UNIVERSIDADE FEDERAL DO CEARÁ
Campus Russas

Redes Adversárias Geradoras e suas aplicações

Prof. Dsc. Nauber Gois

Apresentação

Francisco Nauber Bernardo Gois

Professor Adjunto-A
da Universidade Federal do Ceará

Doutor em Informática Aplicada

Mestre em Informática Aplicada

Especialista em desenvolvimento WEB

naubergois@ufc.br

Why Study GAN Networks?

Why Generative Models?

- **We've only seen discriminative models so far**
  - Given an image **X**, predict a label **Y**
  - Estimates **P(Y|X)**

- **Discriminative models have several key limitations**
  - Can't model **P(X)**, i.e. the probability of seeing a certain image
  - Thus, can't sample from **P(X)**, i.e. **can't generate new images**

- **Generative models (in general) cope with all of above**
  - Can model **P(X)**
  - Can generate new images

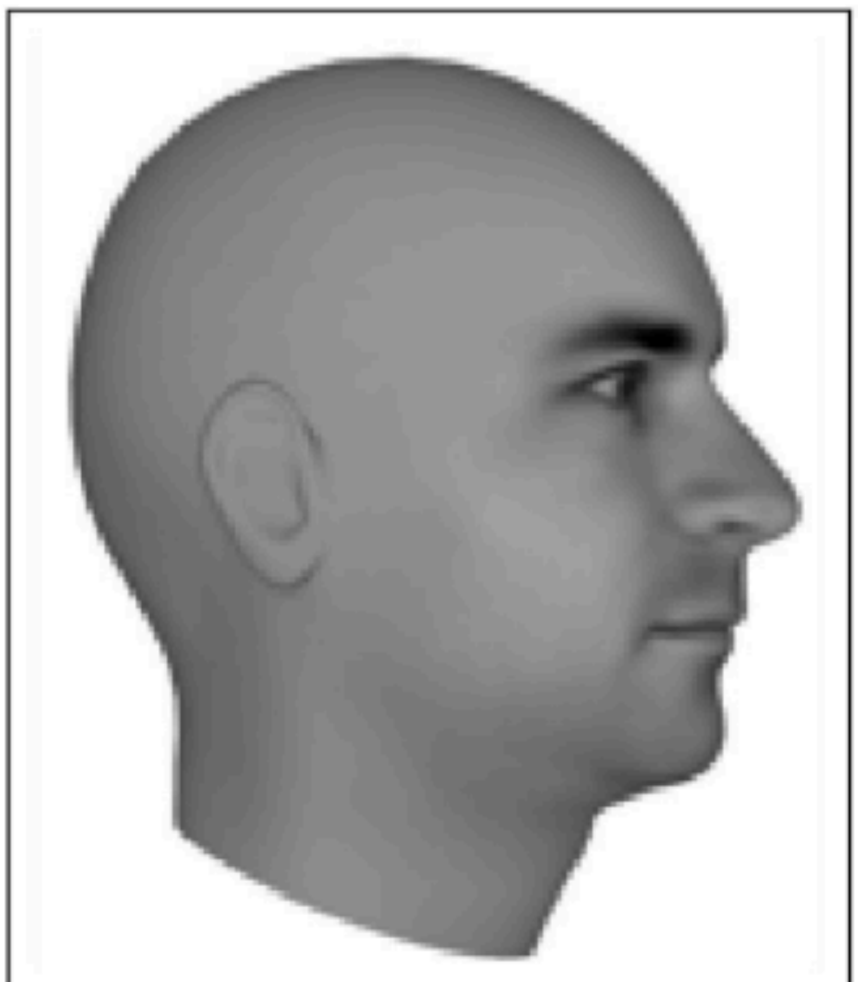
naubergois@ufc.br

Why Generative Models?

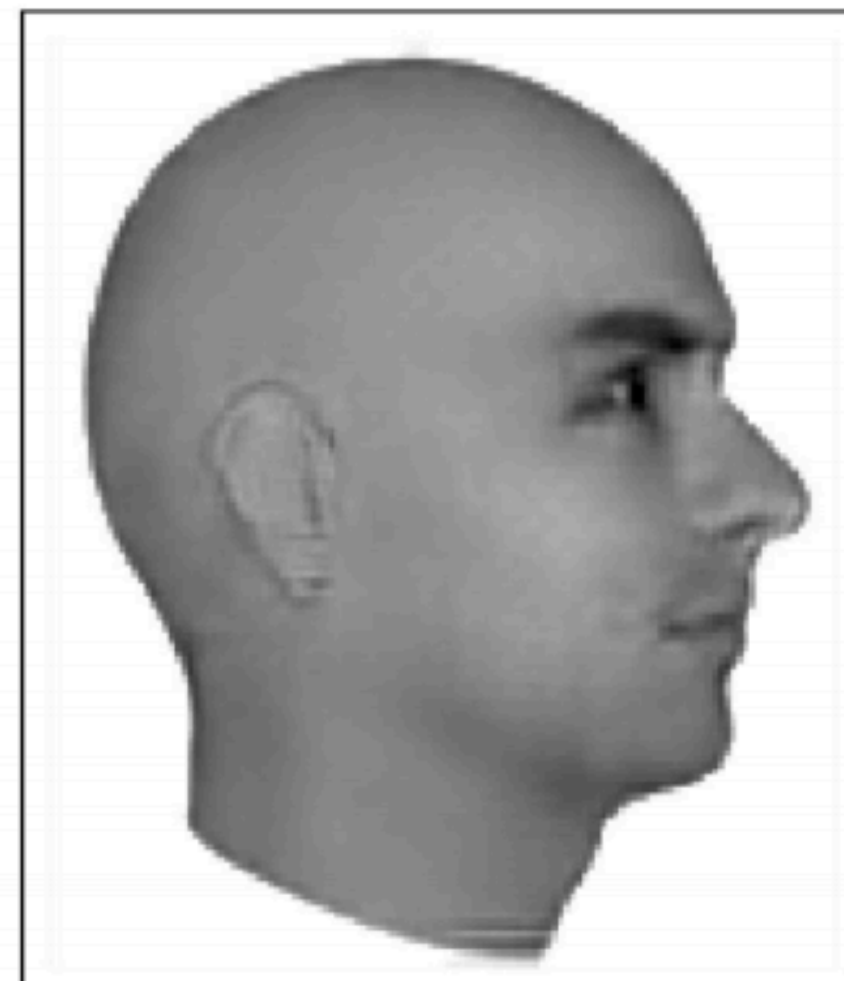
naubergois@ufc.br

Magic of GANs…

Ground Truth



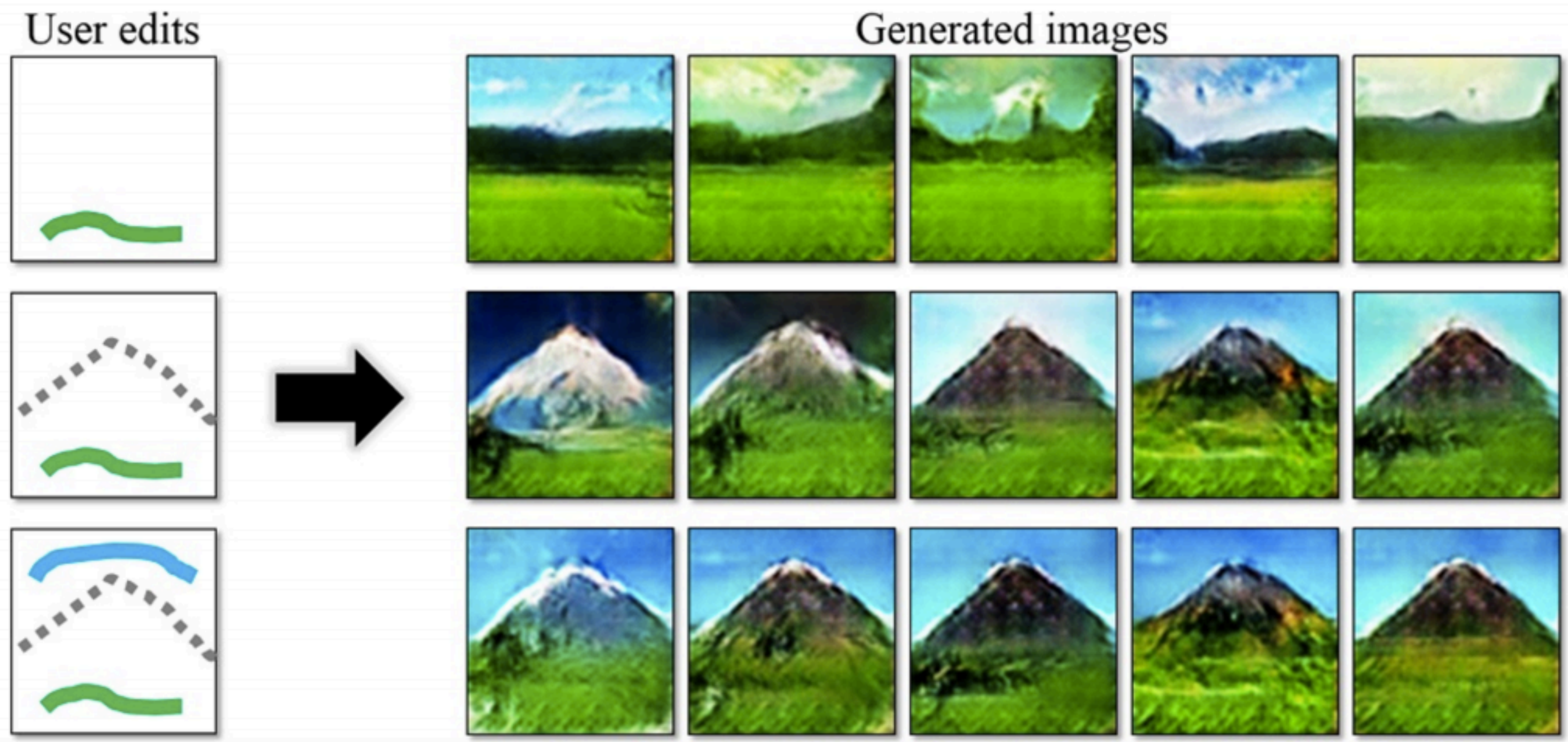
Adversarial



Magic of GANs…

Which one is Computer generated?

Magic of GANs...

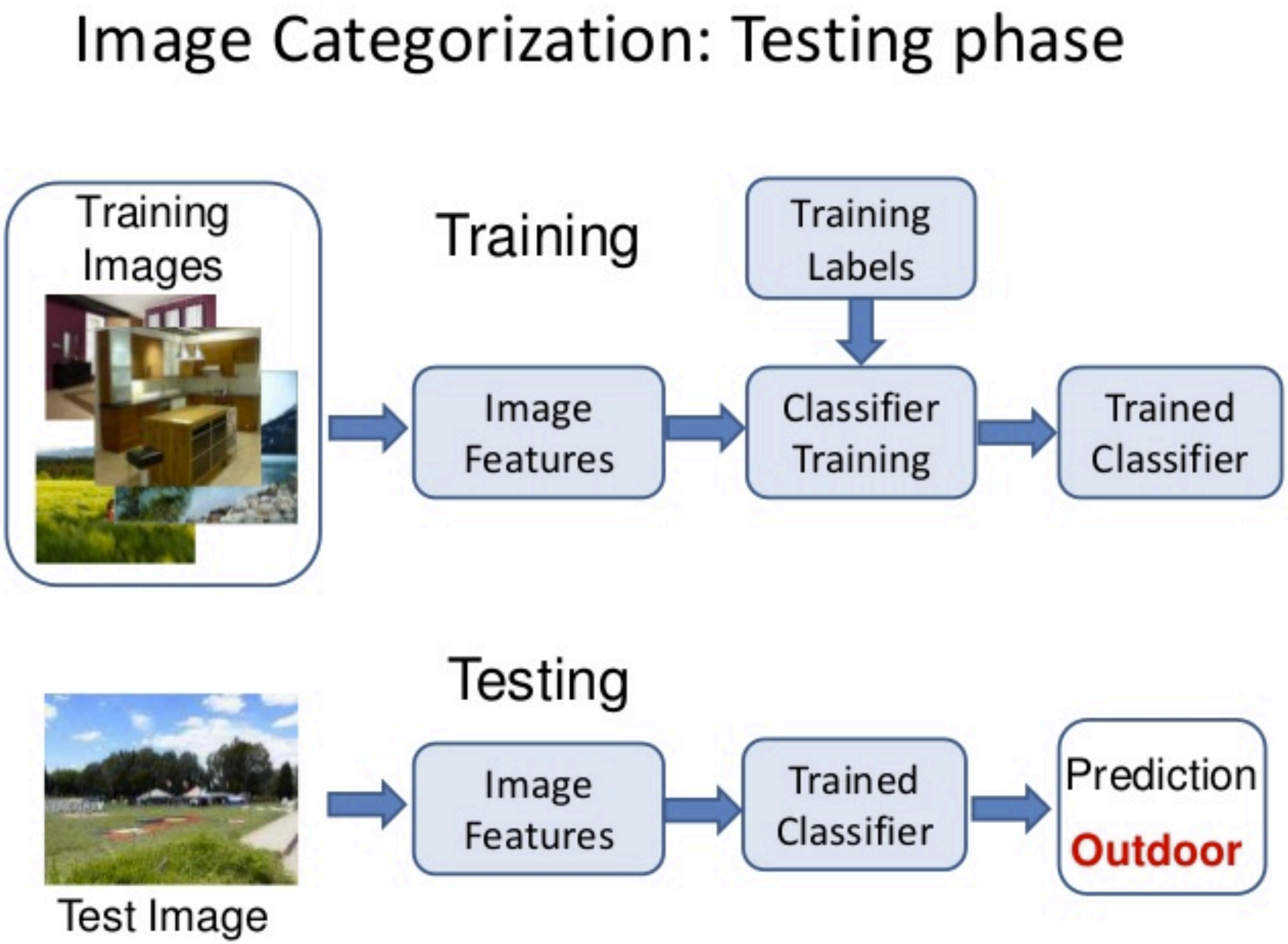


"(...) (GANs) and the variations that are now being proposed is the most interesting idea in the last 10 years in ML (...)"

Yann LeCun
Director of Facebook AI Research

Source: https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-deep-learning/answer/Yann-LeCun

Aprendizado Supervisionado



naubergois@ufc.br

Neural Networks

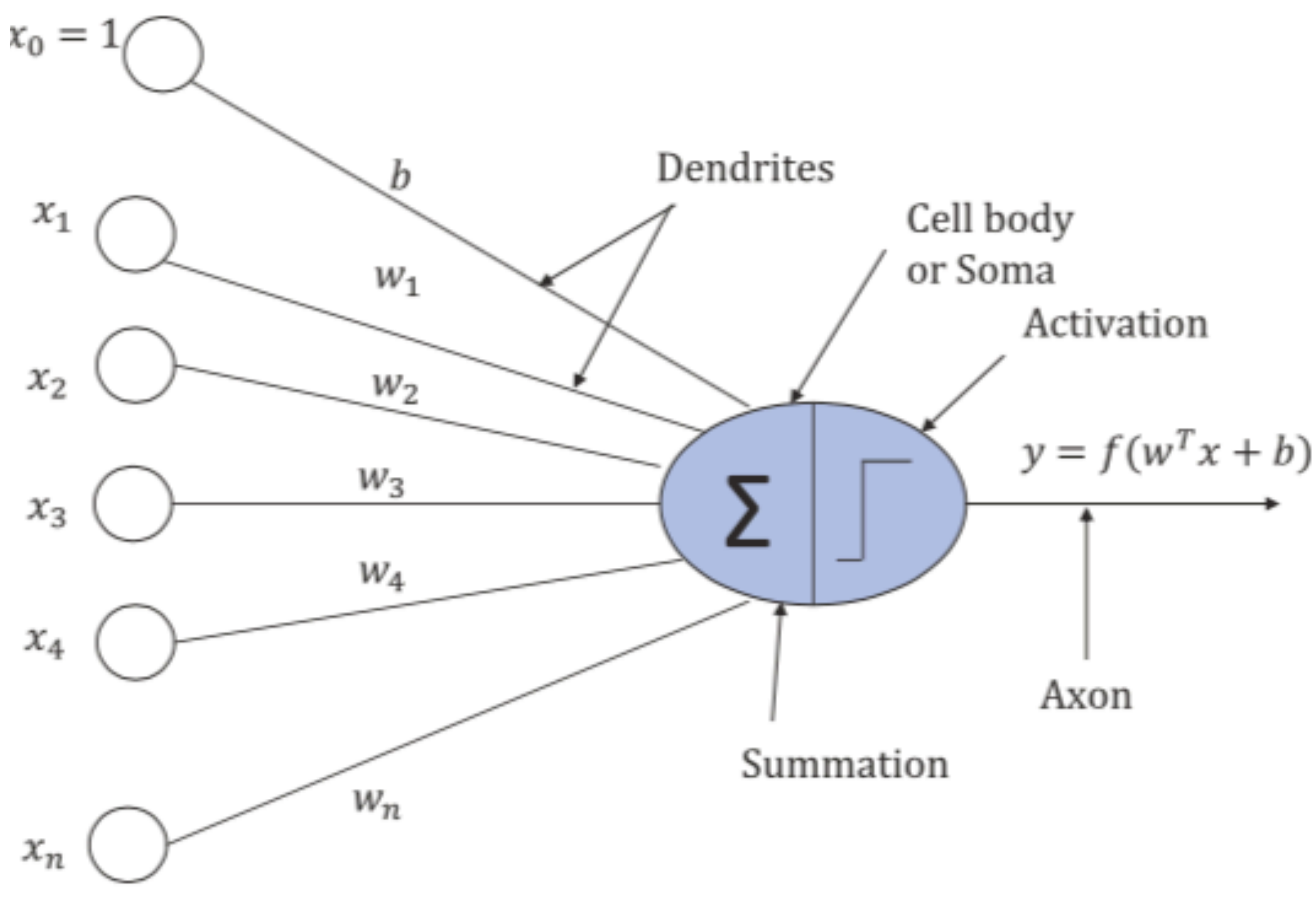
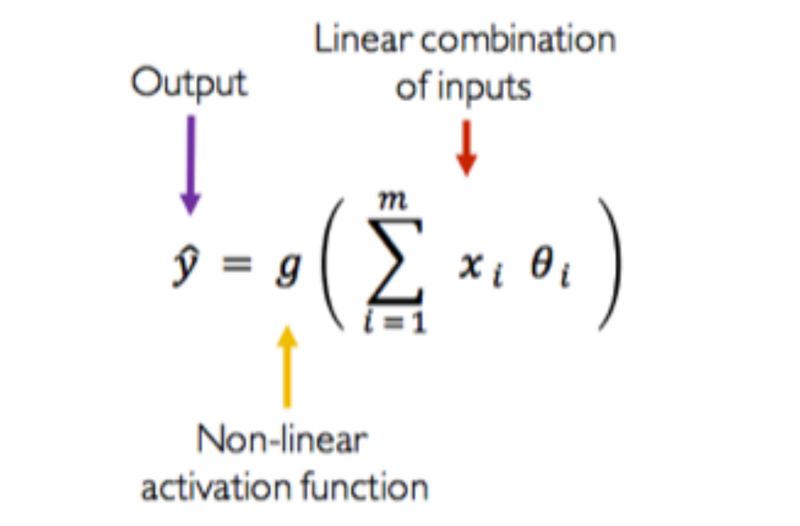
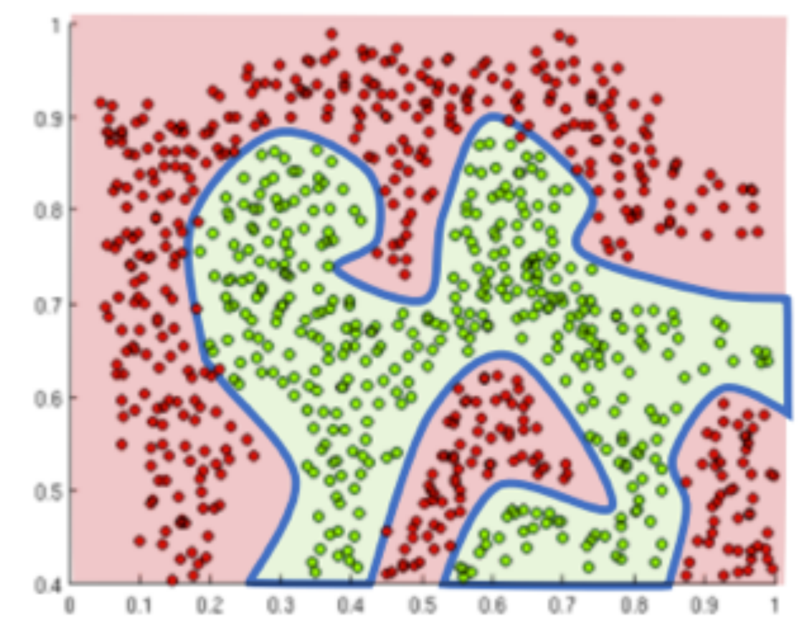


Figure 2-2. Structure of an artificial neuron





Non-linearities allow us to approximate arbitrarily complex functions

naubergois@ufc.br

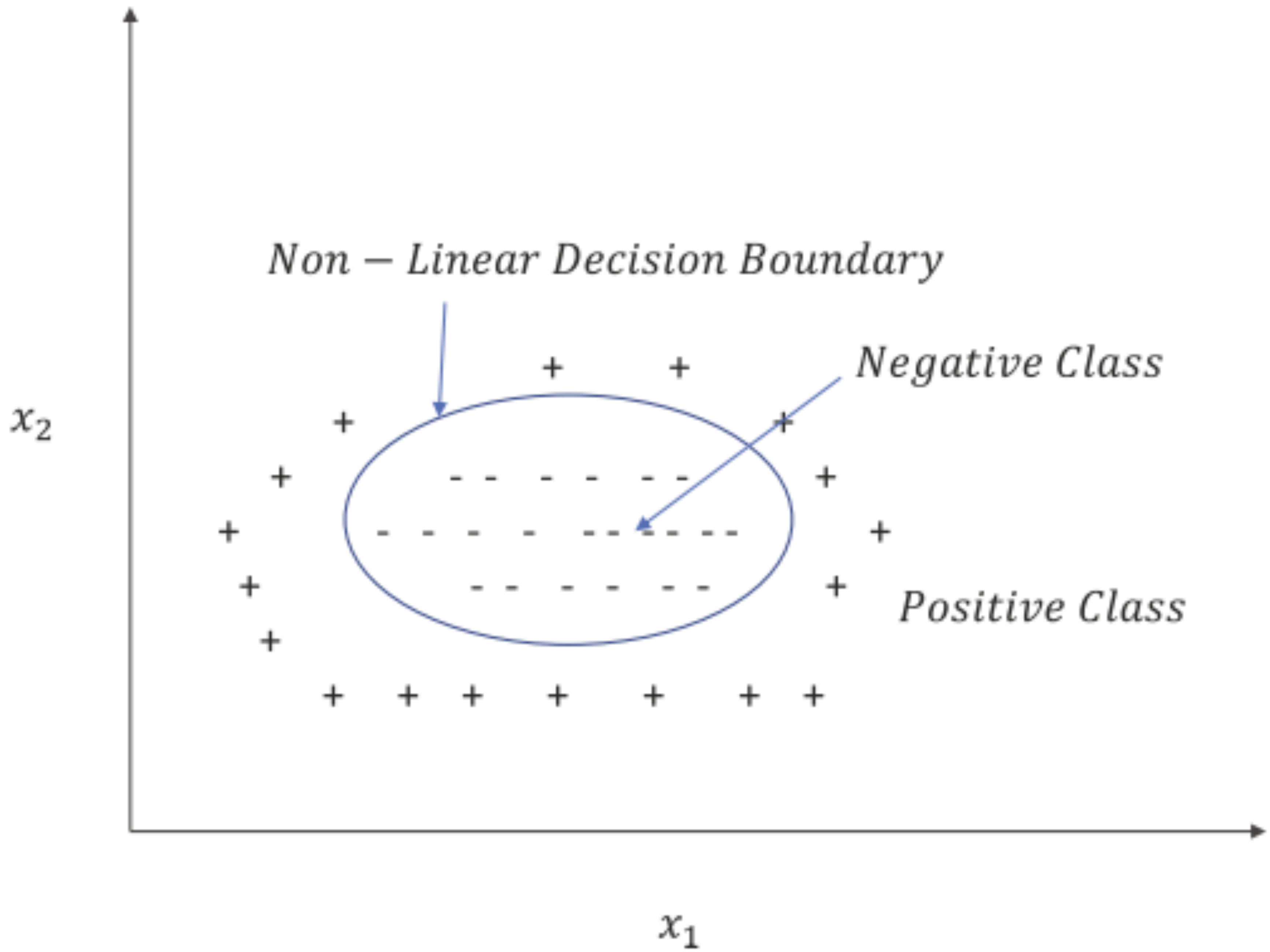
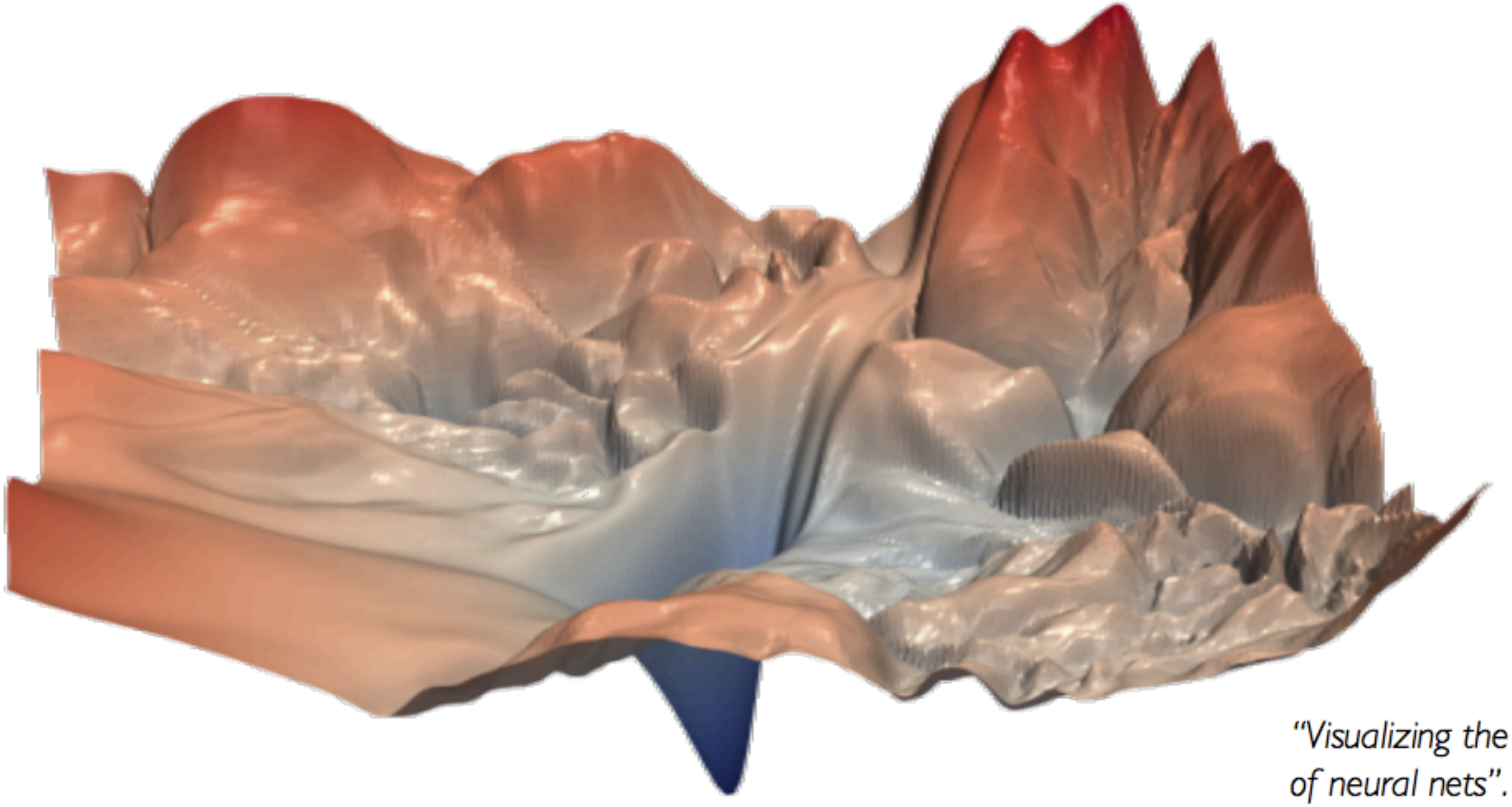


Figure 1-35. *Classification by a non-linear decision boundary*

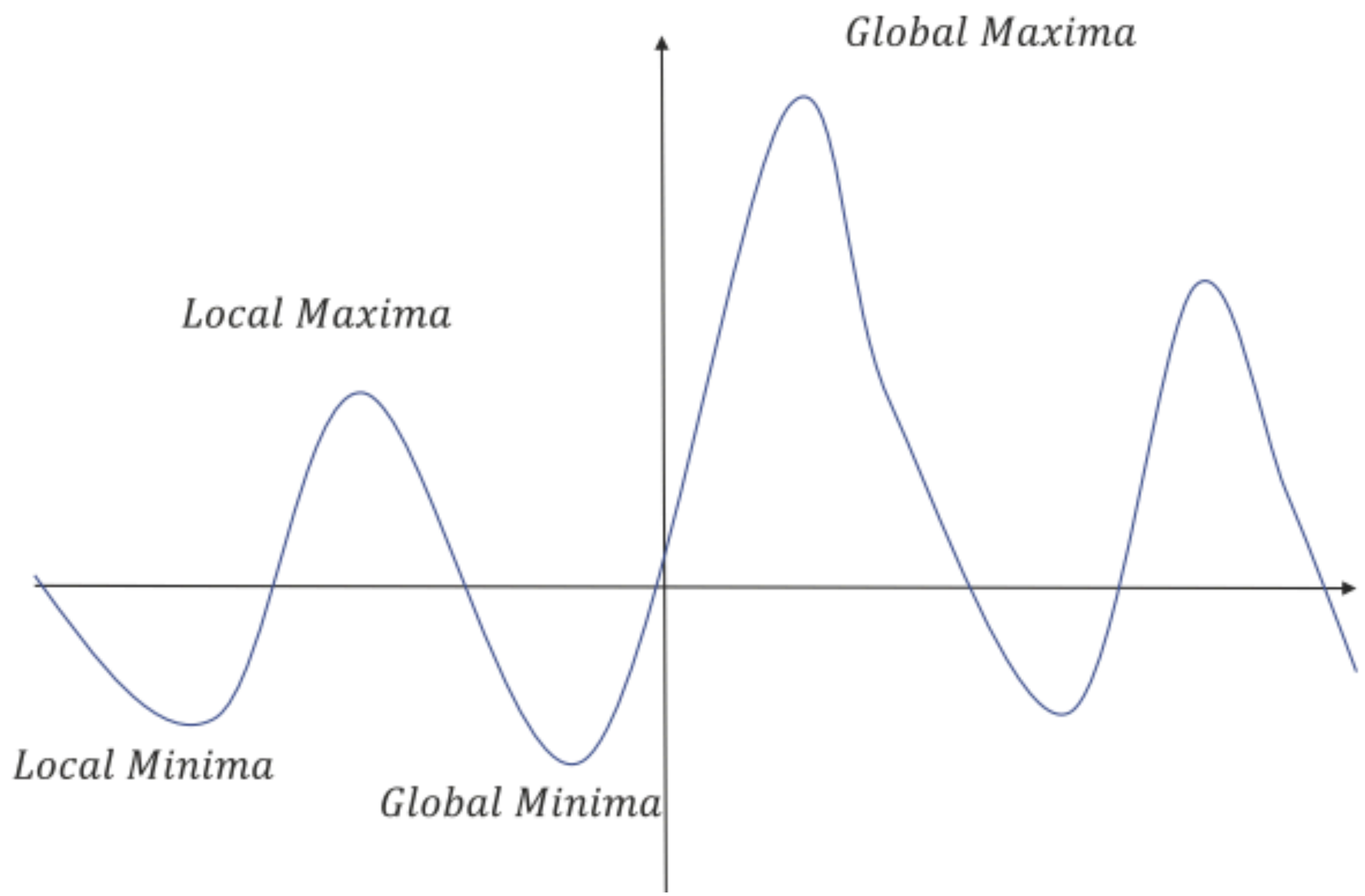
Neural Networks



"Visualizing the loss landscape of neural nets". Dec 2017.

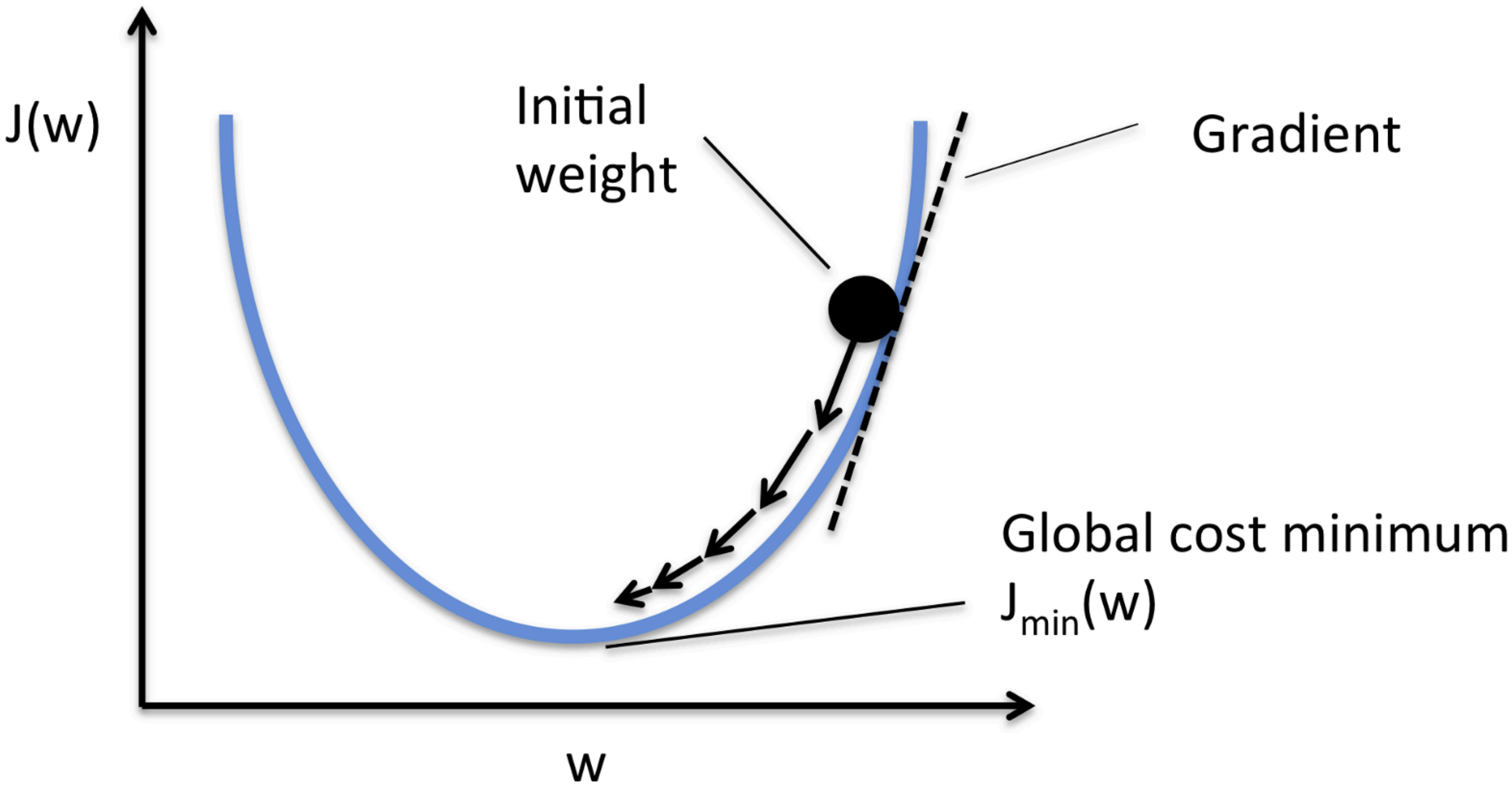
naubergois@ufc.br

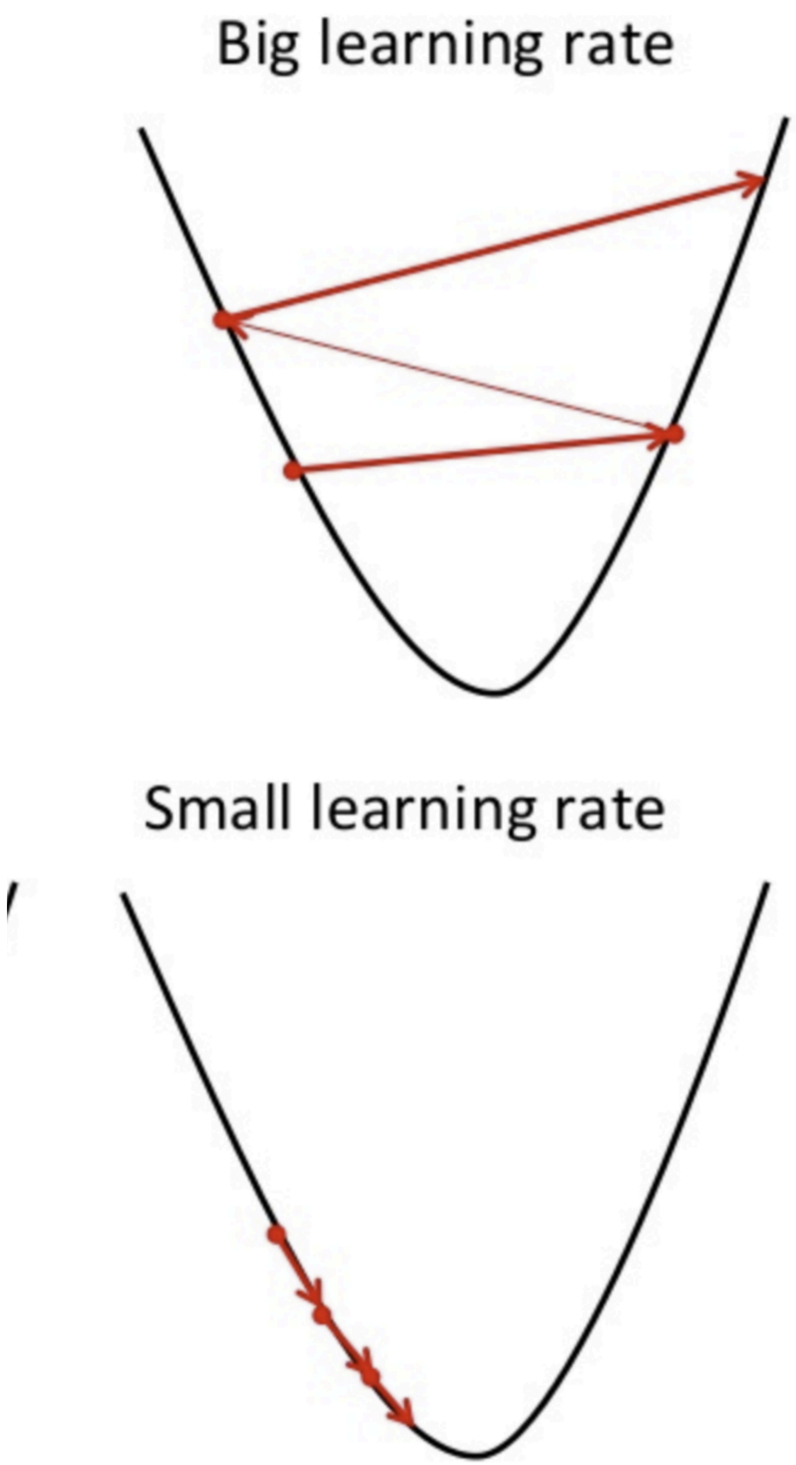
Neural Networks



$$\theta^{(t+1)} = \theta^{(t)} - \eta \nabla C\left(\theta^{(t)}\right)$$

Neural Networks





Neural Networks

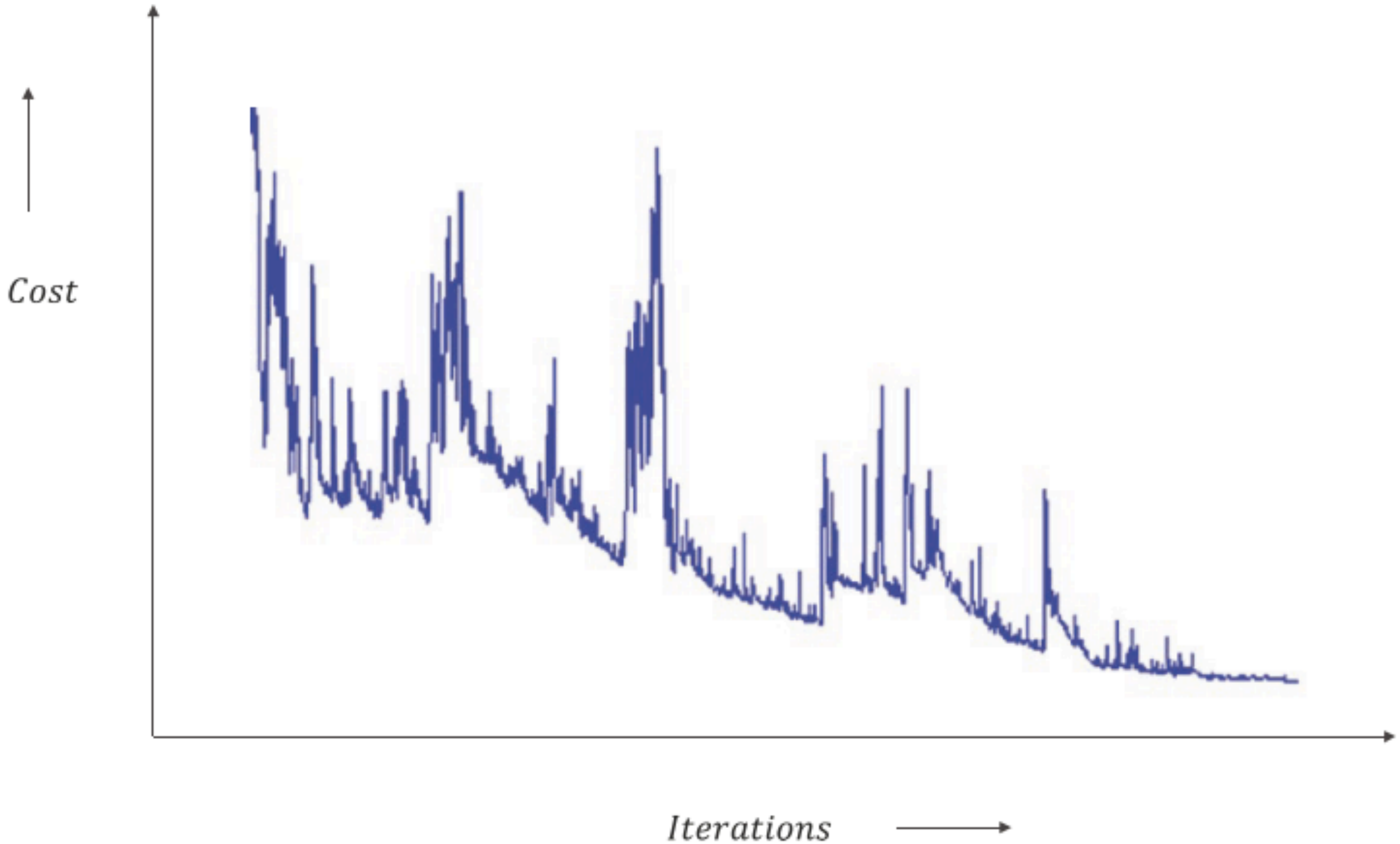


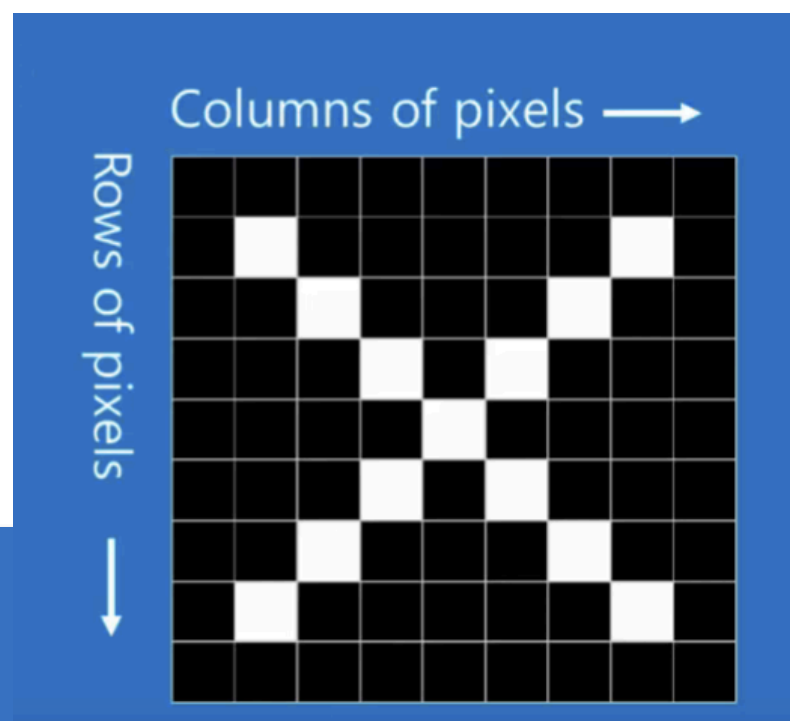
Figure 1-40. Fluctuation in the total cost function value over iterations in stochastic gradient descent

IMAGE

Image

[255,254,255, ….., 255,255,254]
[255,254,255, ….., 255,255,254]
[255,254,255, ….., 255,255,254]
…,
[255,254,255, ….., 255,255,254]
[255,254,255, ….., 255,255,254]
[255,254,255, ….., 255,255,254]

Internal Representation in Matrix form



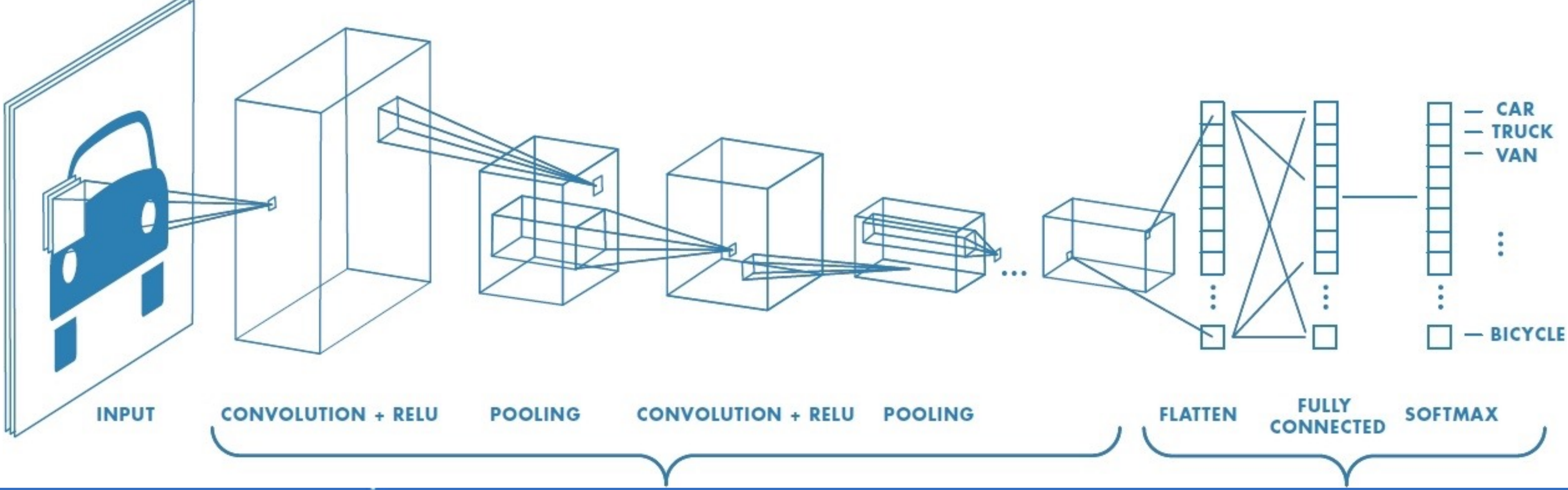
naubergois@ufc.br

Generative Networks vs Discriminative Networks

Supervised Learning

- Challenges
  - Image is **high dimensional** data
    - 224 x 224 x 3 = 150,528
  - Many **variations**
    - viewpoint, illumination, deformation, occlusion, background clutter, intraclass variation

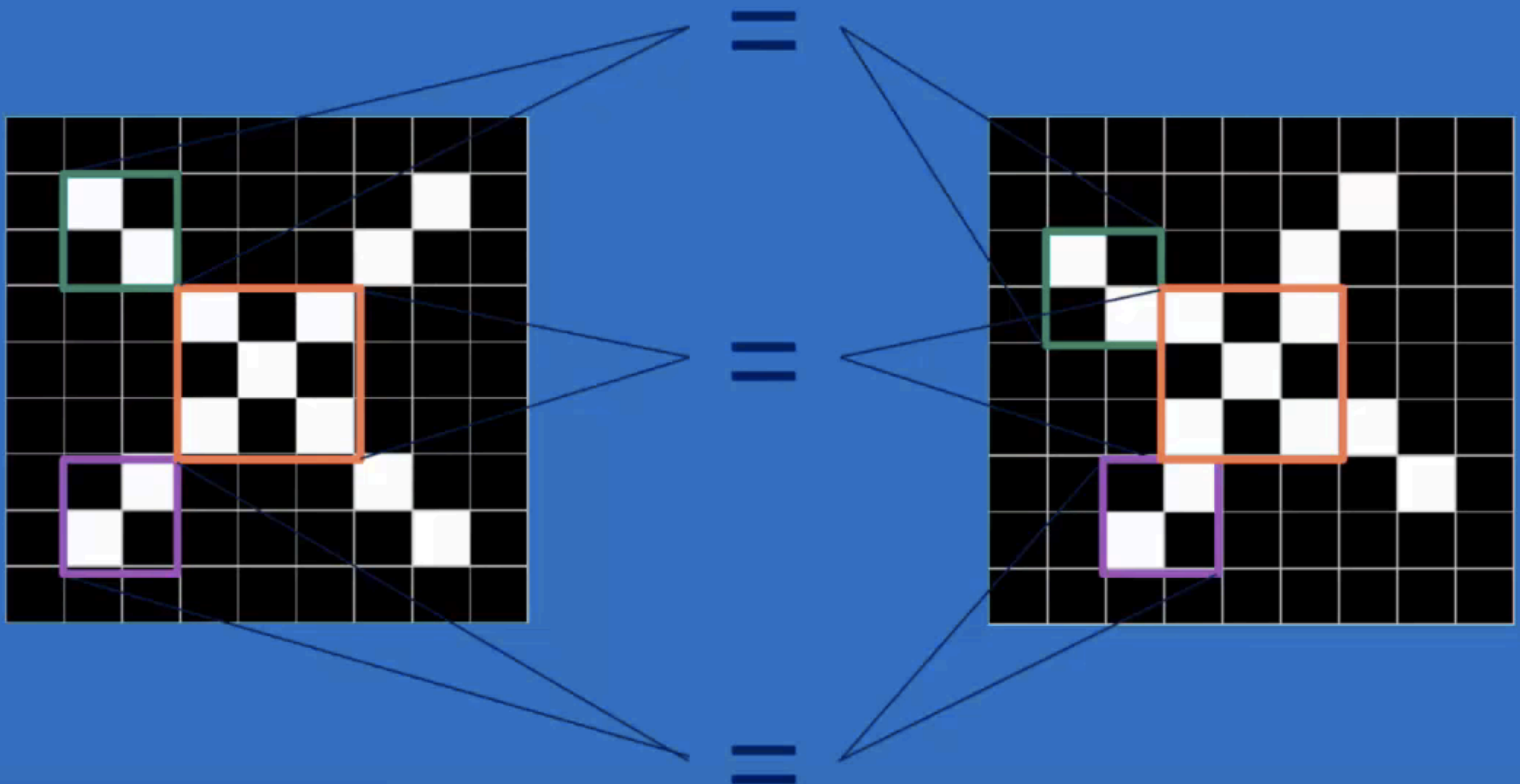
naubergois@ufc.br



-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

$\overset{?}{=}$

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	1	-1	-1	-1
-1	-1	1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

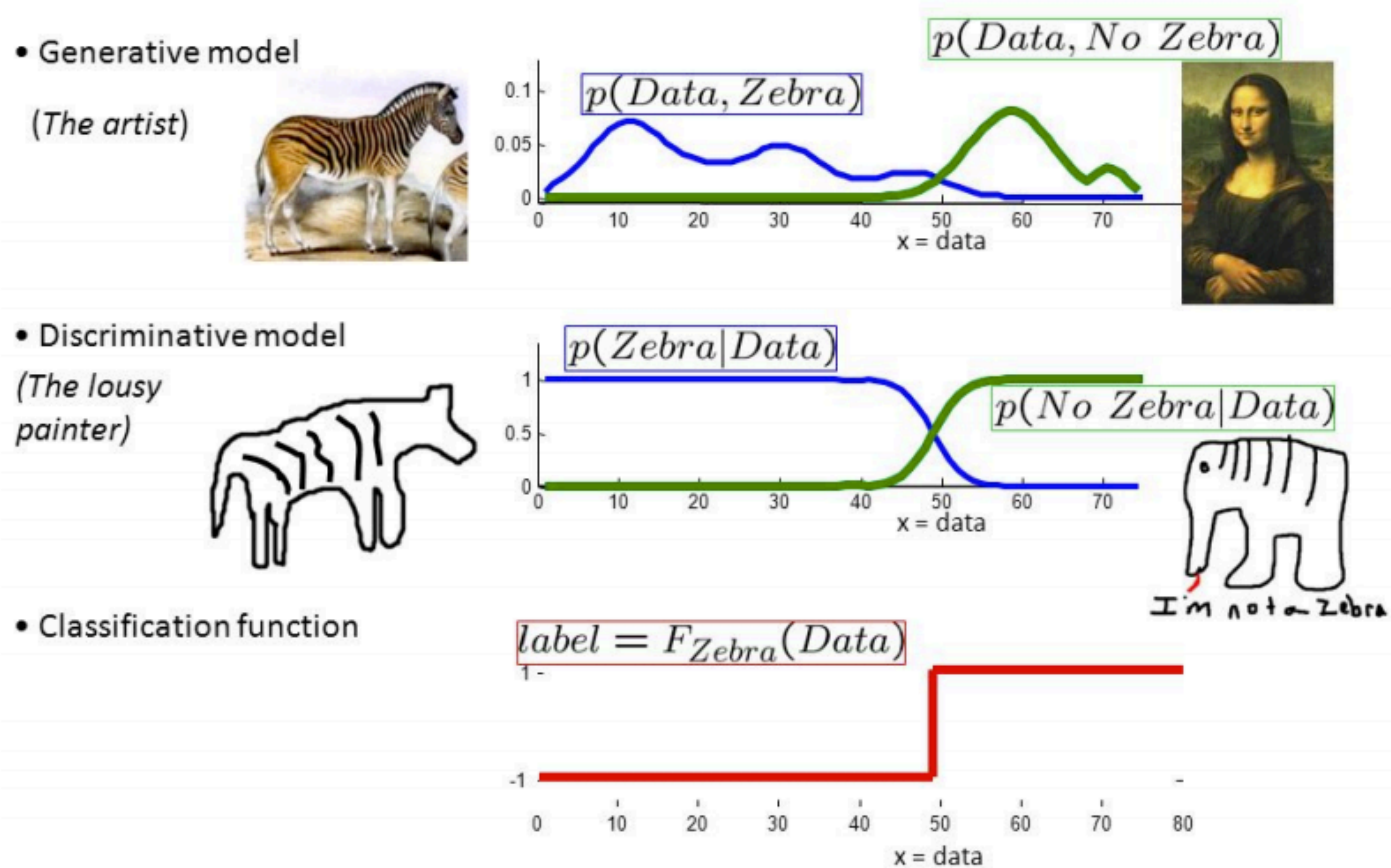


$$\begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 0.77 & -0.11 & 0.11 & 0.33 & 0.55 & -0.11 & 0.33 \\ -0.11 & 1.00 & -0.11 & 0.33 & -0.11 & 0.11 & -0.11 \\ 0.11 & -0.11 & 1.00 & -0.33 & 0.11 & -0.11 & 0.55 \\ 0.33 & 0.33 & -0.33 & 0.55 & -0.33 & 0.33 & 0.33 \\ 0.55 & -0.11 & 0.11 & -0.33 & 1.00 & -0.11 & 0.11 \\ -0.11 & 0.11 & -0.11 & 0.33 & -0.11 & 1.00 & -0.11 \\ 0.33 & -0.11 & 0.55 & 0.33 & 0.11 & -0.11 & 0.77 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 0.33 & -0.55 & 0.11 & -0.11 & 0.11 & -0.55 & 0.33 \\ -0.55 & 0.55 & -0.55 & 0.33 & -0.55 & 0.55 & -0.55 \\ 0.11 & -0.55 & 0.55 & -0.77 & 0.55 & -0.55 & 0.11 \\ -0.11 & 0.33 & -0.77 & 1.00 & -0.77 & 0.33 & -0.11 \\ 0.11 & -0.55 & 0.55 & -0.77 & 0.55 & -0.55 & 0.11 \\ -0.55 & 0.55 & -0.55 & 0.33 & -0.55 & 0.55 & -0.55 \\ 0.33 & -0.55 & 0.11 & -0.11 & 0.11 & -0.55 & 0.33 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix} \otimes \begin{bmatrix} -1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} 0.33 & -0.11 & 0.55 & 0.33 & 0.11 & -0.11 & 0.77 \\ -0.11 & 0.11 & -0.11 & 0.33 & -0.11 & 1.00 & -0.11 \\ 0.55 & -0.11 & 0.11 & -0.33 & 1.00 & -0.11 & 0.11 \\ 0.33 & 0.33 & -0.33 & 0.55 & -0.33 & 0.33 & 0.33 \\ 0.11 & -0.11 & 1.00 & -0.33 & 0.11 & -0.11 & 0.55 \\ -0.11 & 1.00 & -0.11 & 0.33 & -0.11 & 0.11 & -0.11 \\ 0.77 & -0.11 & 0.11 & 0.33 & 0.55 & -0.11 & 0.33 \end{bmatrix}$$

Generative Networks vs Discriminative Networks



http://slideplayer.com/slide/6982498/

naubergois@ufc.br

Generative Networks vs Discriminative Networks

Generative Model

- Find generation function ***g*** : *x = g(z)*, x : data, z : **latent**

naubergois@ufc.br

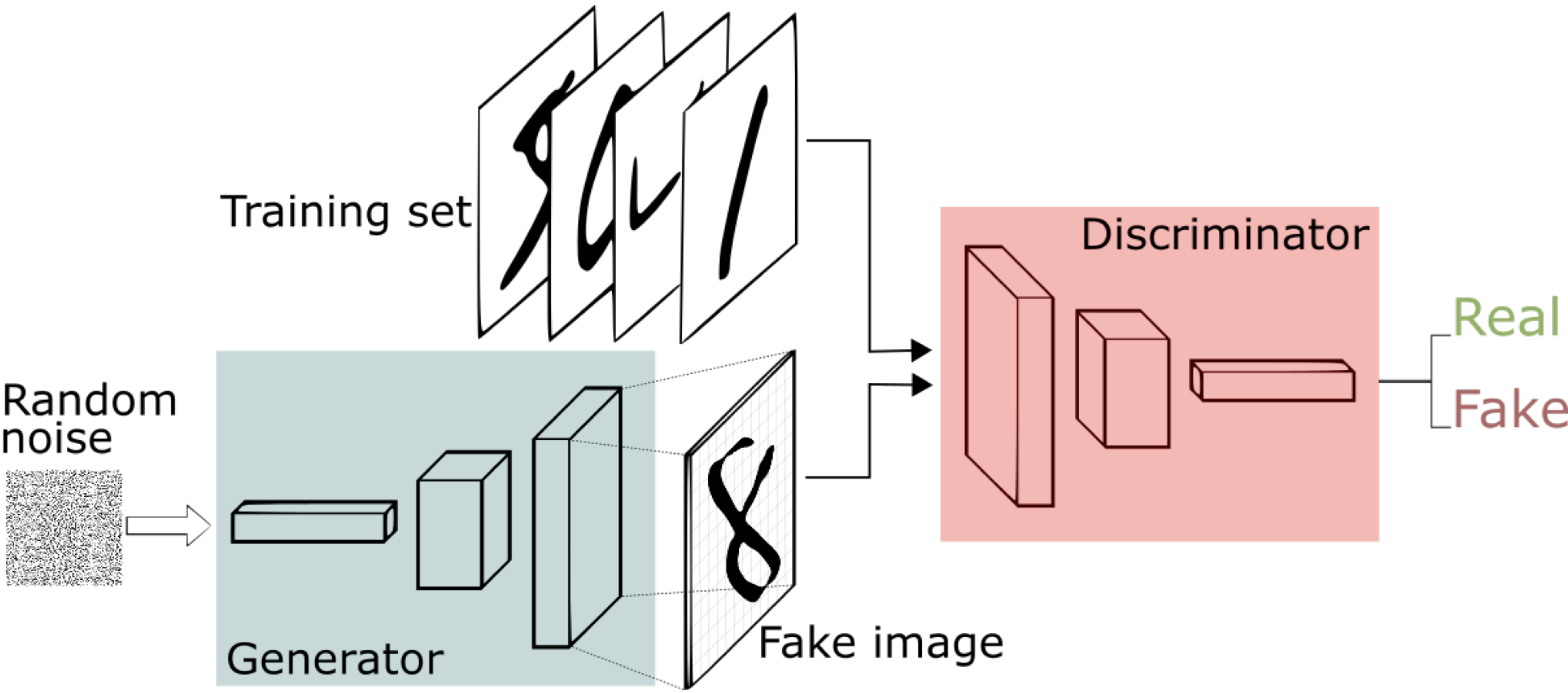
“What I do not understand,
I cannot create.”

Richard Feynman
Nobel in Physics in 1965

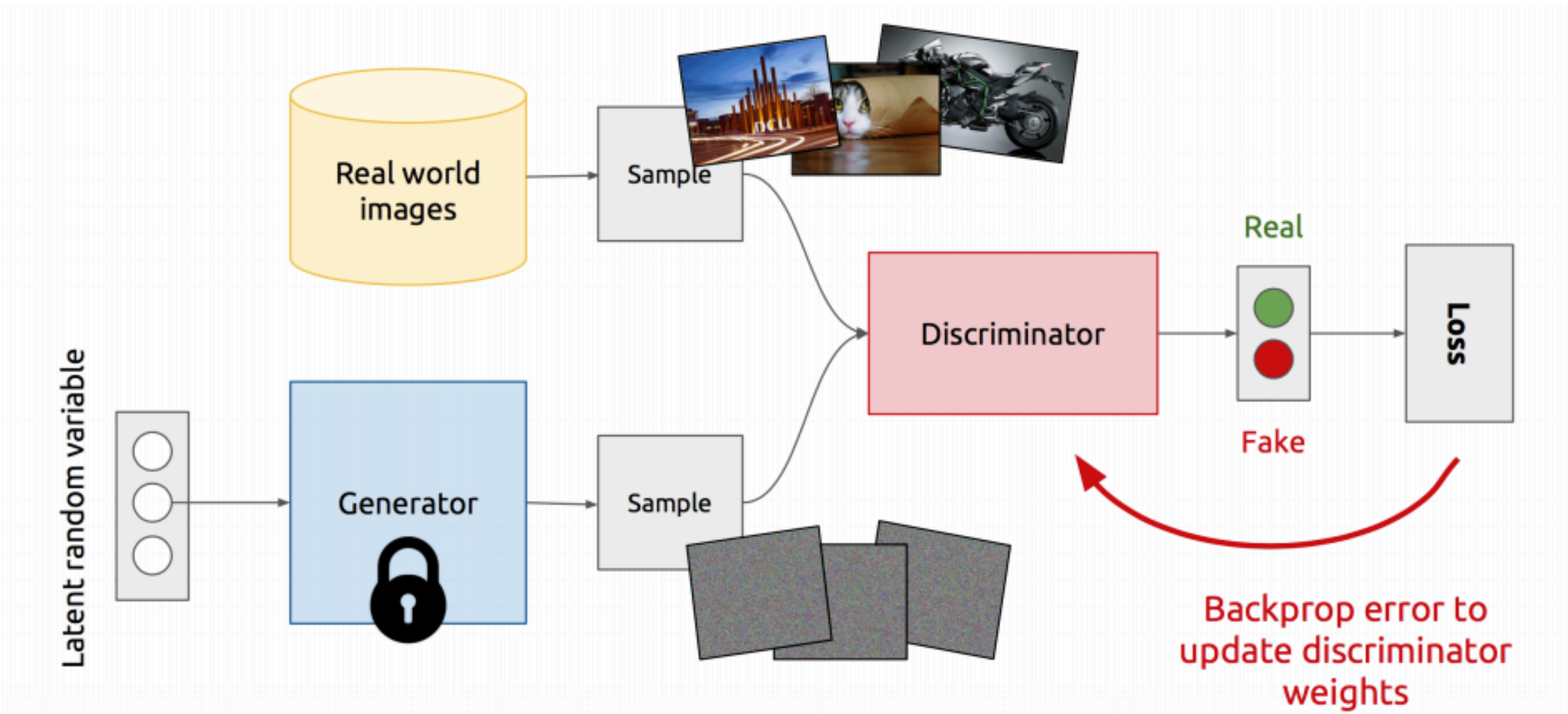
Source: https://openai.com/blog/generative-models/

GANs

GENERATIVE ADVERSARIAL NETWORK



Training Discriminator



GAN's formulation

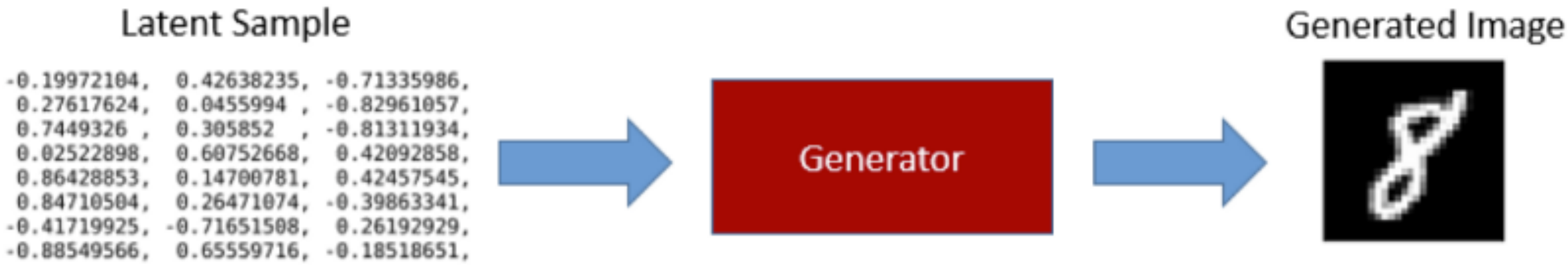
$$\min_G \max_D V(D, G)$$

- It is formulated as a **minimax game**, where:
  - The Discriminator is trying to maximize its reward $\boldsymbol{V(D, G)}$
  - The Generator is trying to minimize Discriminator's reward (or maximize its loss)

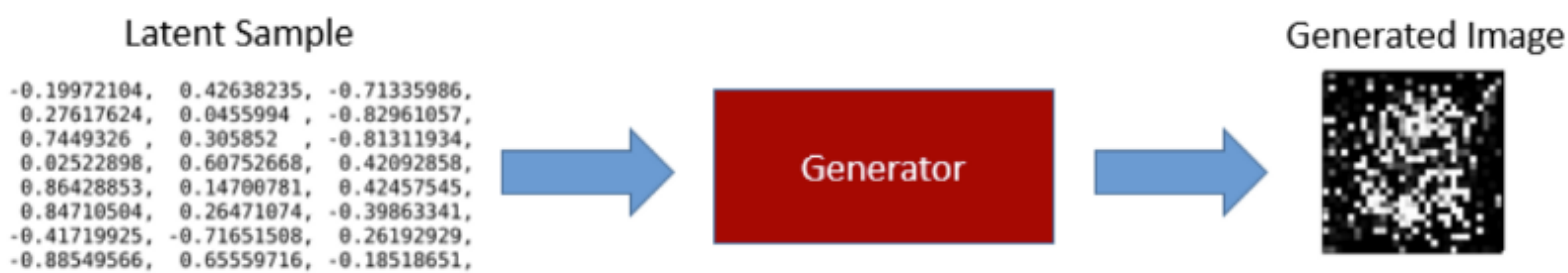
$$V(D, G) = \boxed{\mathbb{E}_{x \sim p(x)}[\log D(x)]} + \boxed{\mathbb{E}_{z \sim q(z)}[\log(1 - D(G(z)))]}$$

- The Nash equilibrium of this particular game is achieved at:
  - $P_{data}(x) = P_{gen}(x) \;\; \forall x$
  - $\mathrm{D}(x) = \frac{1}{2} \;\; \forall x$

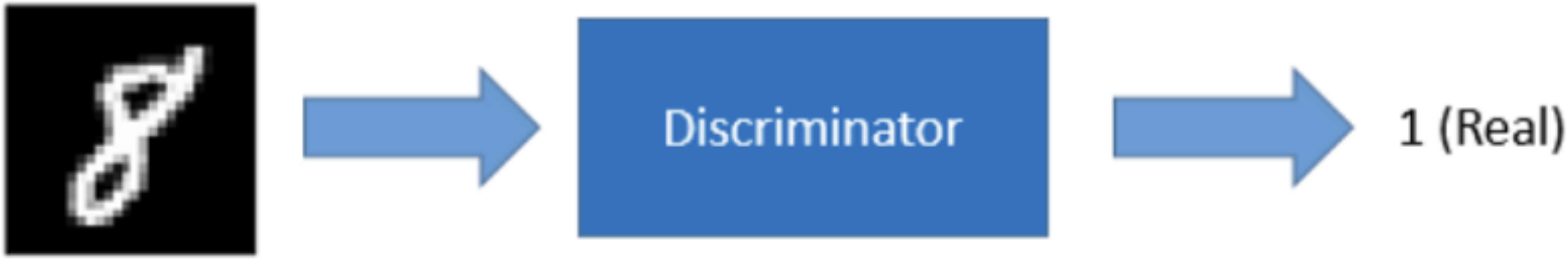
GENERATIVE ADVERSARIAL NETWORK



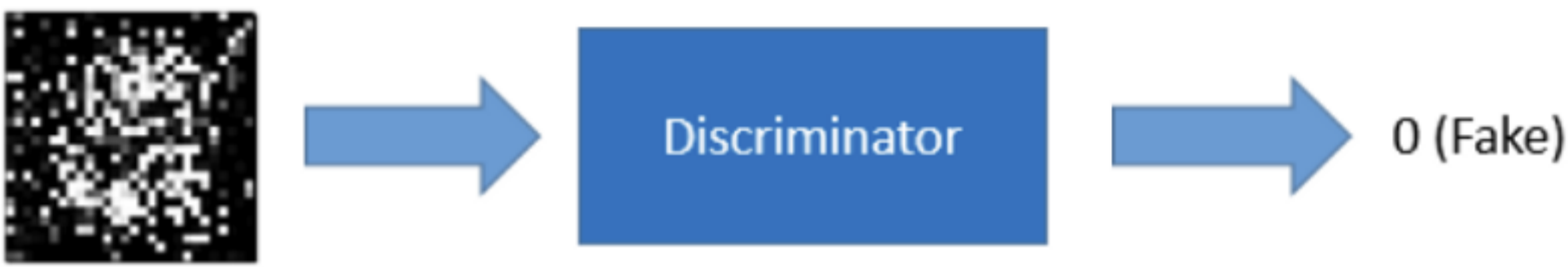

```
generator = Sequential([
    Dense(128, input_shape=(100,)),
    LeakyReLU(alpha=0.01),
    Dense(784),
    Activation('tanh')
], name='generator')
```



GENERATIVE ADVERSARIAL NETWORK




```
discriminator = Sequential([
    Dense(128, input_shape=(784,)),
    LeakyReLU(alpha=0.01),
    Dense(1),
    Activation('sigmoid')
], name='discriminator')
```



GENERATIVE ADVERSARIAL NETWORK

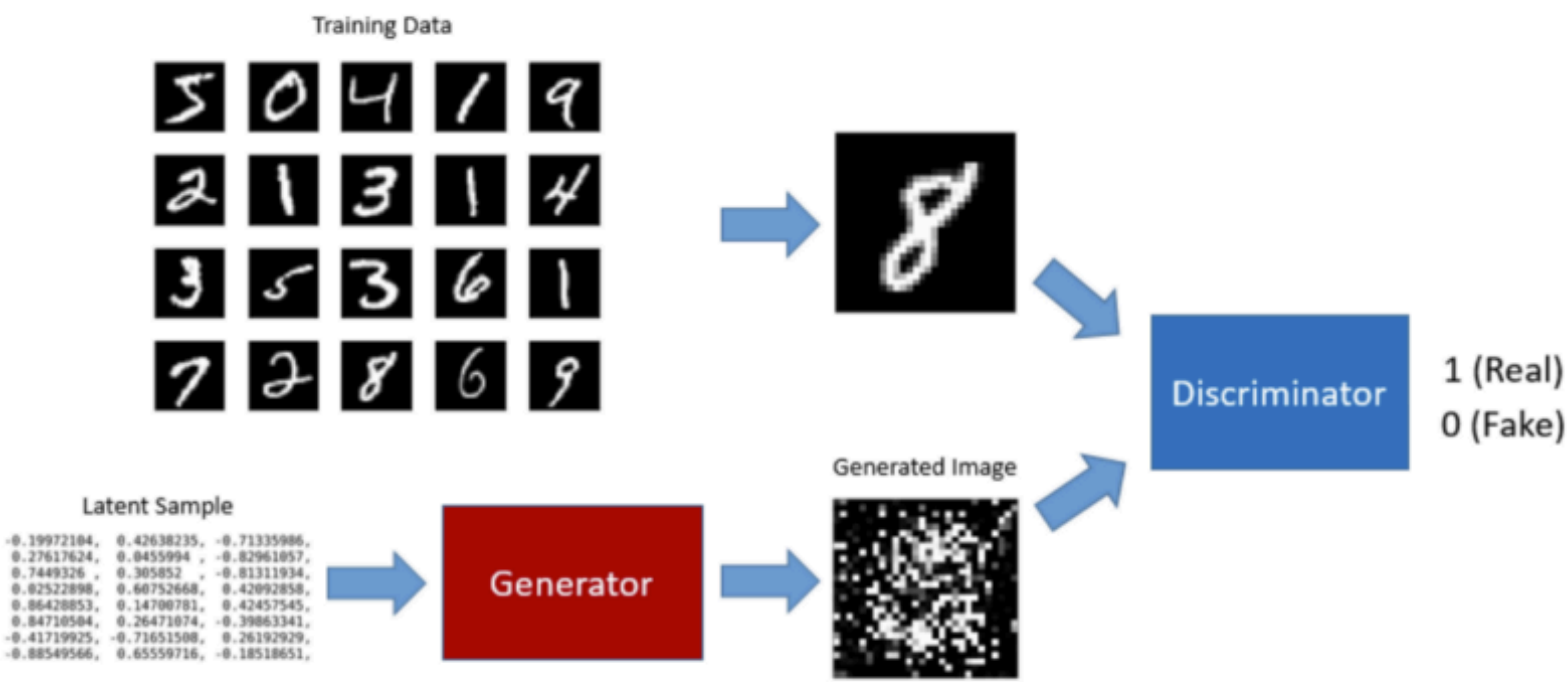
```
gan = Sequential([
    generator,
    discriminator
])
```

GENERATIVE ADVERSARIAL NETWORK

Step 1) Set the discriminator trainable

Step 2) Train the discriminator with the real MNIST digit images and the images generated by the generator to classify the real and fake images.

GENERATIVE ADVERSARIAL NETWORK



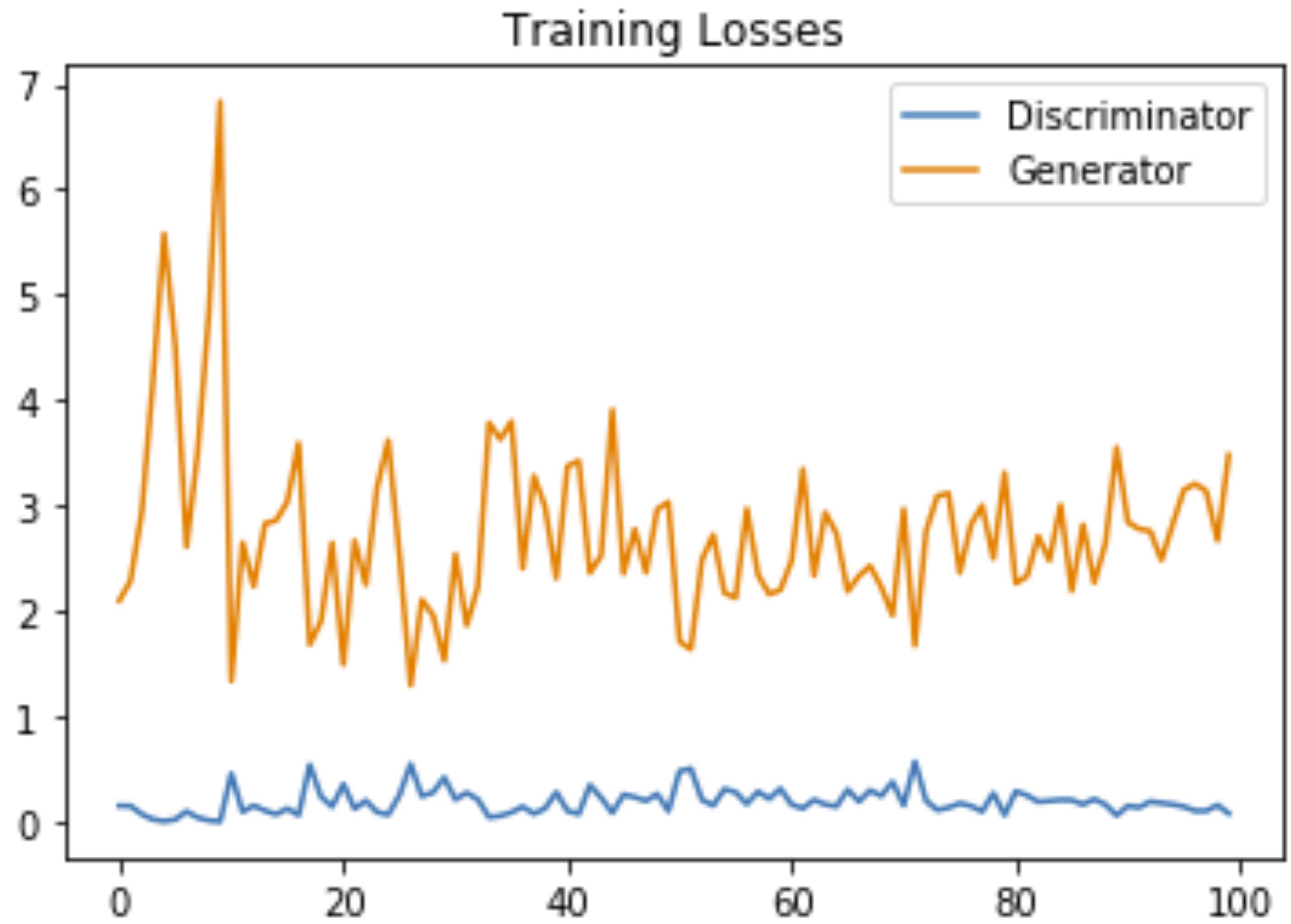
GENERATIVE ADVERSARIAL NETWORK

Step 3) Set the discriminator non-trainable

Step 4) Train the generator as part of the GAN. We feed latent samples into the GAN and let the generator to produce digit images and use the discriminator to classify the image.

GENERATIVE ADVERSARIAL NETWORK

GENERATIVE ADVERSARIAL NETWORK



↓↓ REAL ↓↓

100
FEDERAL RESERVE NOTE
KB 42107303 D
B2
THE UNITED STATES OF AMERICA
THIS NOTE IS LEGAL TENDER FOR ALL DEBTS, PUBLIC AND PRIVATE
Treasurer of the United States.
SERIES 2006 A
Secretary of the Treasury.
FRANKLIN
ONE HUNDRED DOLLARS

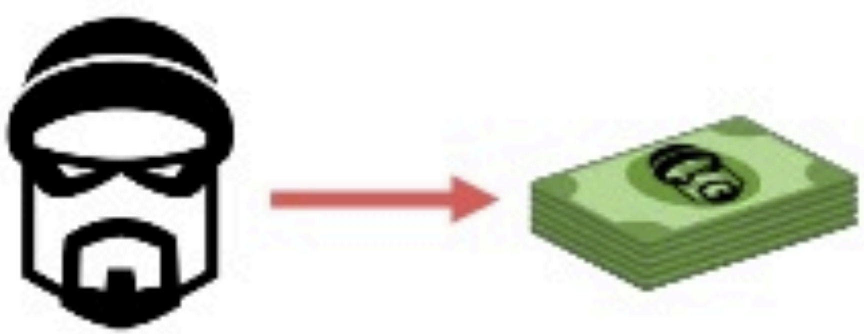
100
FAKE RESERVE NOTE
FF 9559F73A A
F6 COPY
THE UNITED SATES OF AMERICA
THIS IS NOT LEGAL TENDER
FOR MOVIE PROP USE ONLY
Im Not Real
Treasurer of the Treasurer
SERIES 2003 A
FOR MOVIE PROP USE ONLY
Im Not Real Currency
Secretary of the Secretary
ONE HUNDRED DOLLARS

↑↑ FAKE ↑↑

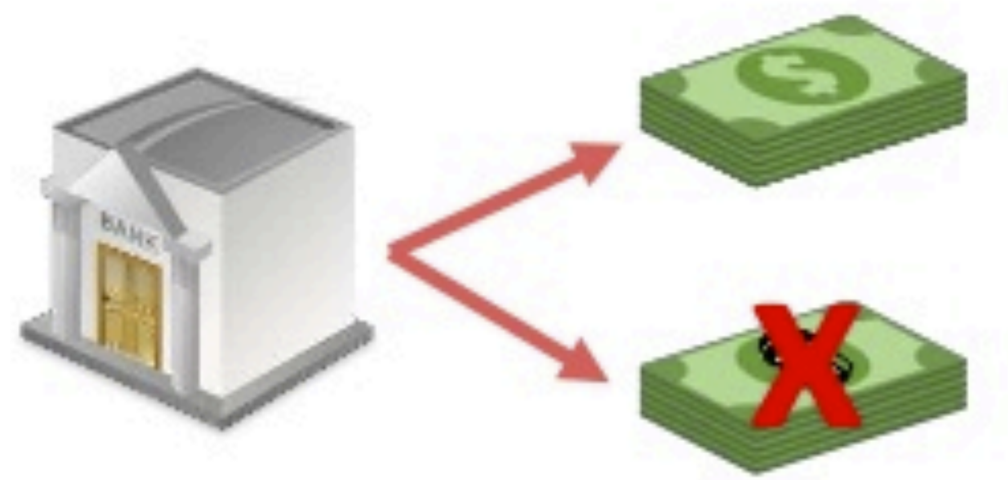
GENERATIVE ADVERSARIAL NETWORK

What are GANs?

First, an intuition

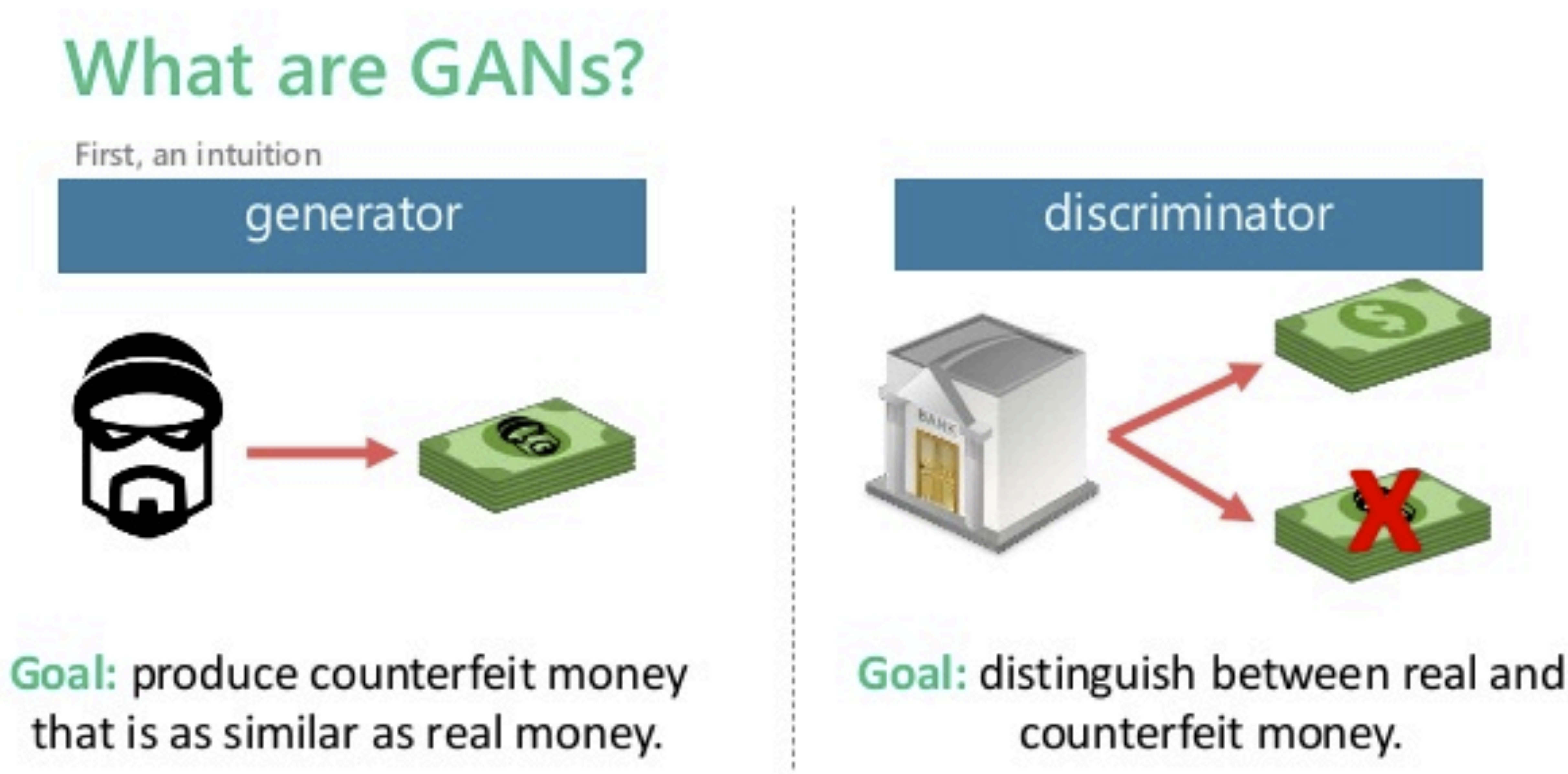


Goal: produce counterfeit money that is as similar as real money.

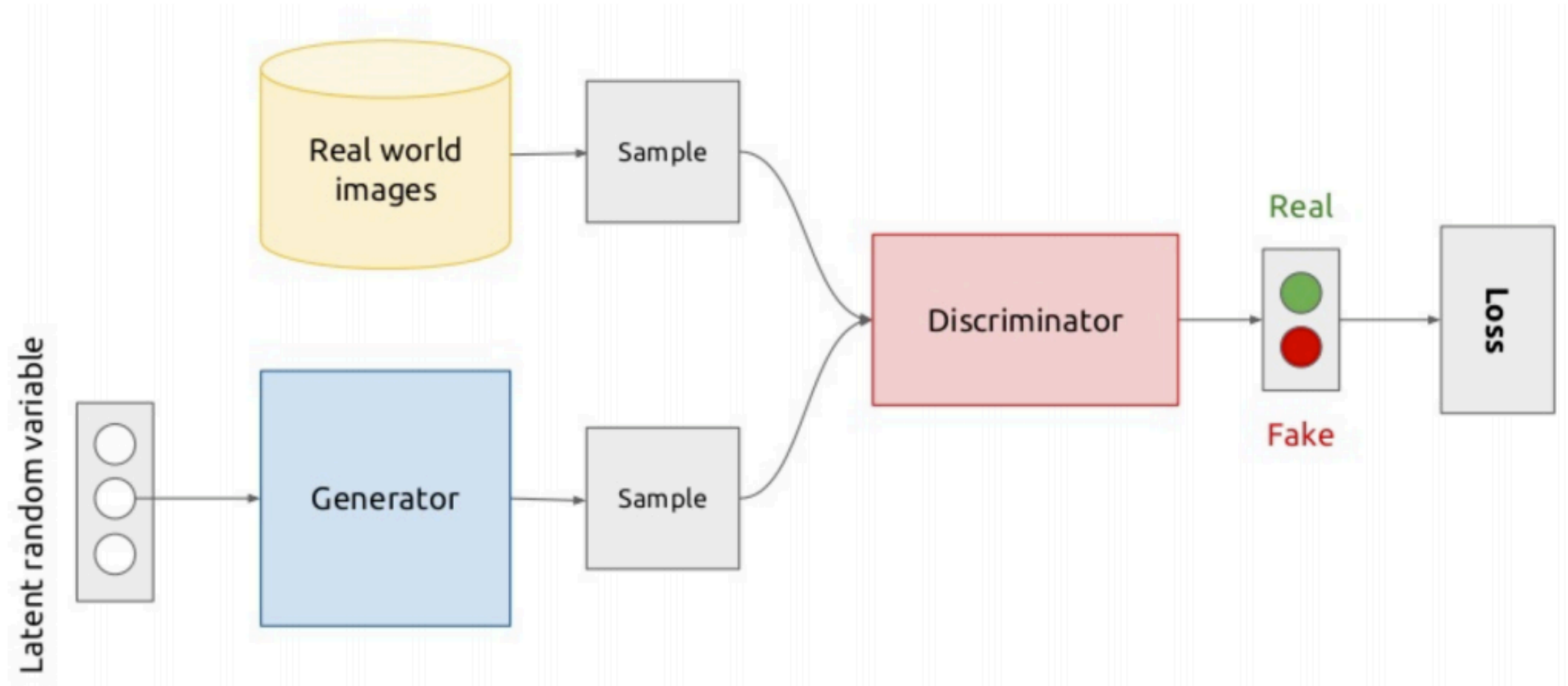


Goal: distinguish between real and counterfeit money.

GENERATIVE ADVERSARIAL NETWORK



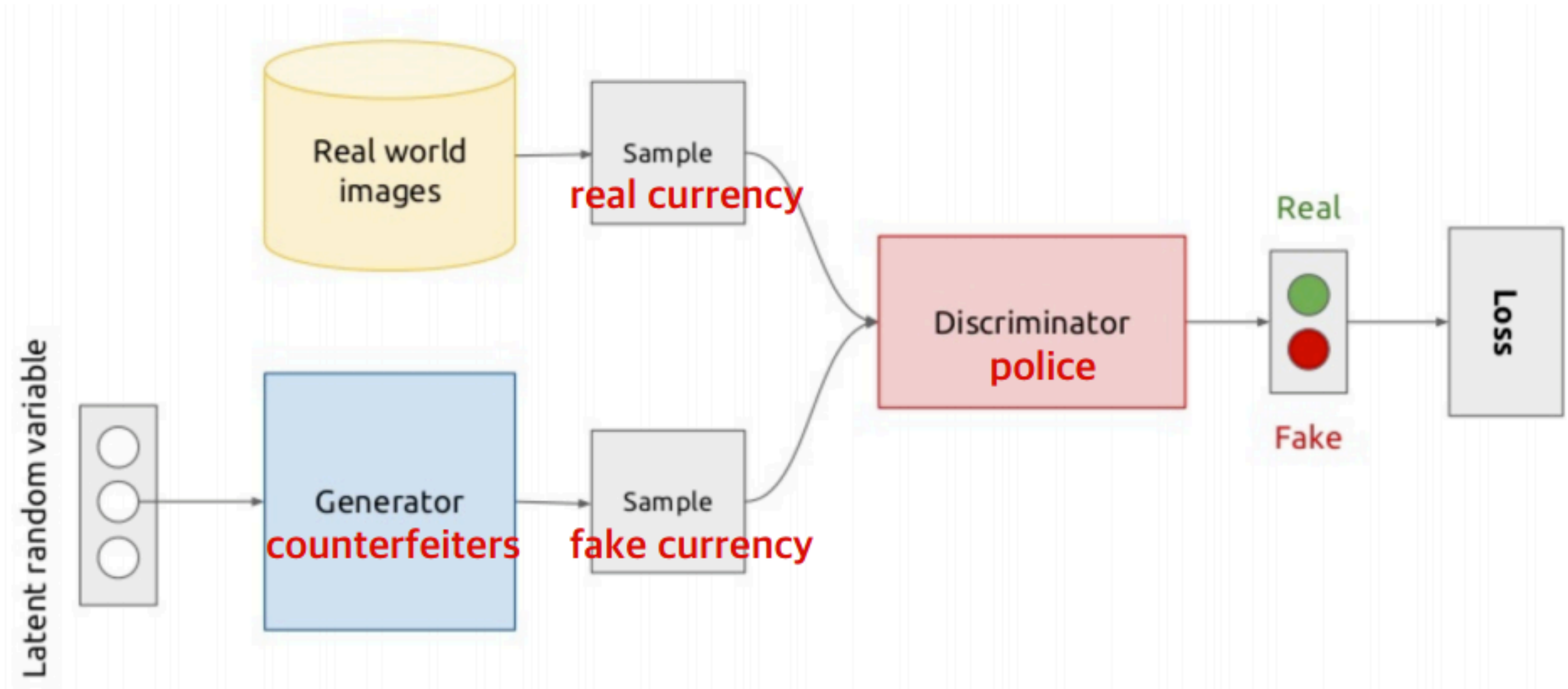
GENERATIVE ADVERSARIAL NETWORK



http://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

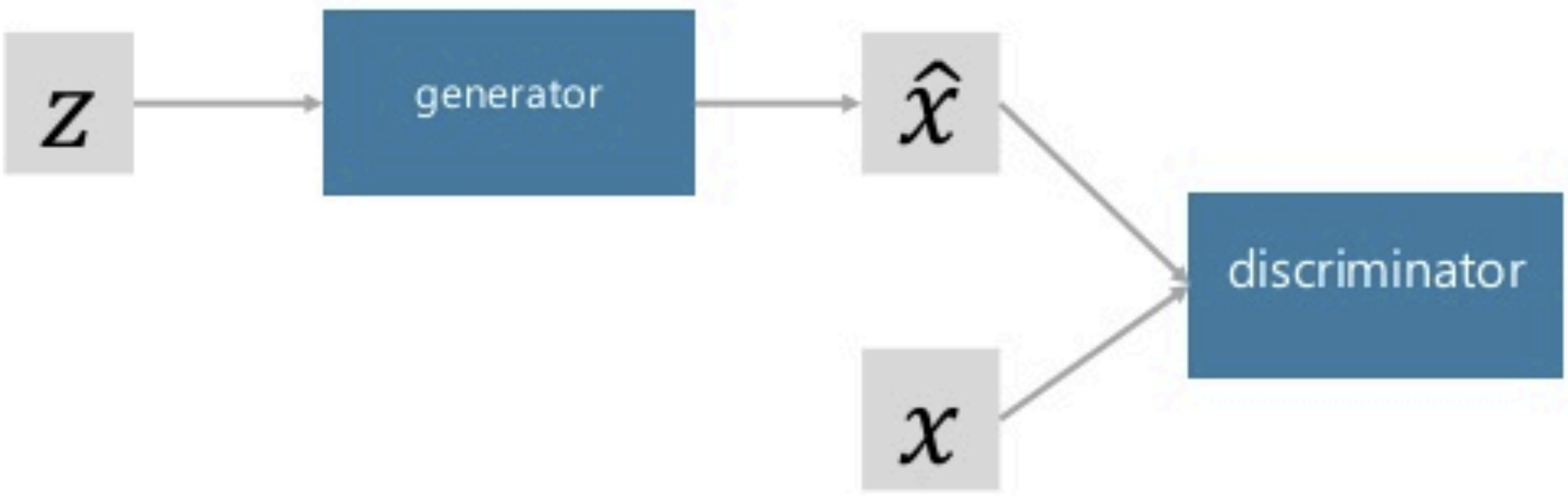
GENERATIVE ADVERSARIAL NETWORK

Adversarial Learning

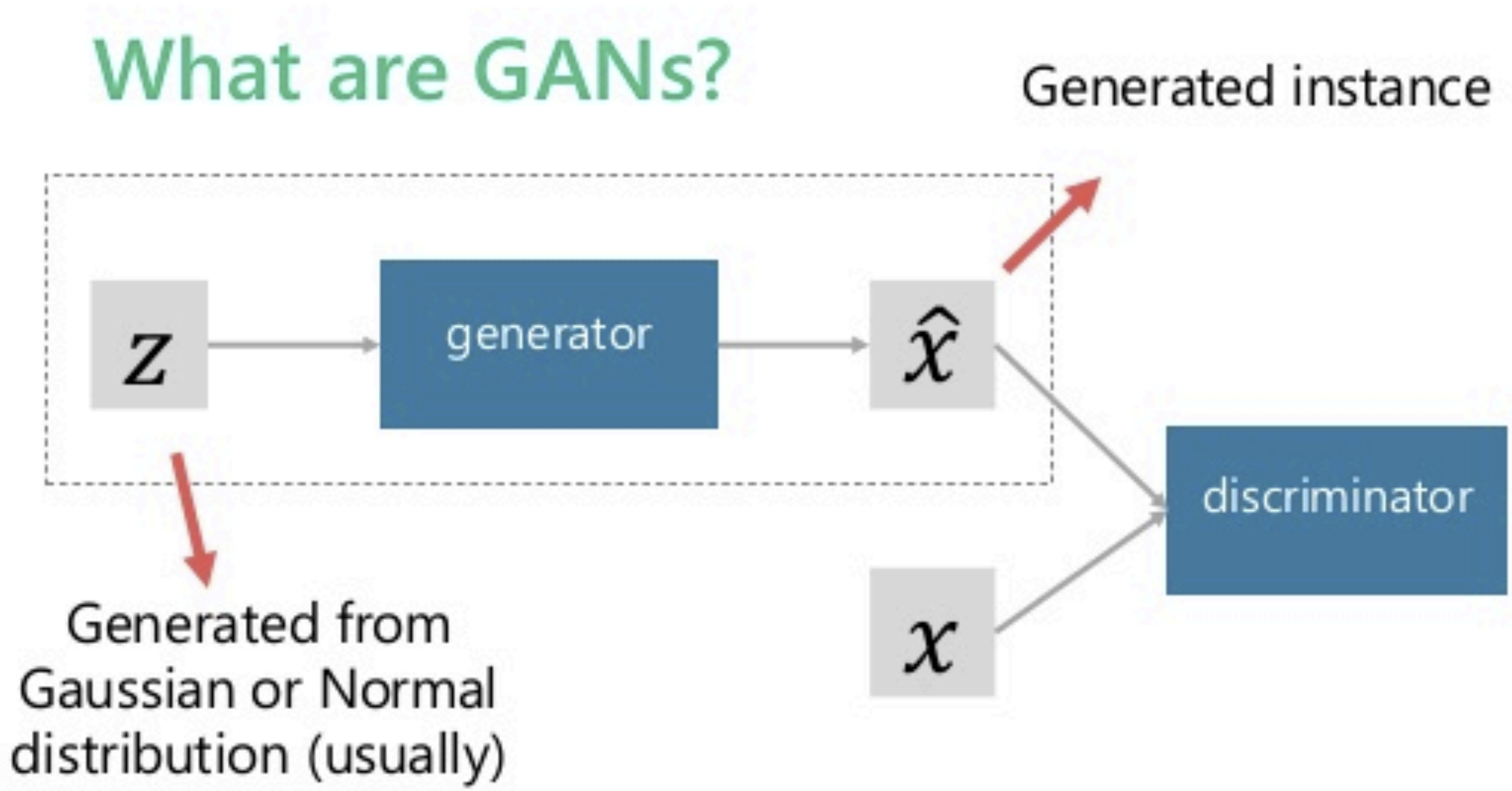


GENERATIVE ADVERSARIAL NETWORK

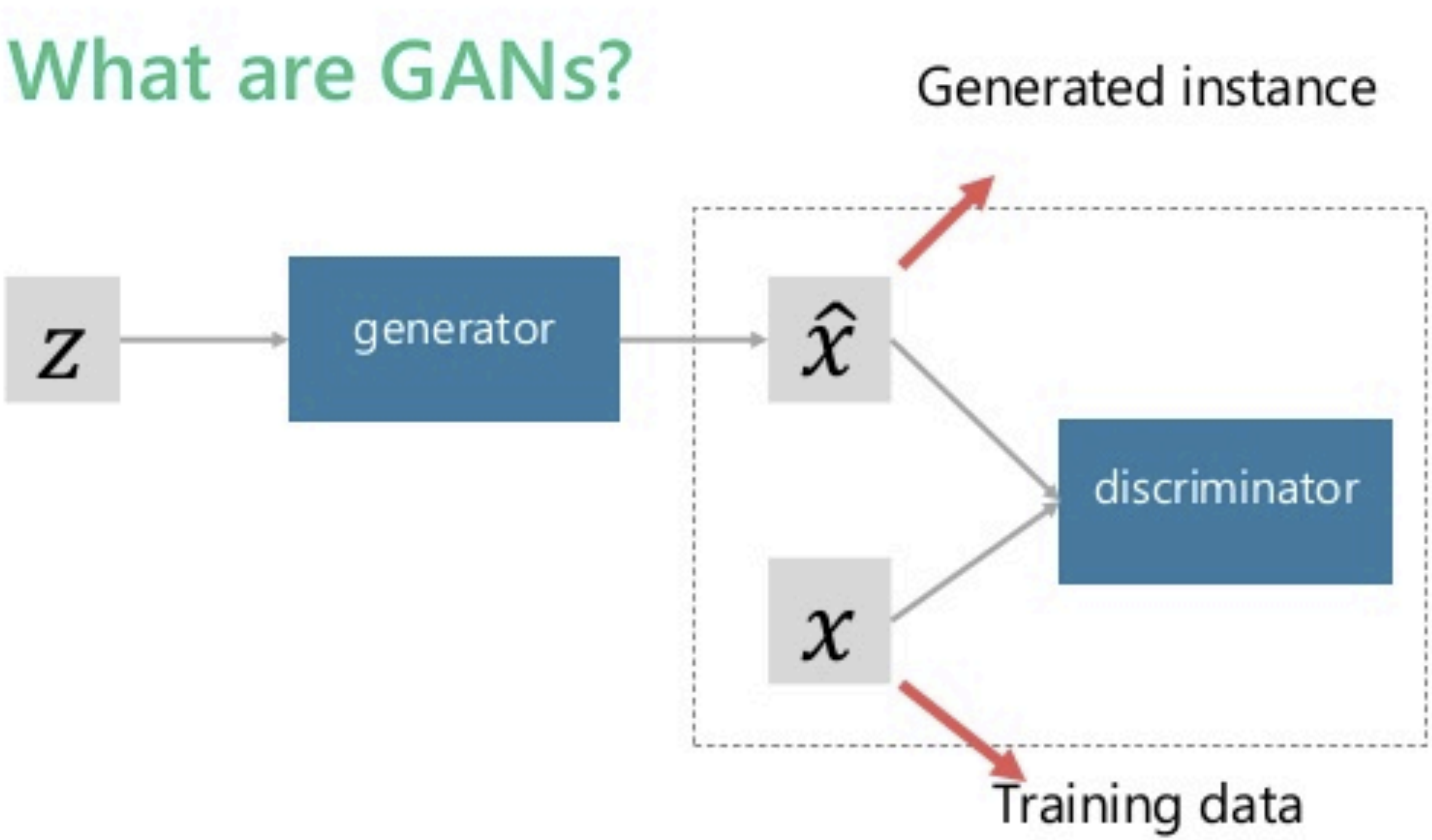
What are GANs?



GENERATIVE ADVERSARIAL NETWORK



GENERATIVE ADVERSARIAL NETWORK



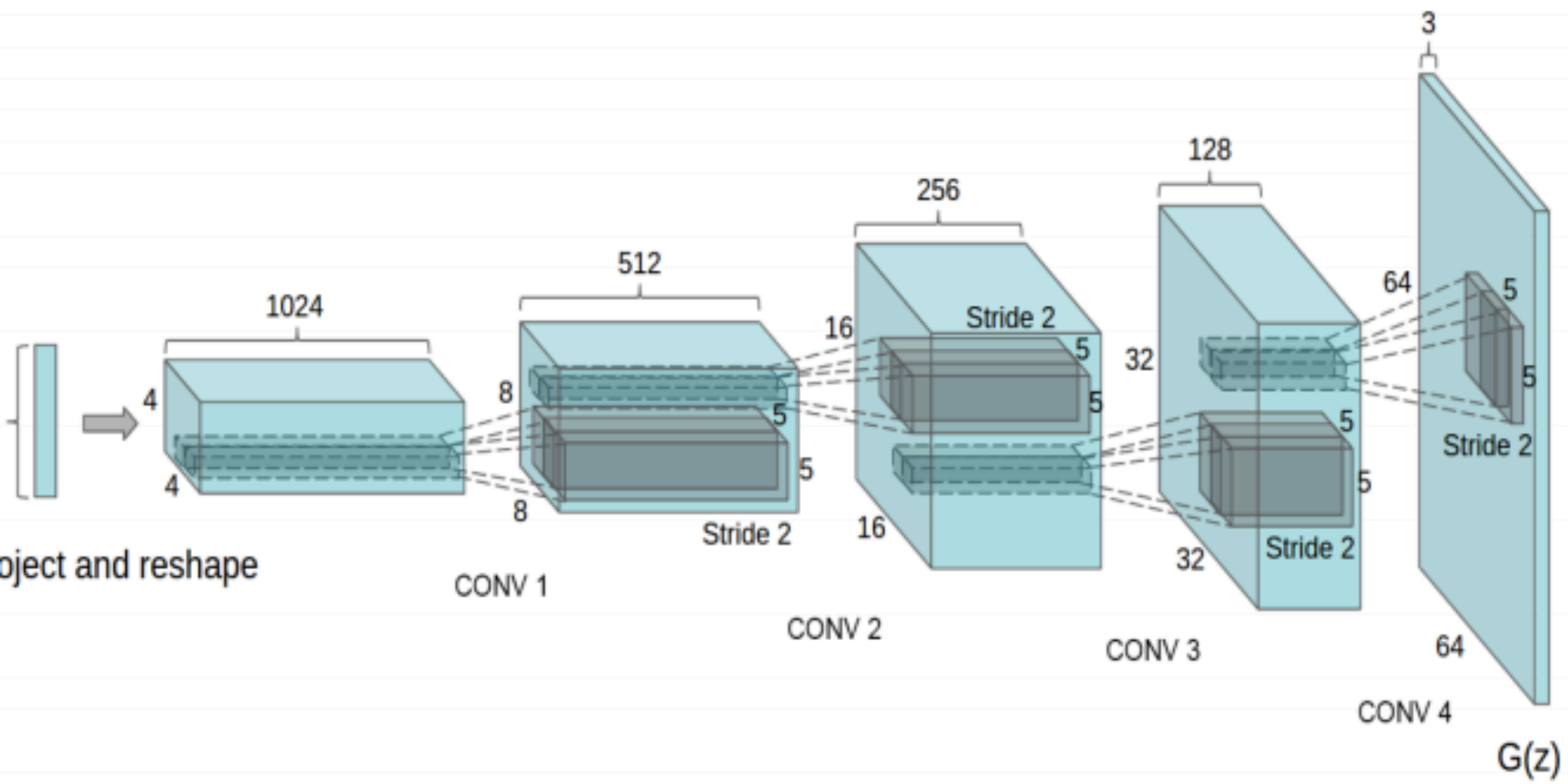
GENERATIVE ADVERSARIAL NETWORK

Generative Adversarial Networks

- Created by Ian Goodfellow (OpenAI);
- Two neural networks compete (*minmax game*)
  - Discriminative network tries to distinguish between real and fake data
  - Generative network tries to generate samples to fool the discriminative one
- Use latent code (z).

Deep Convolutional GANs (DCGANs)

Generator Architecture



Key ideas:

- Replace FC hidden layers with Convolutions
  - **Generator:** Fractional-Strided convolutions
- Use Batch Normalization after each layer
- **Inside Generator**
  - Use ReLU for hidden layers
  - Use Tanh for the output layer

GENERATIVE ADVERSARIAL NETWORK

Result

Image-to-Image Translation

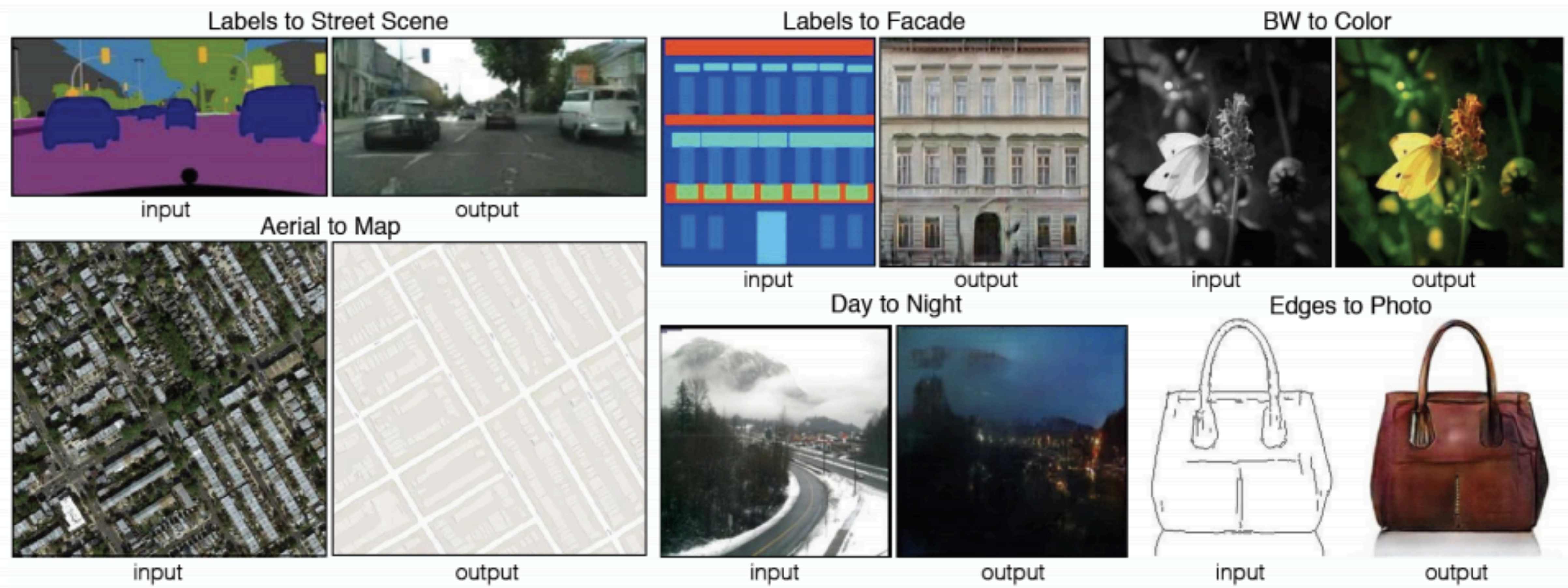


Figure 1 in the original paper.

Link to an interactive demo of this paper

Text-to-Image Synthesis

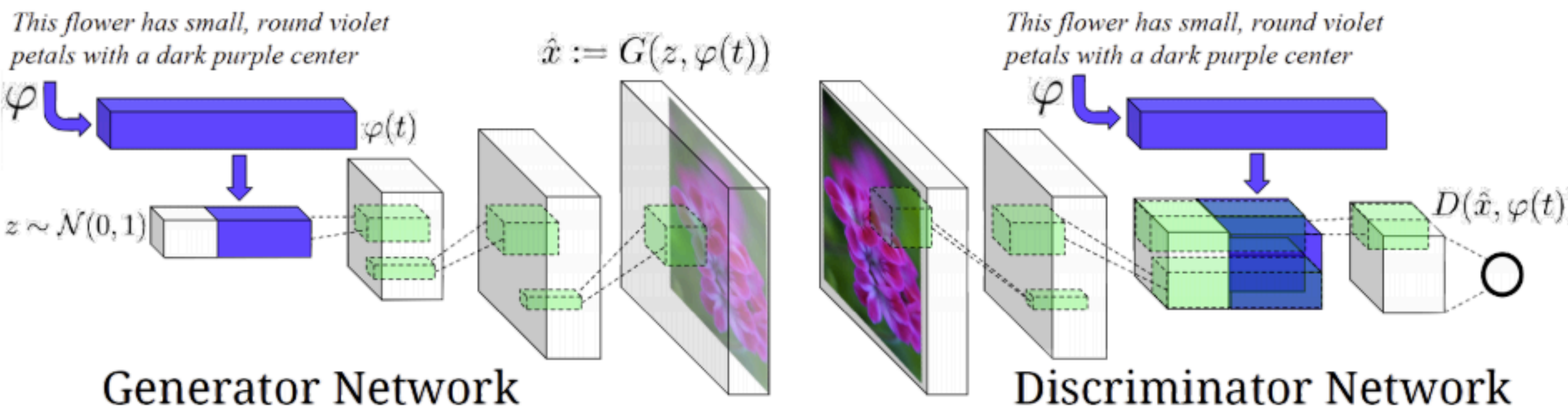


Figure 2 in the original paper.

Positive Example:

Real Image, Right Text

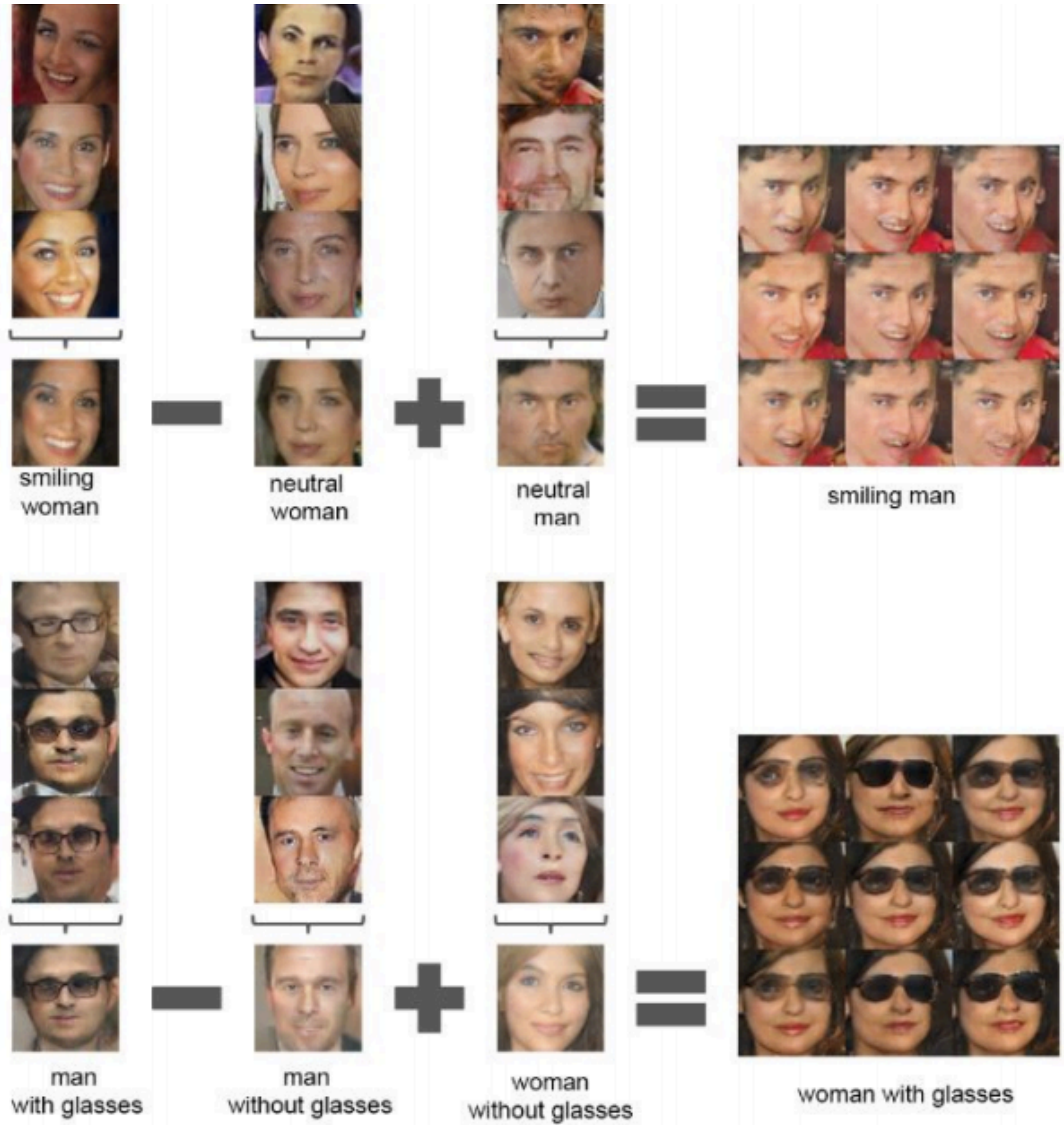
Negative Examples:

Real Image, Wrong Text

Fake Image, Right Text

GENERATIVE ADVERSARIAL NETWORK

DCGAN (15.11)



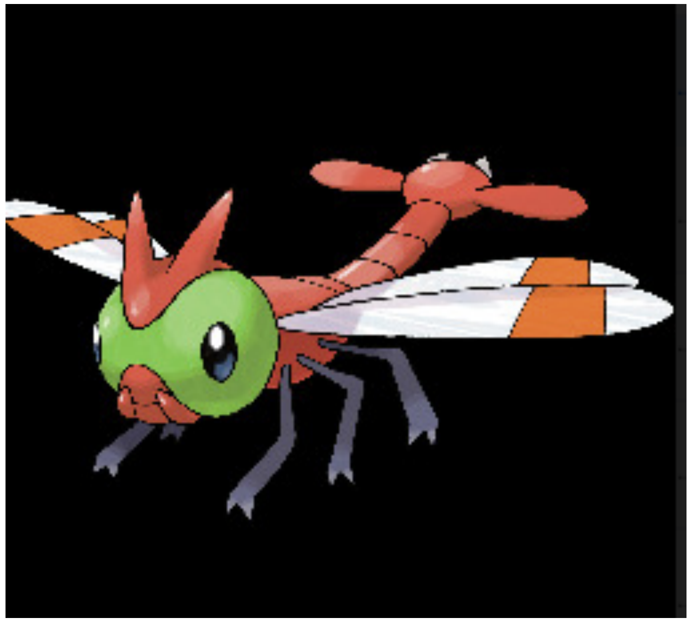
"It should happen until the generator exactly reproduces the true data distribution and the discriminator is guessing at random, unable to find a difference."

OpenAI

Source: https://openai.com/blog/generative-models/

GENERATIVE ADVERSARIAL NETWORK

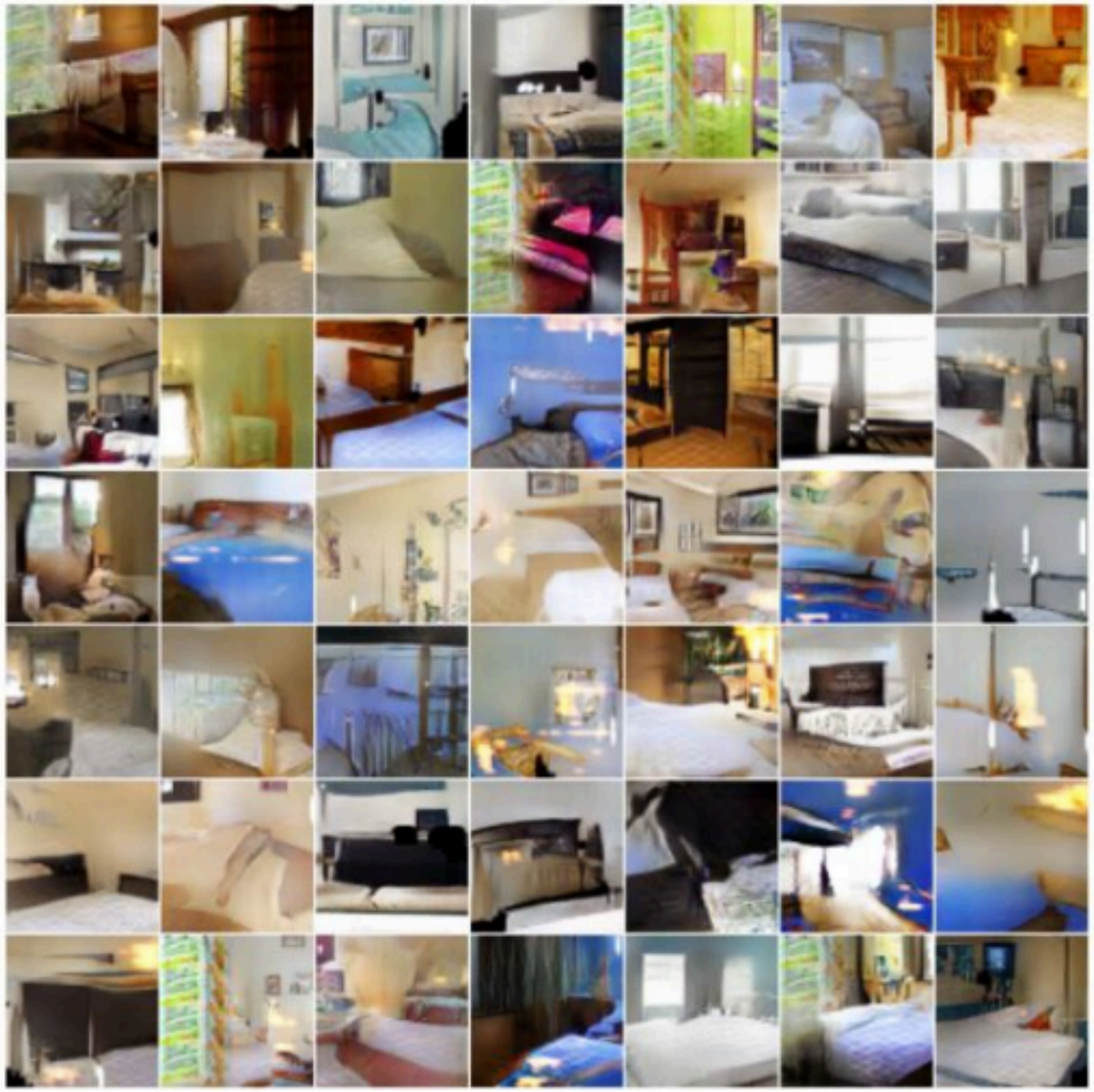


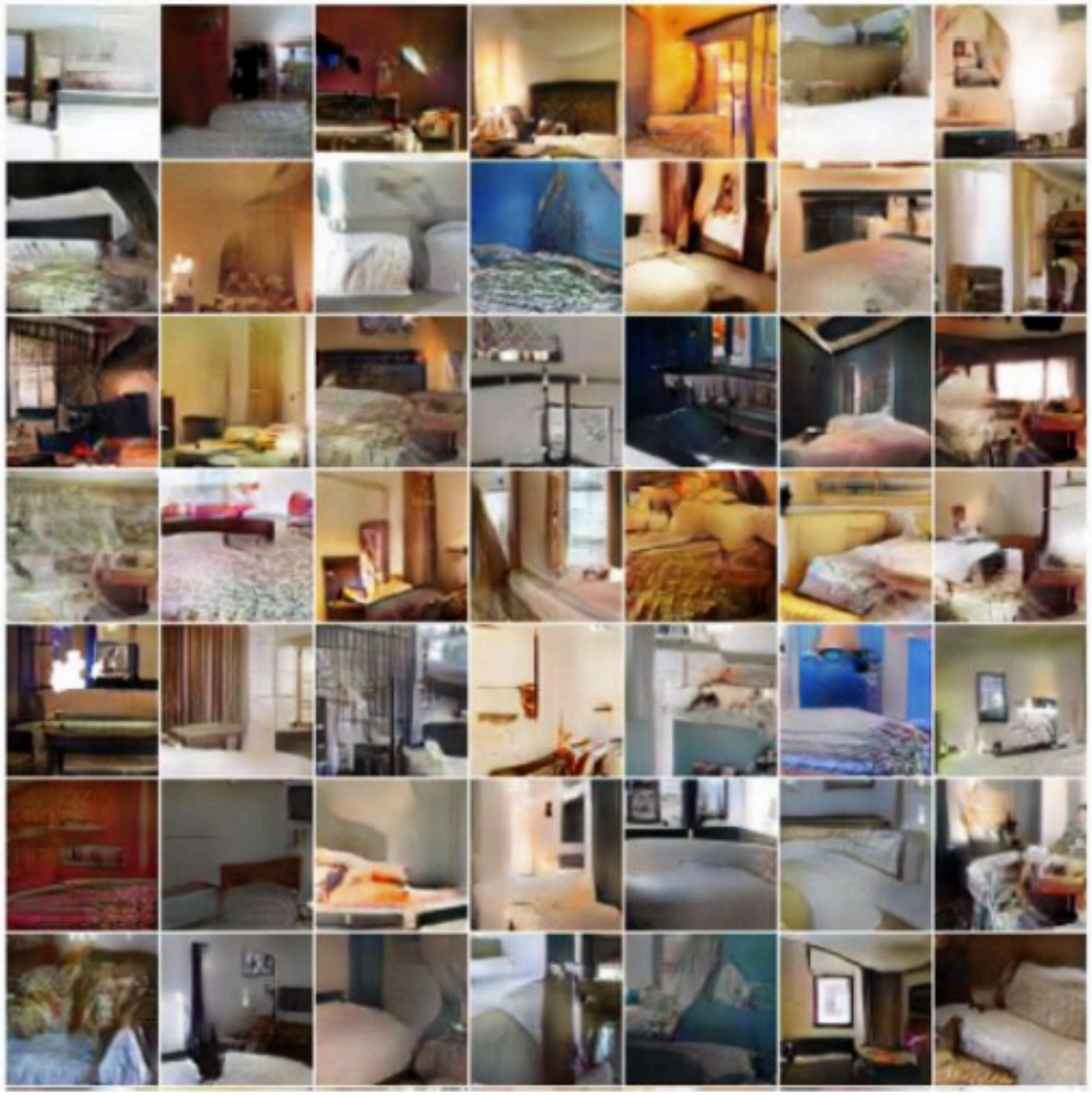



GENERATIVE ADVERSARIAL NETWORK

GENERATIVE ADVERSARIAL NETWORK

EBGAN (16.09)





GENERATIVE ADVERSARIAL NETWORK

Generative Adversarial Text to Image Synthesis

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran REEDSCOT[1], AKATA[2], XCYAN[1], LLAJAN[1]
Bernt Schiele, Honglak Lee SCHIELE[2],HONGLAK[1]
[1] University of Michigan, Ann Arbor, MI, USA (UMICH.EDU)
[2] Max Planck Institute for Informatics, Saarbrücken, Germany (MPI-INF.MPG.DE)

Abstract

Automatic synthesis of realistic images from text would be interesting and useful, but current AI systems are still far from this goal. However, in recent years generic and powerful recurrent neural network architectures have been developed to learn discriminative text feature representations. Meanwhile, deep convolutional generative adversarial networks (GANs) have begun to generate highly compelling images of specific categories, such as faces, album covers, and room interiors. In this work, we develop a novel deep architecture and GAN formulation to effectively bridge these advances in text and image modeling, translating visual concepts from characters to pixels. We demonstrate the capability of our model to generate plausible images of birds and flowers from detailed text descriptions.

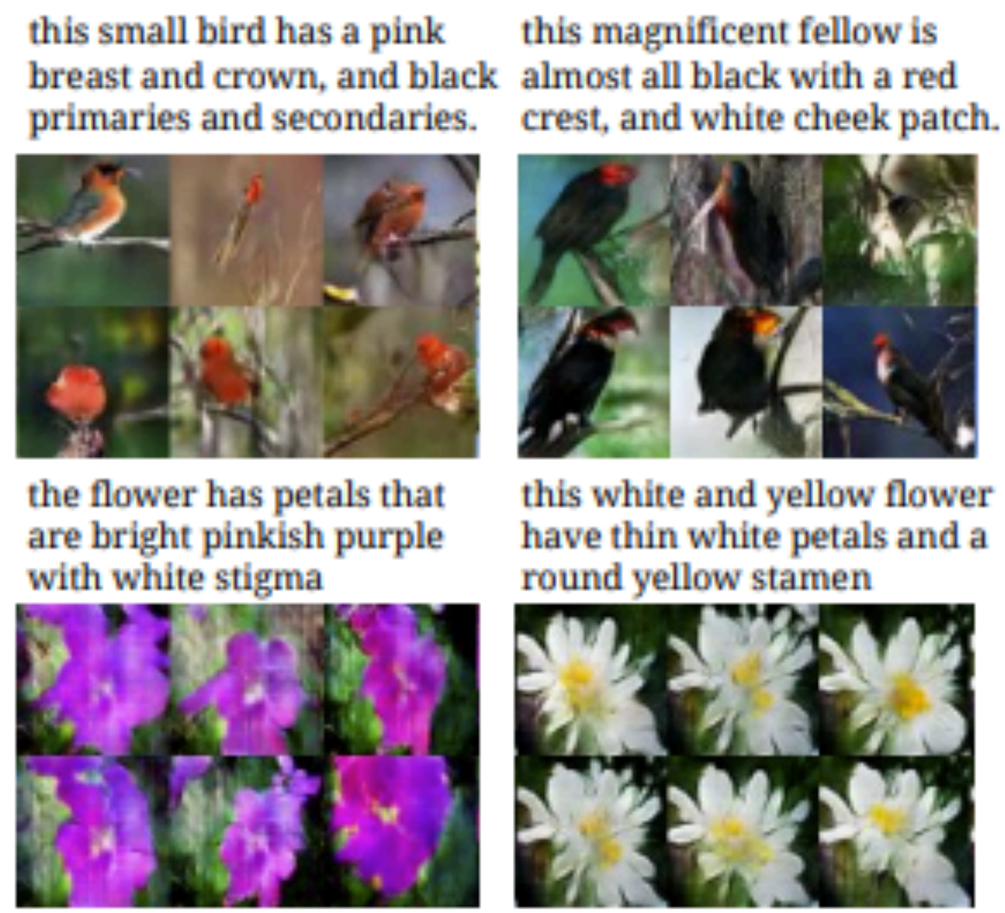
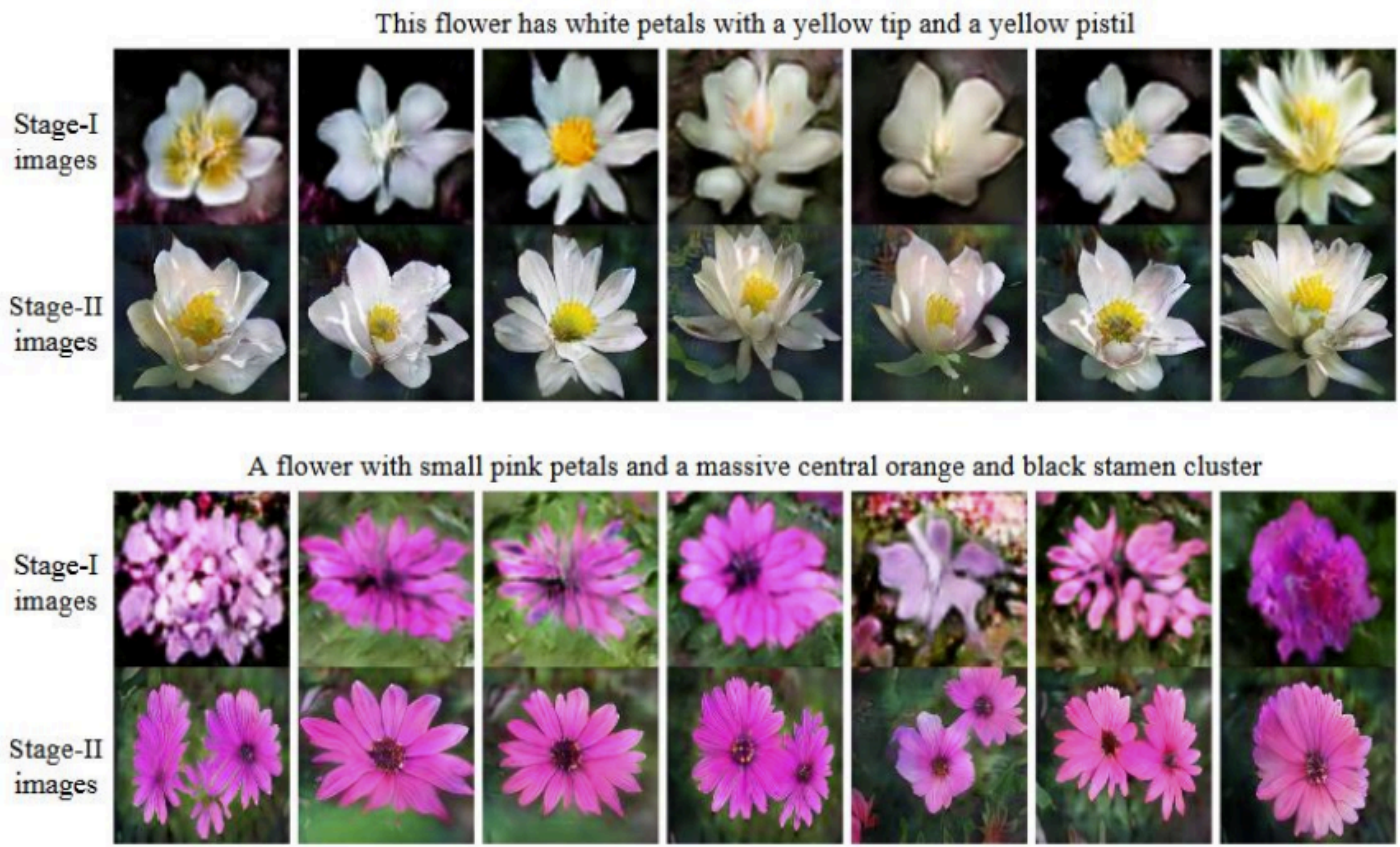


Figure 1. Examples of generated images from text descriptions. Left: captions are from zero-shot (held out) categories, unseen text. Right: captions are from the training set.

GENERATIVE ADVERSARIAL NETWORK

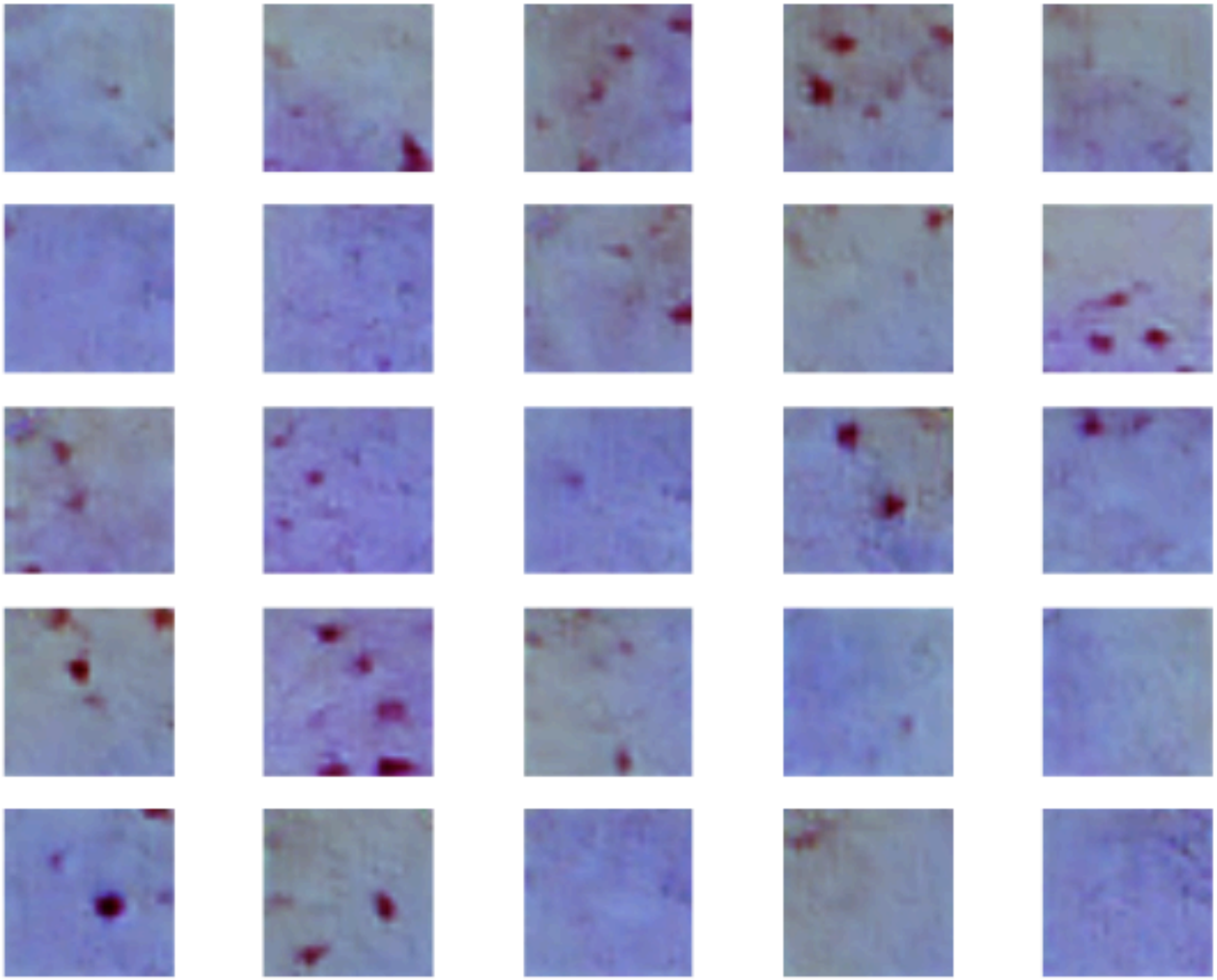
StackGAN (16.12)

StackGAN (16.12)



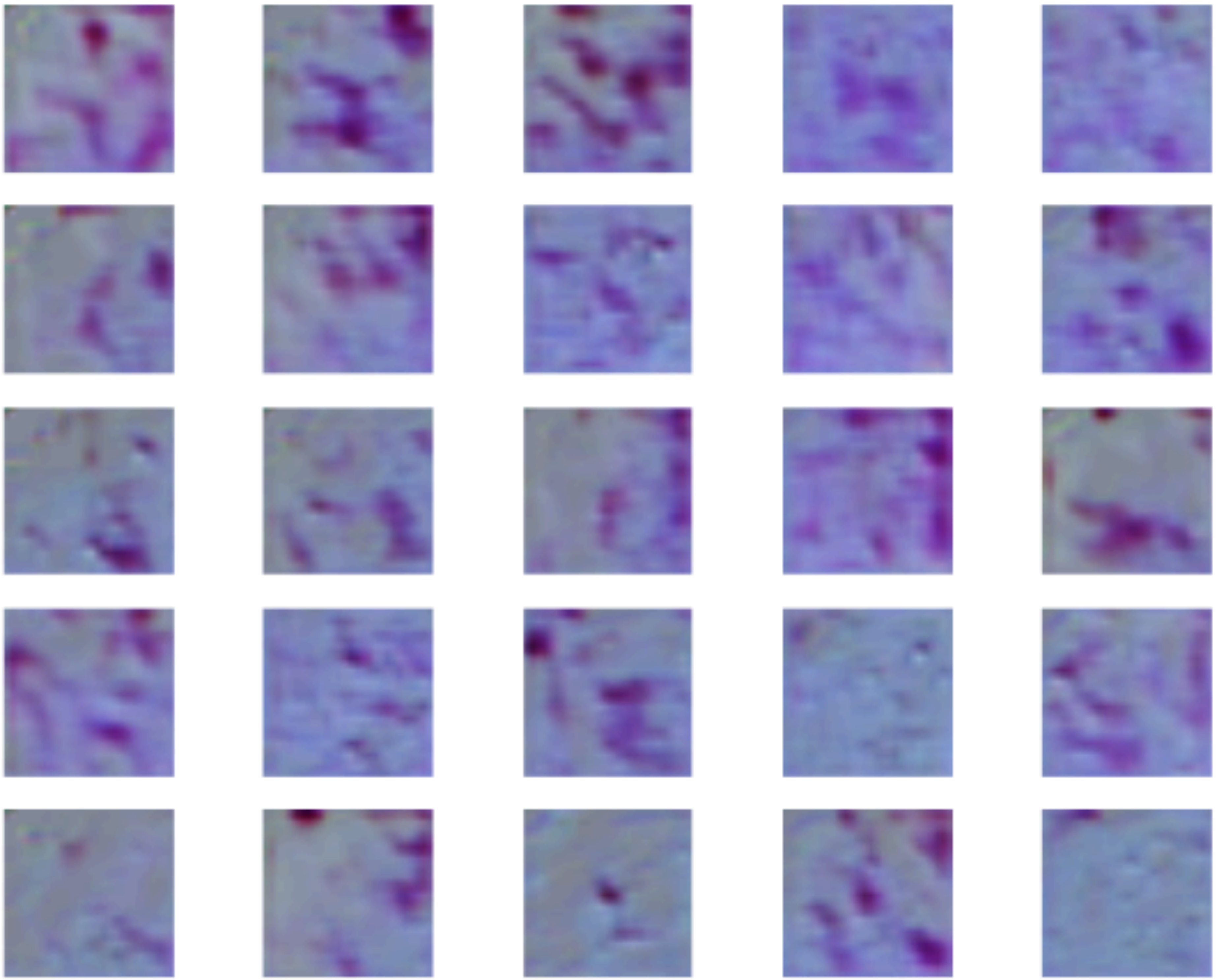
GENERATIVE ADVERSARIAL NETWORK

GENERATION 50



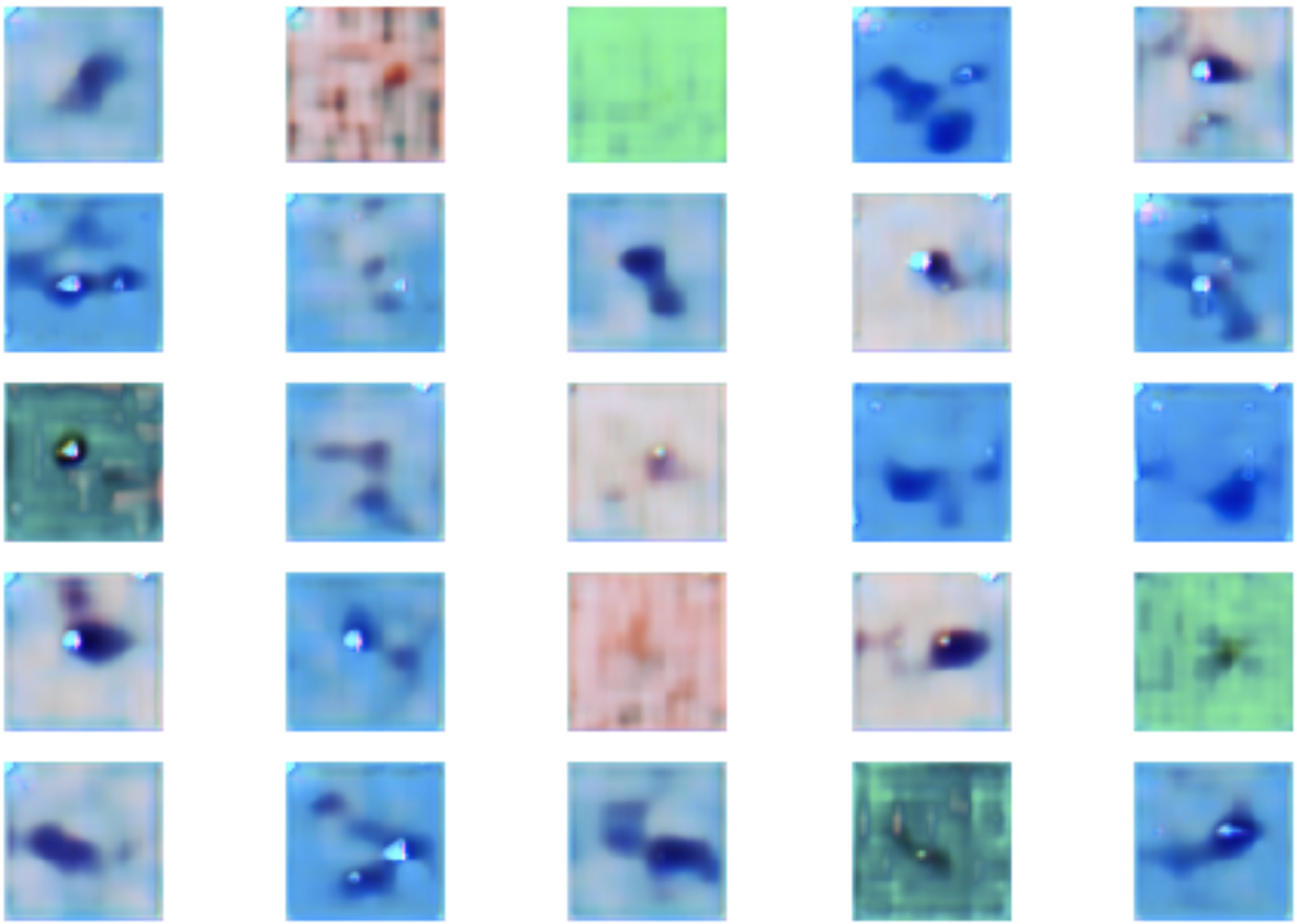
GENERATIVE ADVERSARIAL NETWORK

GENERATION 450



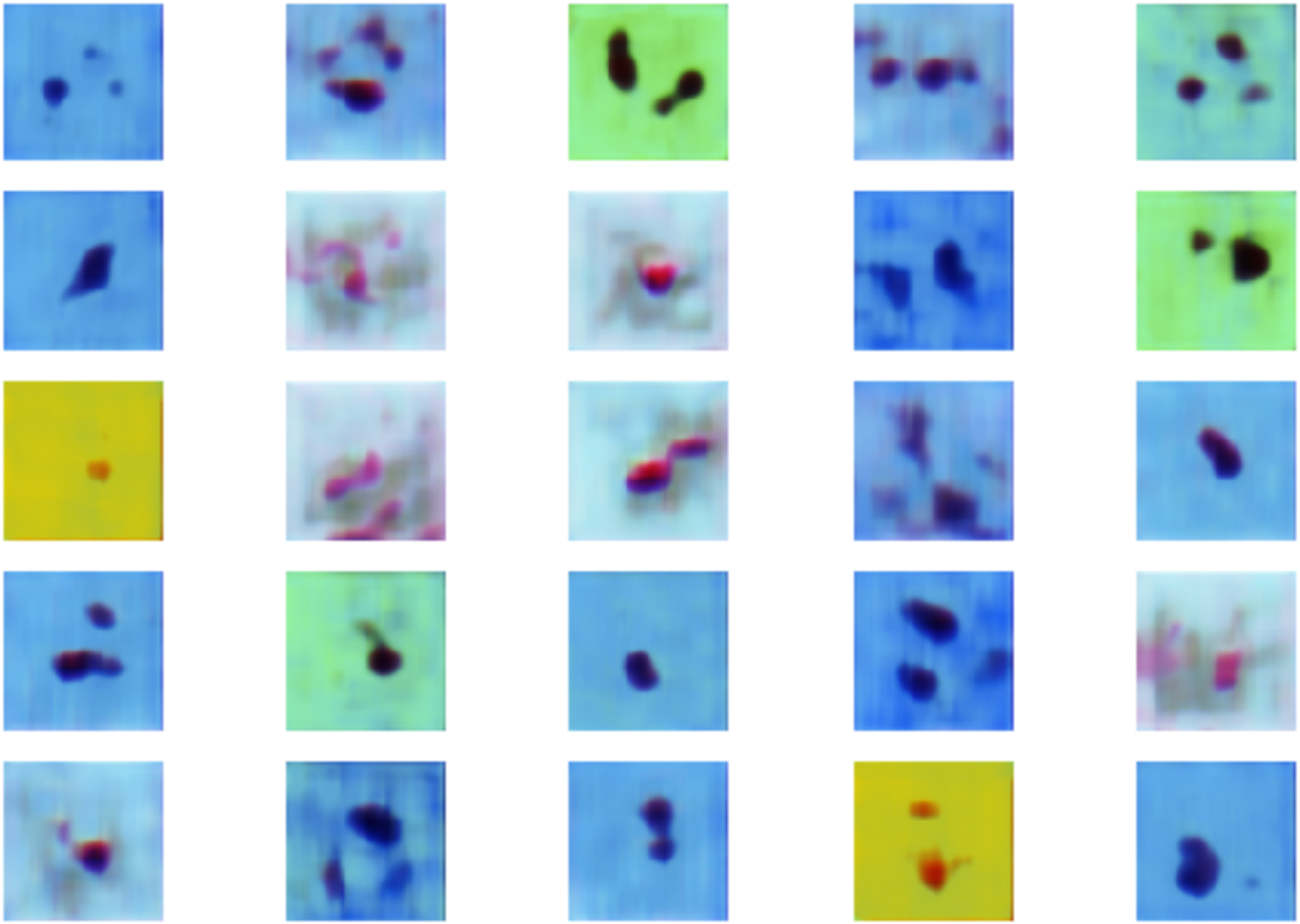
GENERATIVE ADVERSARIAL NETWORK

GENERATION 2000



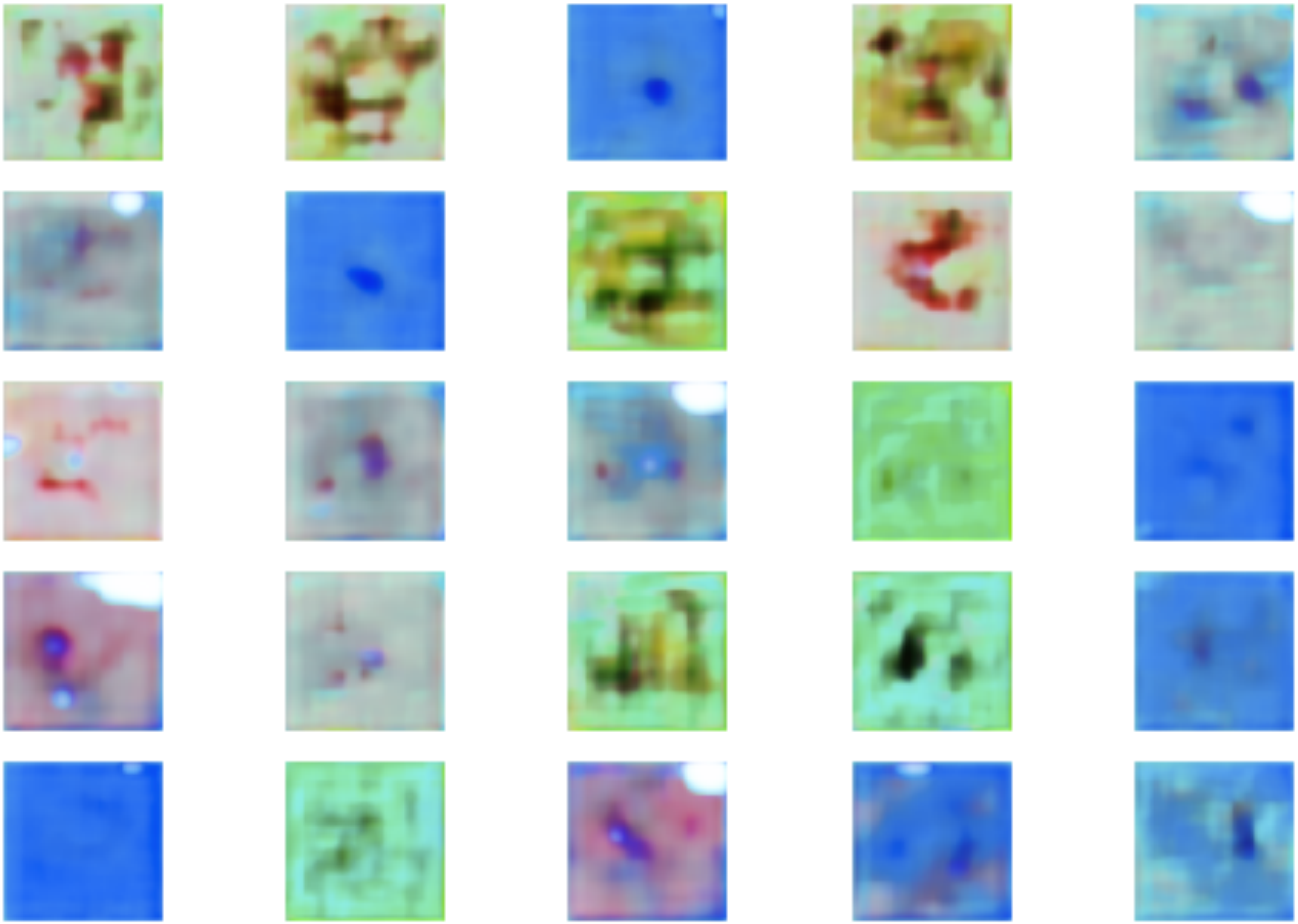
GENERATIVE ADVERSARIAL NETWORK

GENERATION 4000



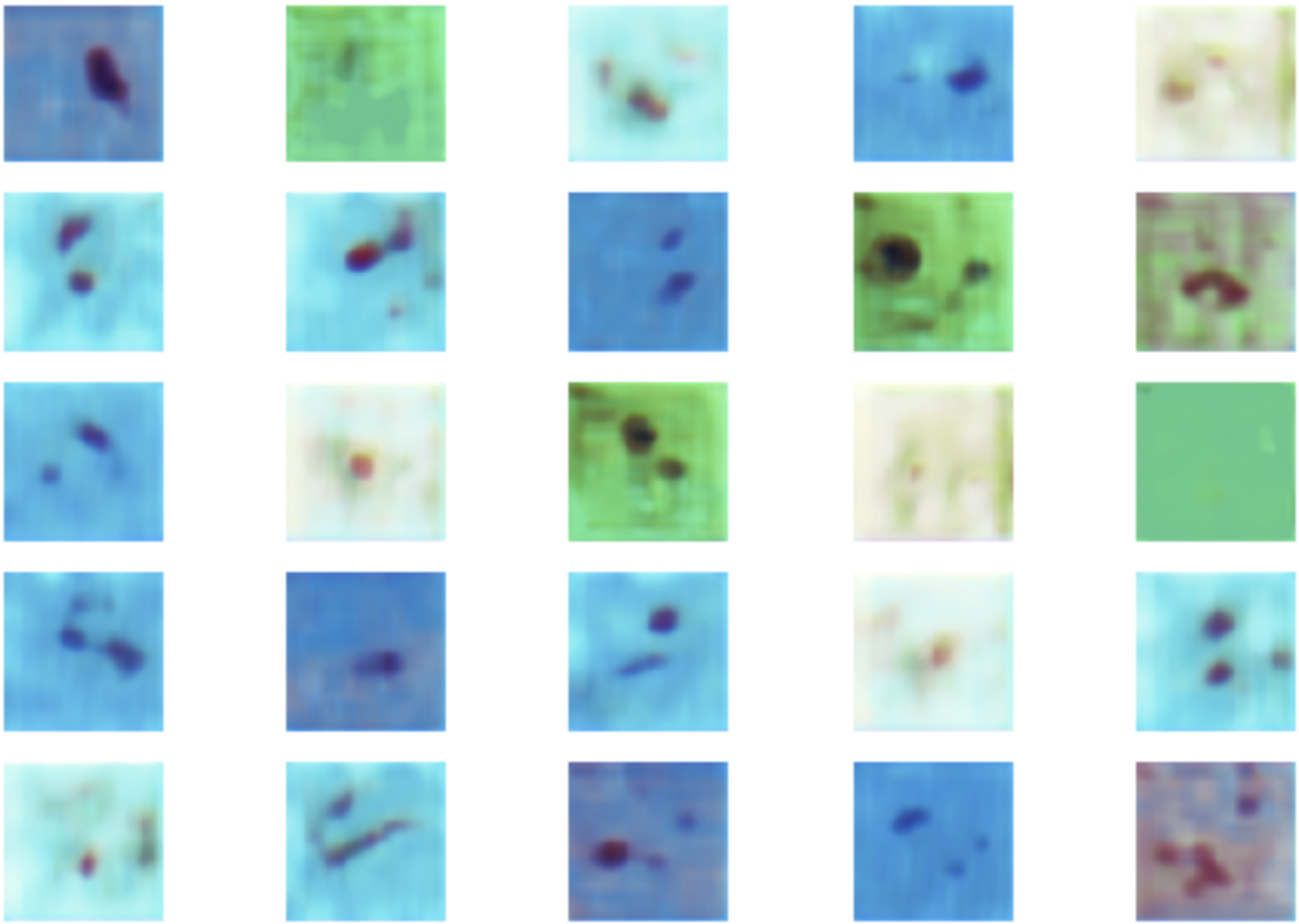
GENERATIVE ADVERSARIAL NETWORK

GENERATION 10000



GENERATIVE ADVERSARIAL NETWORK

GENERATION 26000



- **Deep Learning models (in general) involve a single player**
  - The player tries to maximize its reward (minimize its loss).
  - Use SGD (with Backpropagation) to find the optimal parameters.
  - SGD has convergence guarantees (under certain conditions).
  - **Problem:** With non-convexity, we might converge to local optima.

$$\min_{G} L(G)$$

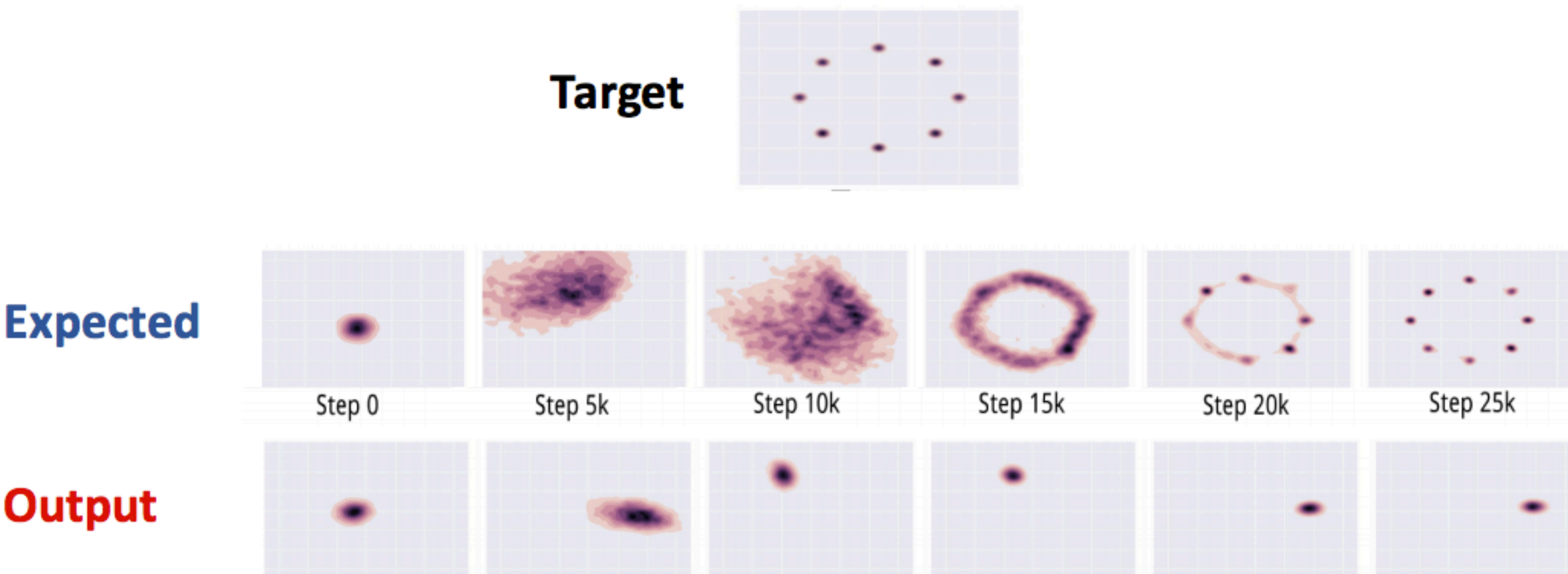
- **GANs instead involve two (or more) players**
  - Discriminator is trying to maximize its reward.
  - Generator is trying to minimize Discriminator's reward.

$$\min_{G} \max_{D} V(D, G)$$

  - SGD was not designed to find the Nash equilibrium of a game.
  - **Problem:** We might not converge to the Nash equilibrium at all.

Mode-Collapse

- **Generator fails to output diverse samples**



Metz, Luke, et al. **"Unrolled Generative Adversarial Networks."** arXiv preprint arXiv:1611.02163 (2016).