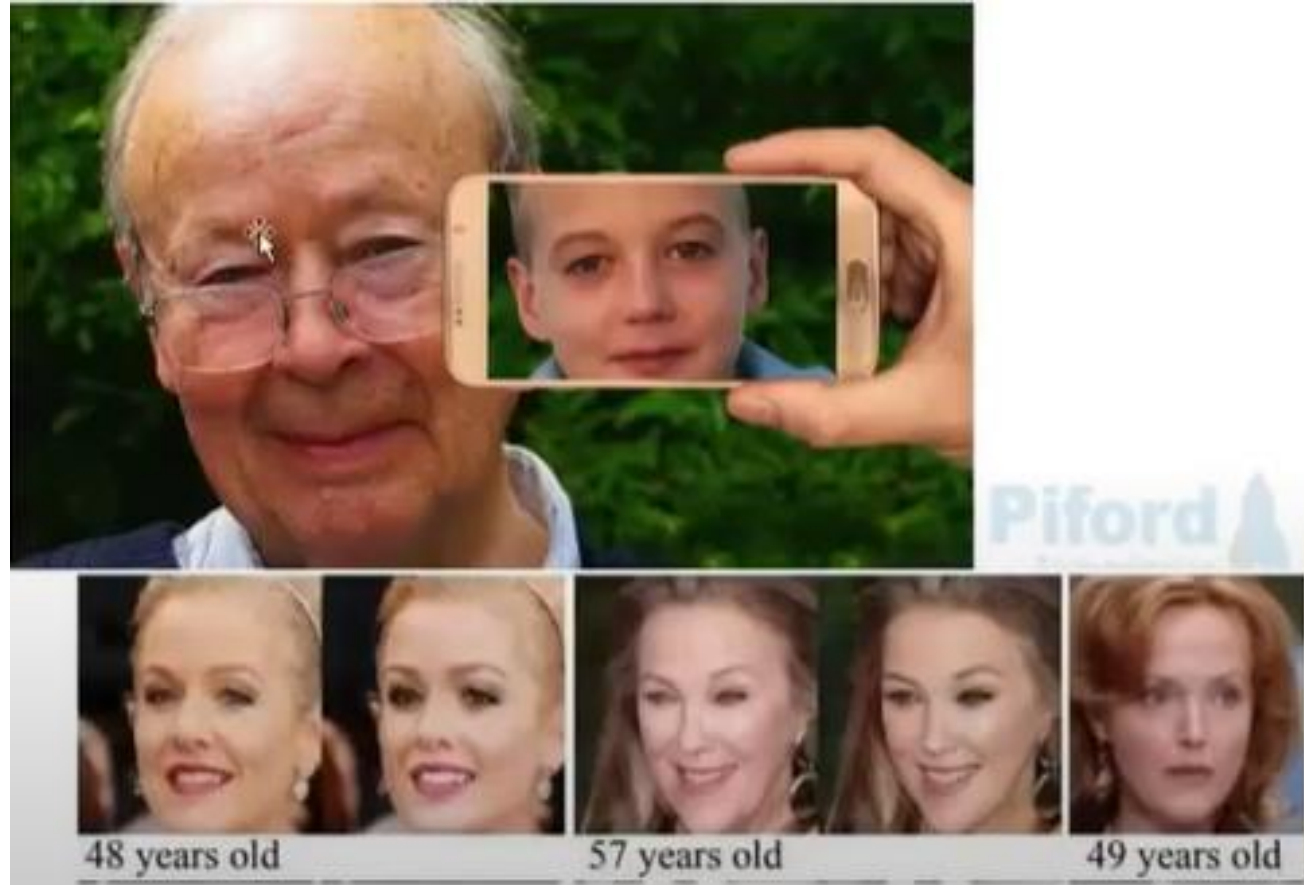
Look younger



Change your style



Face de-aging



Text to Image Generation

This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face



Text to Image Translation

StackGAN: Text to Photo-realistic Image Synthesis

This flower has yellow petals along with green and yellow stamen



This flower is red and yellow in color, with petals that are ruffled and curled



This flower is white and pink in color, with petals that are oval shaped



A yellow flower with large petal with a large long pollen tubes



The images on the top of each block are real images corresponding to the text description.

Photos to Emojis

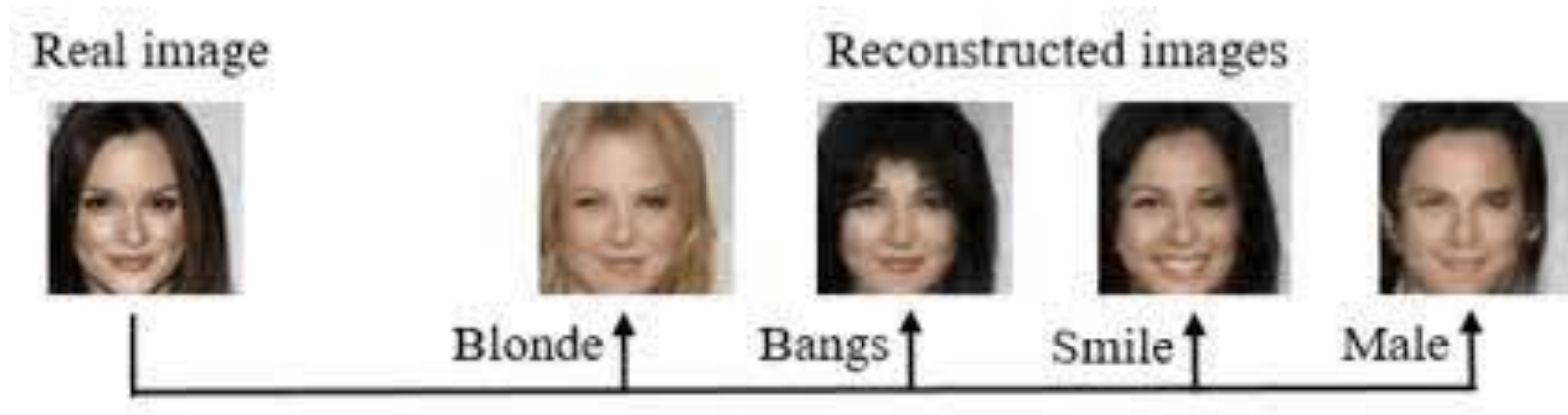
XGAN tremendously successful in generating fun cartoon avatar directly from photos



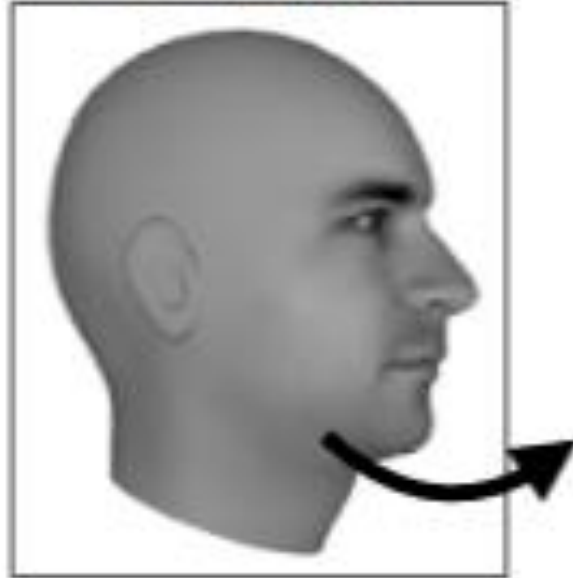
Image Editing

invertible Conditional GANs for image editing

[Guim Perarnau](#), [Joost van de Weijer](#), [Bogdan Raducanu](#), [Jose M. Álvarez](#)



Ground Truth



What happens next?

Predicting the next frame in a video : You train a GAN on video sequences and let it predict what would occur next

Increasing Resolution of an image : Generate a high resolution photo from a comparatively low resolution.

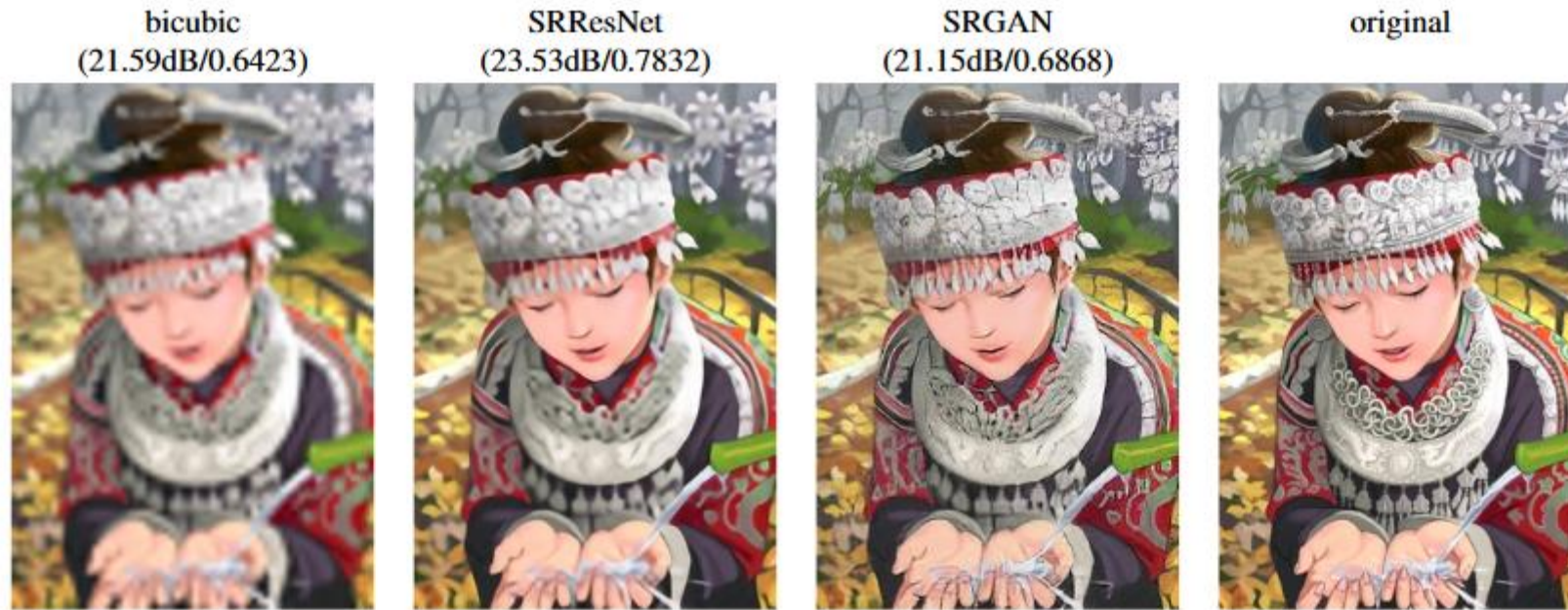


Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]

Interactive Image Generation : Draw simple strokes and let the GAN draw an impressive picture for you!



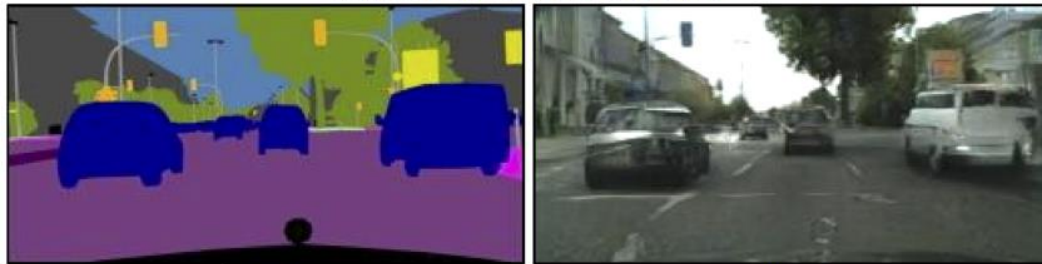
Image to Image Translation

: Generate an image from another image. For example, given on the left, you have labels of a street scene and you can generate a real looking photo with GAN. On the right, you give a simple drawing of a handbag and you get a real looking drawing of a handbag.



Pix2Pix GAN Image-to-image translation with conditional adversarial networks

Labels to Street Scene



input

output

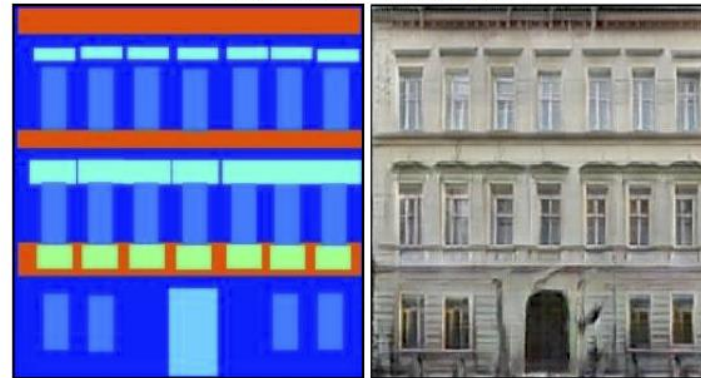
Aerial to Map



input

output

Labels to Facade



input

output

BW to Color



input

output

Day to Night



input

output

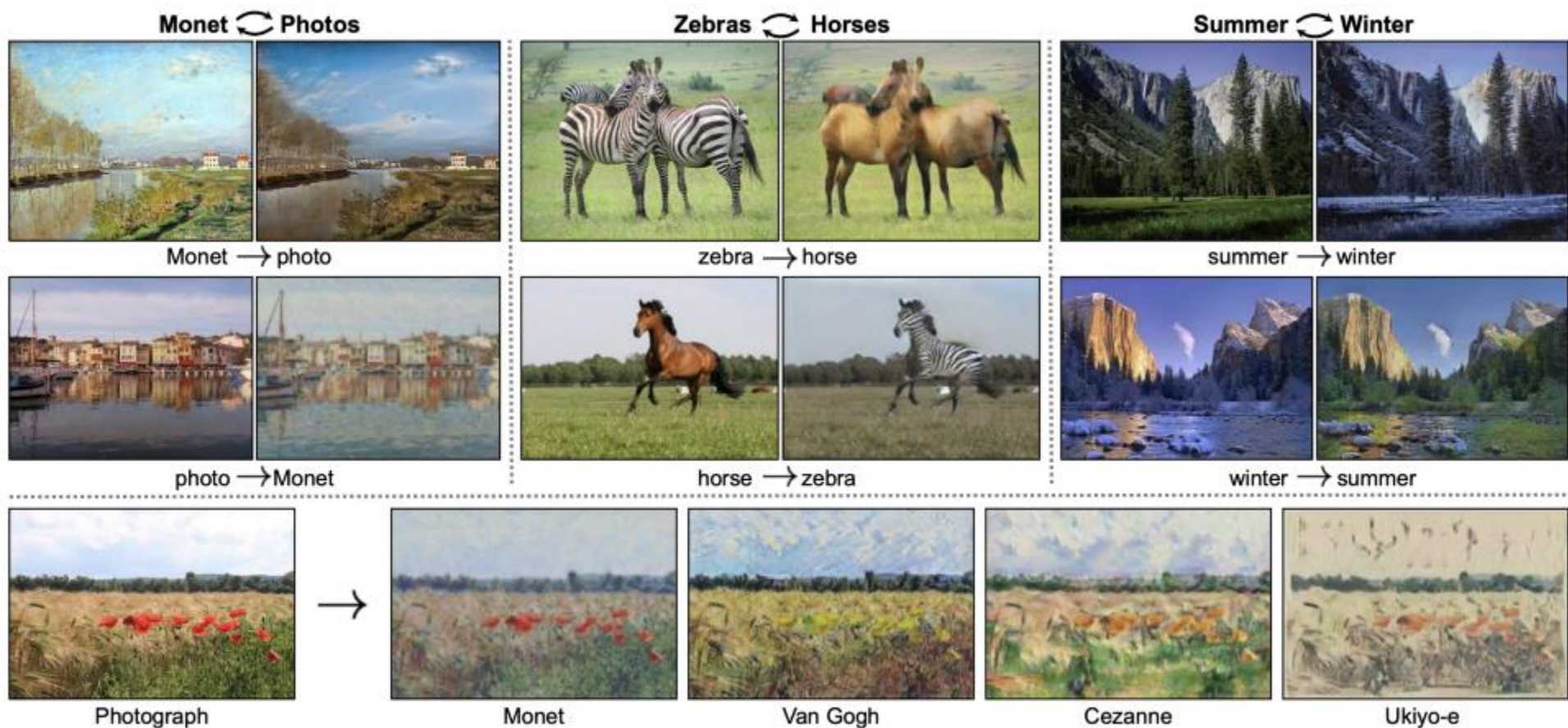
Edges to Photo



input

output

Image to image Translation (Cycle GAN)



Creative GANs for generation, of text, music poerty, lyrics, metaphors

MuseGAN: Using GANs to generate original Music



Creative GANs for generating poems, lyrics, and metaphors

Asir Saeed
MLT Labs
asir@mltokyo.ai

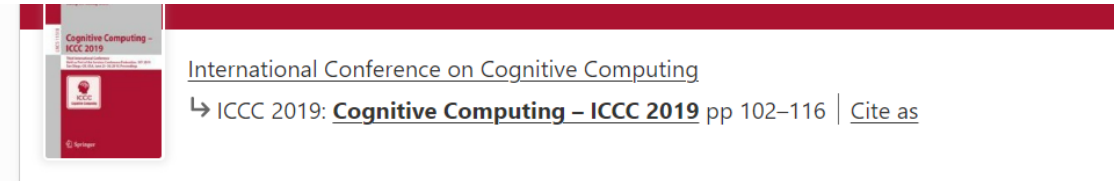
Suzana Ilić
University of Innsbruck, MLT Labs
suzana@mltokyo.ai

Eva Zangerle
University of Innsbruck
eva.zangerle@uibk.ac.at

Abstract

Generative models for text have substantially contributed to tasks like machine translation and language modeling, using maximum likelihood optimization (MLE). However, for creative text generation, where multiple outputs are possible and originality and uniqueness are encouraged, MLE falls short. Methods optimized for MLE lead to outputs that can be generic, repetitive and incoherent. In this work, we use a Generative Adversarial Network framework to alleviate this problem. We evaluate our framework on poetry, lyrics and metaphor datasets, each with widely different characteristics, and report better performance of our objective function over other generative models.


1 Introduction and related work



International Conference on Cognitive Computing

→ ICC 2019: **Cognitive Computing – ICC 2019** pp 102–116 | [Cite as](#)

Using GAN to Generate Sport News from Live Game Stats

[Changliang Li](#) , [Yixin Su](#), [Ji Qi](#) & [Min Xiao](#)

Conference paper | [First Online: 19 June 2019](#)

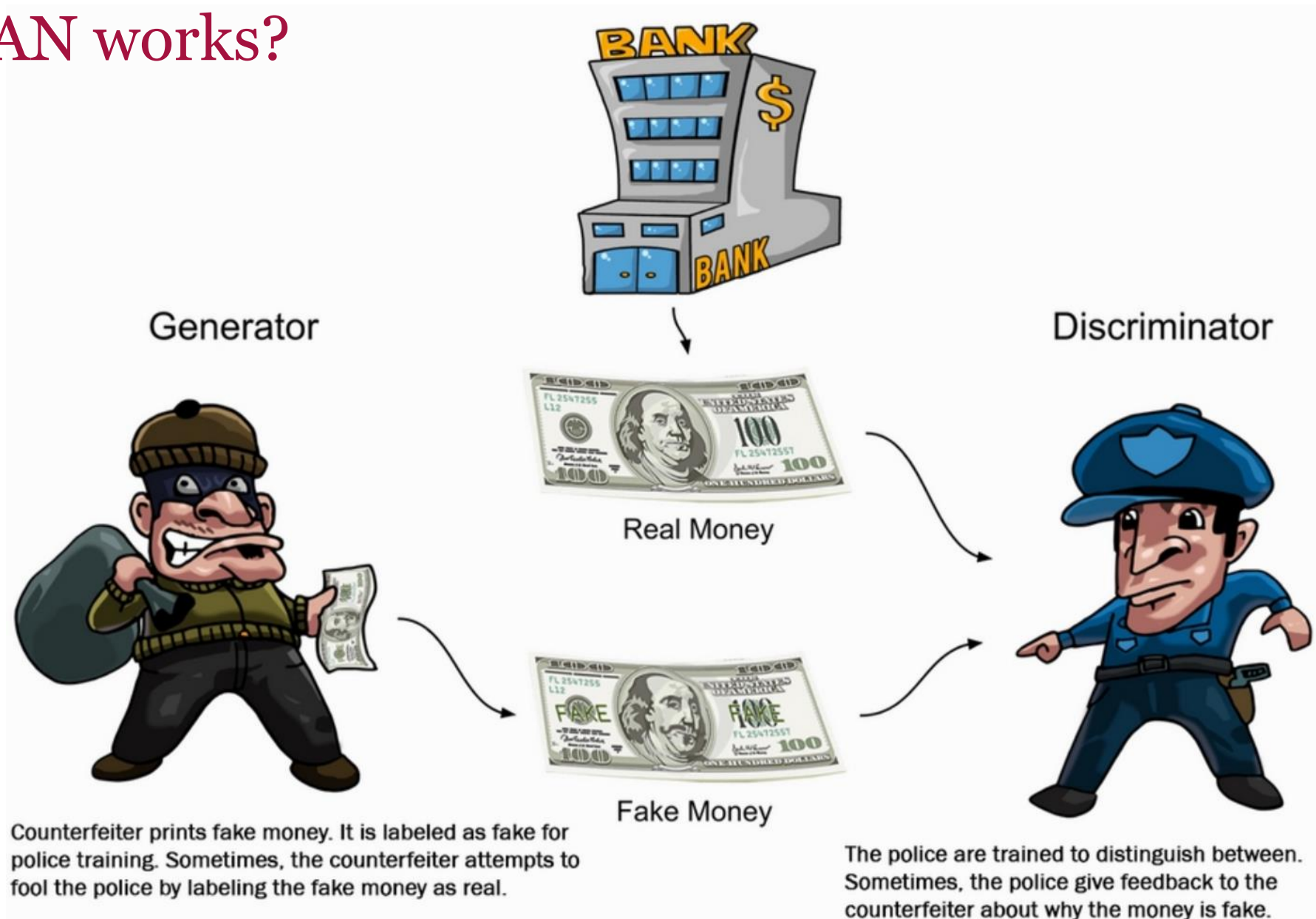
938 Accesses | 1 Citations

Part of the [Lecture Notes in Computer Science](#) book series (LNISA, volume 11518)

Abstract

One goal in artificial intelligence field is to create well-formed and human-like natural language text given data input and a specific goal. Some data-to-text solutions have been proposed and successfully used in real applied domains. Our work focuses on a new domain, Automatic Sport News Generating, which aims to produce sport news immediately after each match is over so that both time and labor can be saved on writing the news articles. We

How GAN works?



Counterfeiter prints fake money. It is labeled as fake for police training. Sometimes, the counterfeiter attempts to fool the police by labeling the fake money as real.

The police are trained to distinguish between. Sometimes, the police give feedback to the counterfeiter about why the money is fake.

Generative Adversarial Networks(GAN in short) is an advancement in the field of Machine Learning which is capable of generating new data samples including Text, Audio, Images, Videos, etc. using previously available data.

How does GAN Work?

GANs consists of two ANN or CNN models:

1. Generator Model: Used to generate new images which look like real images.
2. Discriminator Model: Used to classify images as real or fake.

Simply, for getting a powerful hero (generator), we need a more powerful opponent (discriminator)!

The Generator Model

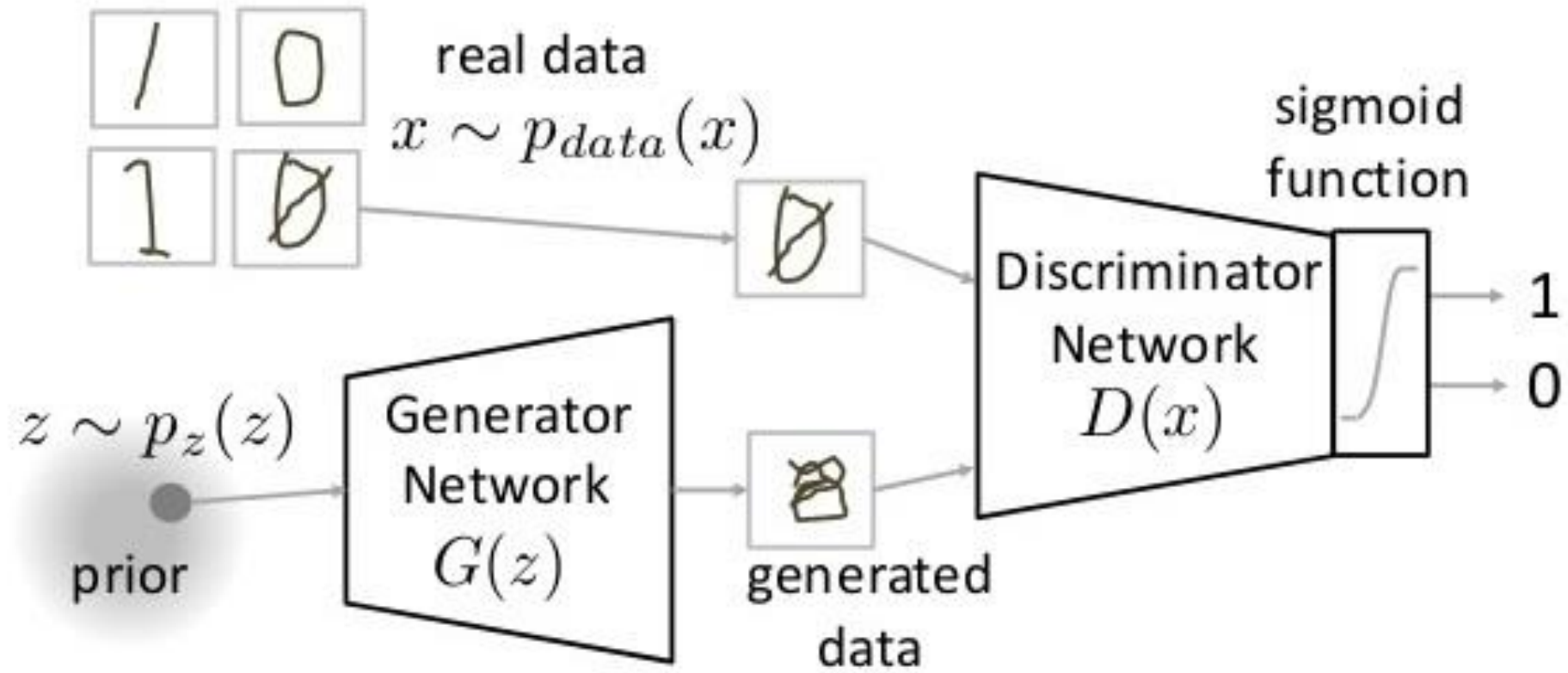
The Generator Model generates new images by taking a fixed size random noise as an input. Generated images are then fed to the Discriminator Model.

The main goal of the Generator is to fool the Discriminator by generating images that look like real images and thus makes it harder for the Discriminator to classify images as real or fake.

The Discriminator Model

The Discriminator Model takes an image as an input (generated and real) and classifies it as real or fake. Generated images come from the Generator and the real images come from the training data.

The discriminator model is the simple binary classification model.



The Generator Model G takes a random input vector z as an input and generates the images $G(z)$.

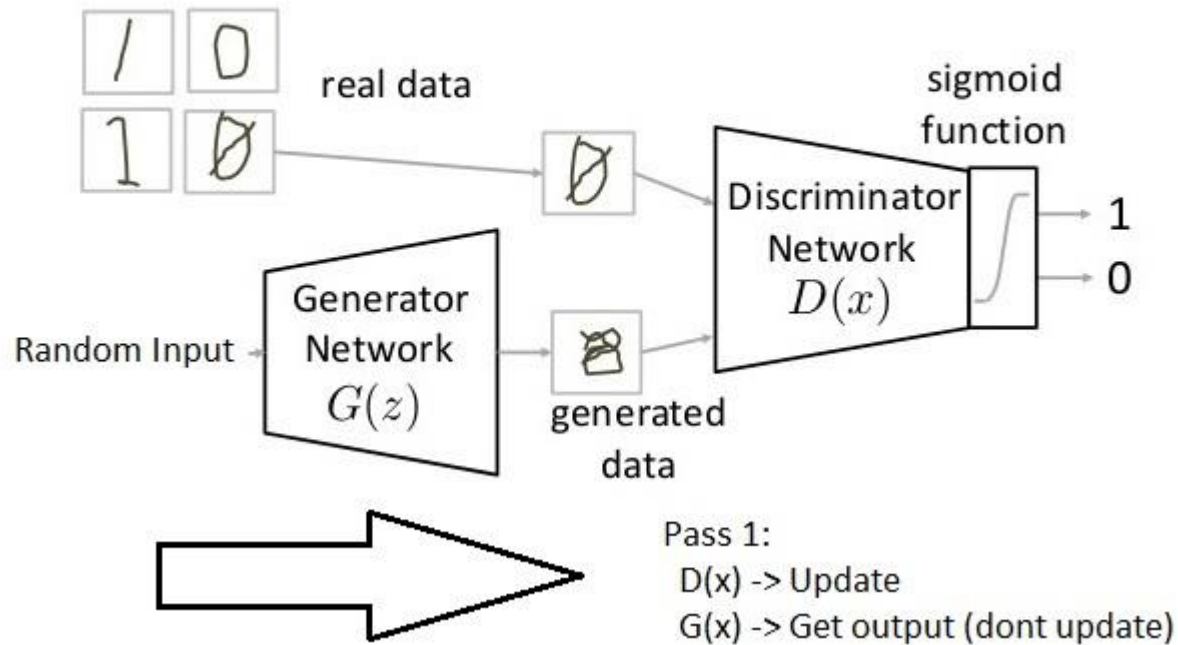
These generated images along with the real images x from training data are then fed to the Discriminator Model D .

The Discriminator Model then classifies the images as real or fake.

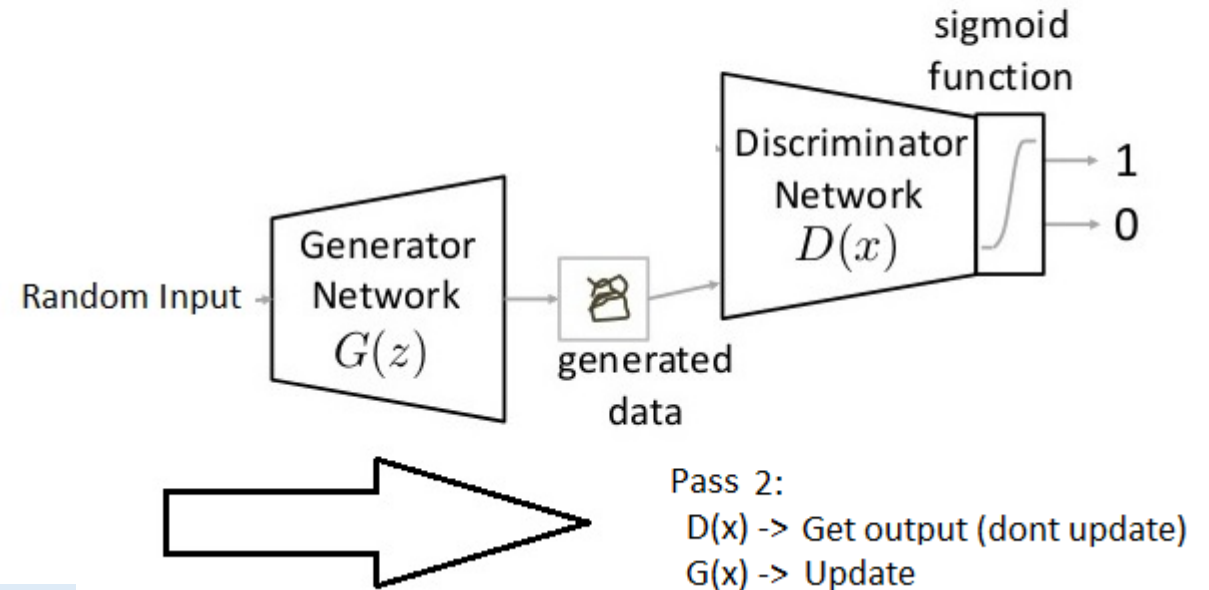
Then, we have to measure the loss and this loss has to be back propagated to update the weights of the Generator and the Discriminator.

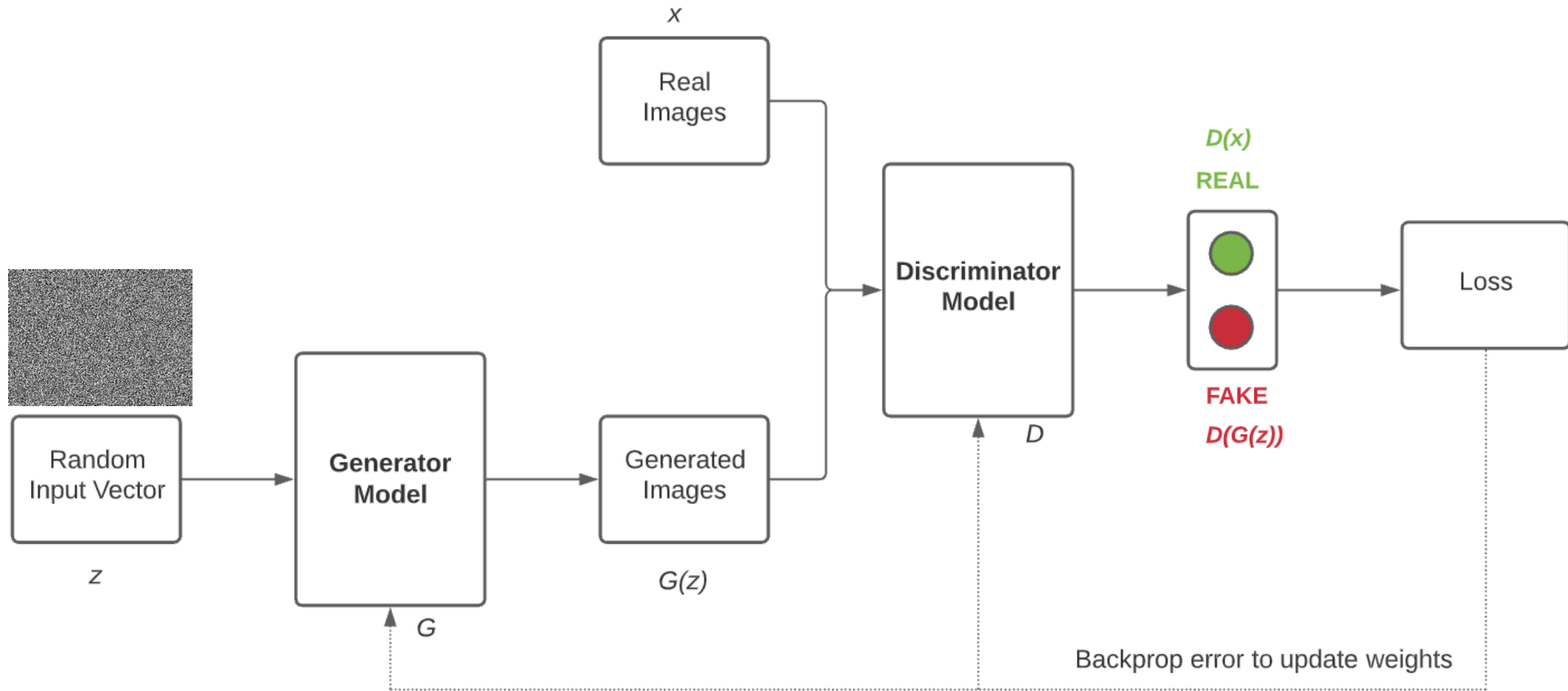
Steps of training

When we are training the Generator, we have to freeze the Discriminator and back propagate errors to only update the Generator.



When we are training the Discriminator, we have to freeze the Generator and back propagate errors to only update the Discriminator.





Steps to train GAN

Step 1: Define the problem. Do you want to generate fake images or fake text. Here you should completely define the problem and collect data for it.

Step 2: Define architecture of GAN. Define how your GAN should look like. Should both your generator and discriminator be multi layer perceptrons, or convolutional neural networks? This step will depend on what problem you are trying to solve.

Step 3: Train Discriminator on real data for n epochs. Get the data you want to generate fake on and train the discriminator to correctly predict them as real. Here value n can be any natural number between 1 and infinity.

Step 4: Generate fake inputs for generator and train discriminator on fake data. Get generated data and let the discriminator correctly predict them as fake.

Step 5: Train generator with the output of discriminator. Now when the discriminator is trained, you can get its predictions and use it as an objective for training the generator. Train the generator to fool the discriminator.

Step 6: Repeat step 3 to step 5 for a few epochs.

Step 7: Check if the fake data manually if it seems legit. If it seems appropriate, stop training, else go to step 3

Loss Function of GAN

$$\min_G \max_D V(D, G)$$

$$V(D, G) = \underbrace{\mathbb{E}_{x \sim p_{data}(x)} [\log D(x)]}_{\text{Discriminator output for real data}} + \underbrace{\mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]}_{\text{Discriminator output for Fake data}}$$

Discriminator output for real data

Discriminator output for Fake data

Generator wants to minimize the $V(D, G)$ whereas the Discriminator wants to maximize the $V(D, G)$.

1. $\mathbb{E}_{x \sim p_{data}(x)} [\log D(x)]$: Average log probability of D when real image is input.

2. $\mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$: Average log probability of D when the generated image is input.

G	Generator Model
D	Discriminator Model
z	Random Noise (Fixed size input vector)
x	Real Image
$G(z)$	Image generated by Generator (Fake Image)
$p_{data}(x)$	Probability Distribution of Real Images
$p_z(z)$	Probability Distribution of Fake Images
$D(G(z))$	Discriminator's output when the generated image is an input
$D(x)$	Discriminator's output when the real image is an input

Min-max

- $\min \max(D, G)$

As stated above, the discriminator seeks to maximize the average of the log probability of real images and the log of the inverse probability for fake images.

- discriminator: maximize $\log D(x) + \log(1 - D(G(z)))$

The generator seeks to minimize the log of the inverse probability predicted by the discriminator for fake images. This has the effect of encouraging the generator to generate samples that have a low probability of being fake.

- generator: minimize $\log(1 - D(G(z)))$

Generator's perspective

The Generator wants to minimize the loss function $V(D, G)$ by generating images that look like real images and tries to fool the Discriminator.

The second term suggests that the Generator wants to make $D(G(z))$ as close to 1 (instead of 0) as possible and thus minimize the term eventually (1 – larger number will result in a smaller number). So that the Discriminator fails and **misclassifies** the images.

Thus, The Generator tries to minimize the second term.

Discriminator's perspective

The Discriminator wants to maximize the loss function $V(D, G)$ by correctly classifying real and fake images.

The first term suggests that the Discriminator wants to make $D(x)$ as close to 1 as possible, i.e. correctly classifying real images as real.

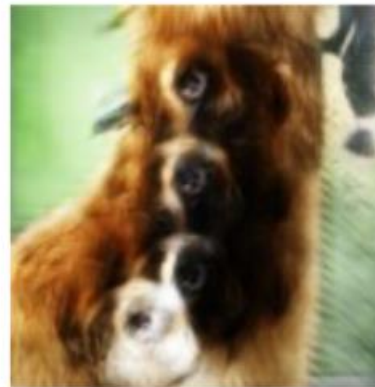
The second term suggests that the Discriminator wants to make $D(G(x))$ as close to 0 as possible, i.e. correctly classifying fake images as fake and thus maximize the term eventually (1 – smaller number will result in a larger number). *Note: Probability lies in the range of 0-1.*

Thus, The Discriminator tries to maximize both the terms.

Challenges of GAN

Problem with Counting: GANs fail to differentiate how many of a particular object should occur at a location.

Problems with Counting

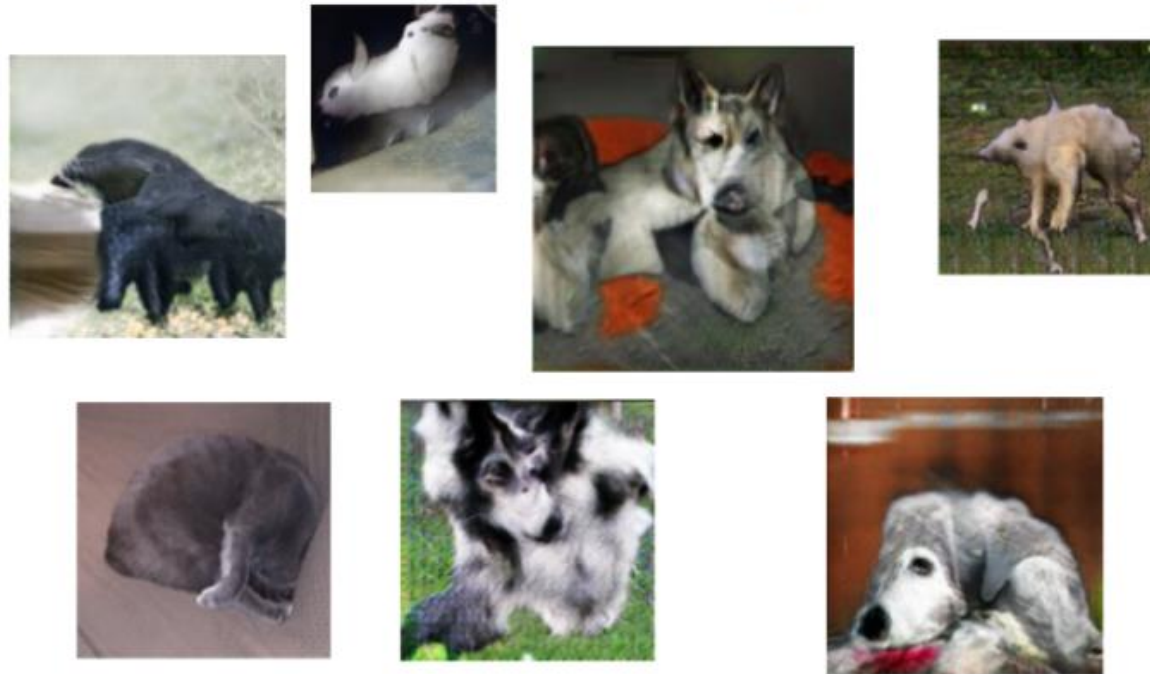


(Goodfellow 2016)

Challenges of GAN

Problems with Perspective: GANs fail to adapt to 3D objects. It doesn't understand perspective, i.e. difference between frontview and backview.

Problems with Perspective

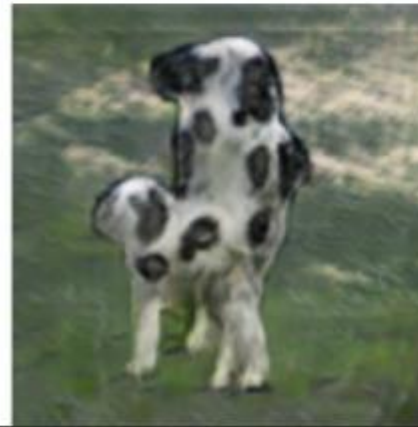


(Goodfellow 2016)

Challenges of GAN

Problems with Global Structures: Same as the problem with perspective, GANs do not understand a holistic structure. For example, in the bottom left image, it gives a generated image of a quadruple cow, i.e. a cow standing on its hind legs and simultaneously on all four legs. That is definitely not possible in real life!

Problems with Global Structure



(Goodfellow 2016)

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
- Sample minibatch of m examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(x^{(i)}) + \log \left(1 - D(G(z^{(i)})) \right) \right].$$

end for

- Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D(G(z^{(i)})) \right).$$

end for

Thank You

Namah Shivaya