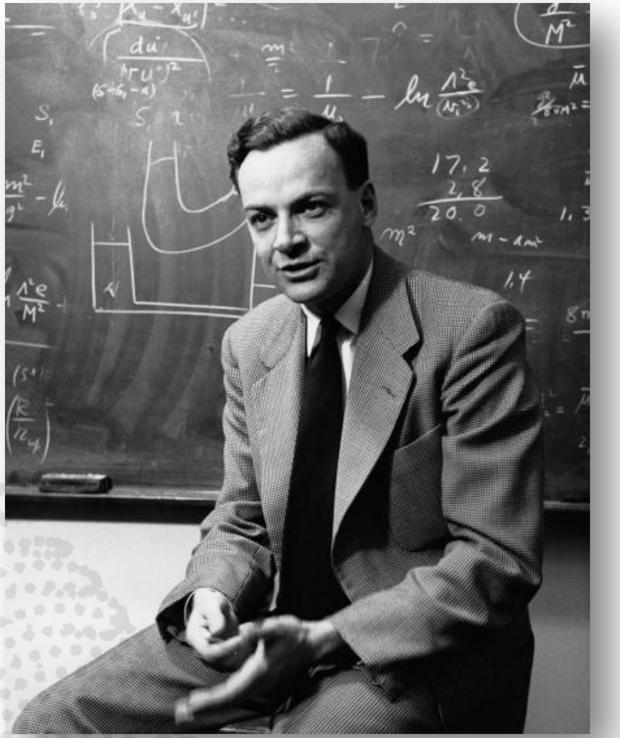
GAN gift of Imagination

WIEM CHAKROUN, AI Research engineering and data scientist at InstaDeep
w.chakroun@instaddep.com

Introduction - The limits of machine learning

- The common AI methods that are used in real life applications rely on huge labeled datasets, this kind of methods are called **Supervised Learning**.
- Human Intelligence is also about **creativity** and **abstraction**.

"What I cannot create, I do not understand."
-- Richard Feynman



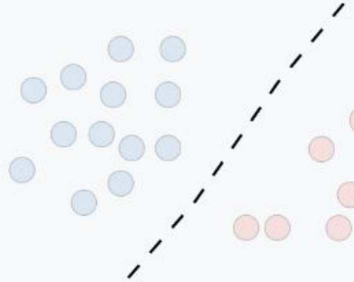
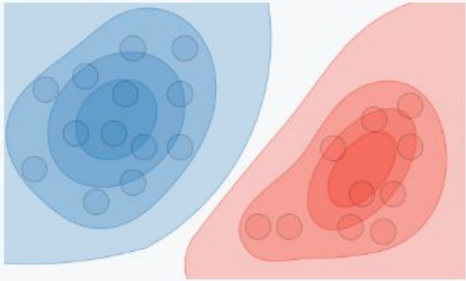
Unsupervised Learning

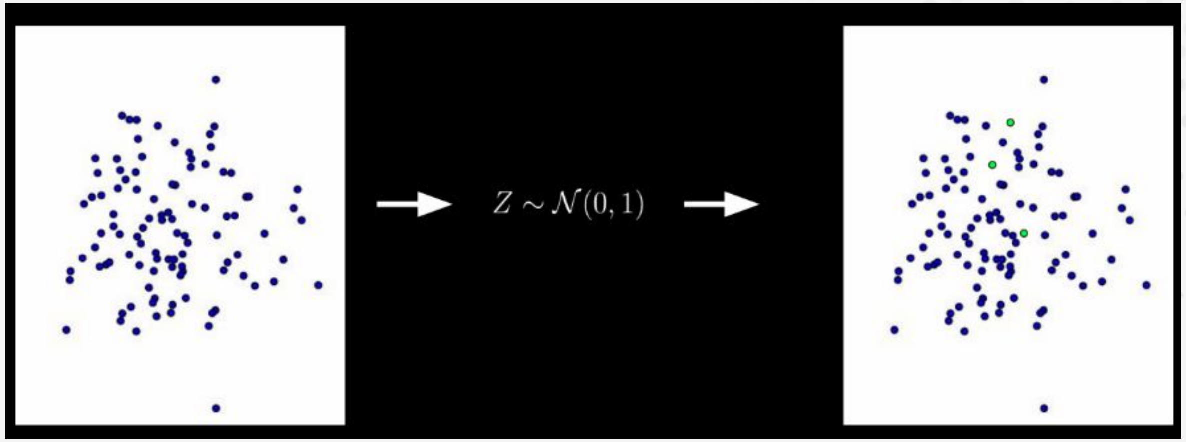
- No labels are provided during training
- Find a function that describes the hidden structure from unlabelled data:
  - Density Estimation
  - Clustering
  - Feature Learning
  - Dimensionality Reduction

InstaDeep™

Generative Models

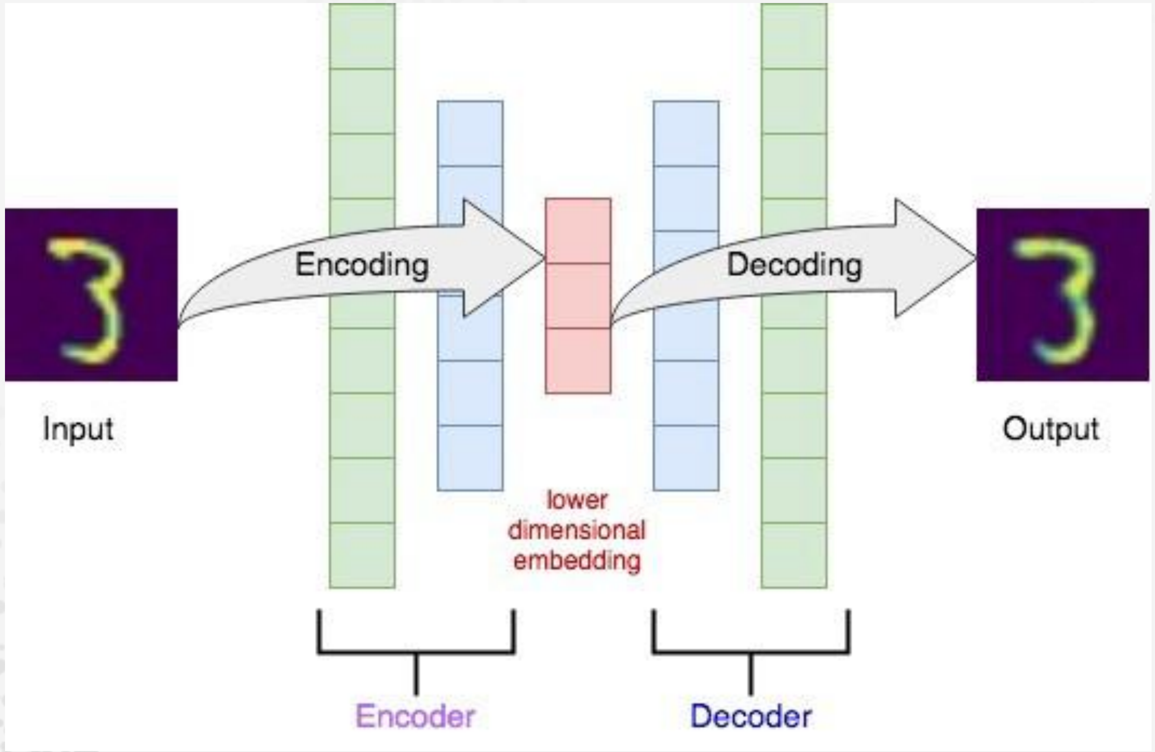
- Generate new samples follows the same probabilistic distribution of a given a training dataset.

	Discriminative model	Generative model
Goal	Directly estimate $P(y\|x)$	Estimate $P(x\|y)$ to then deduce $P(y\|x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GDA, Naive Bayes



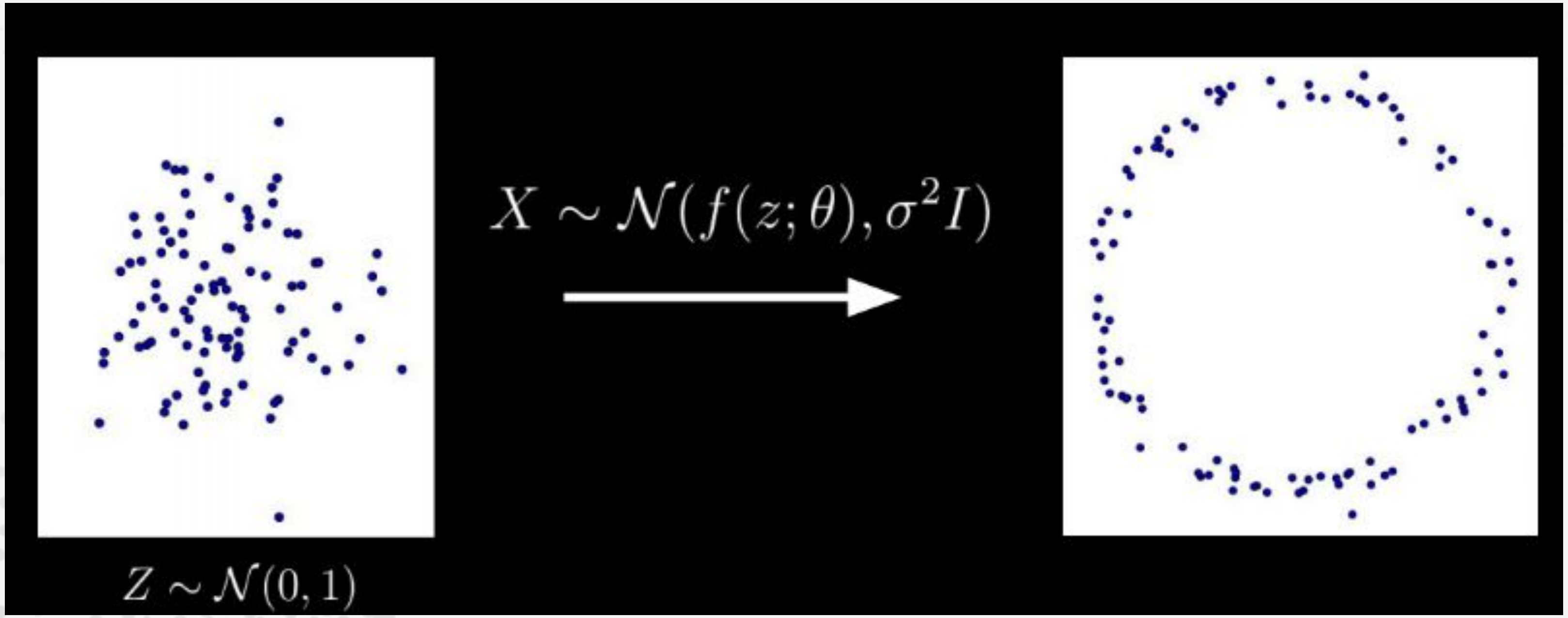
Auto-Encoders

- A neural network where the output is the input itself.

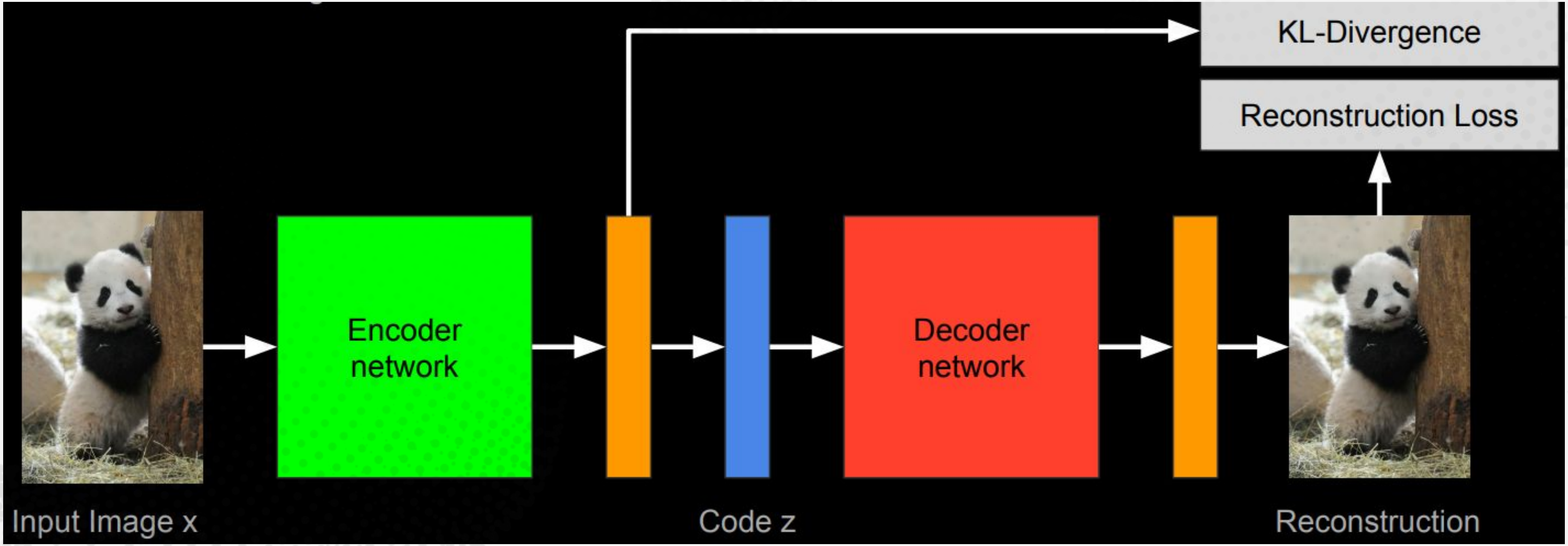


Variational Auto-Encoders

- Given a bunch of random variables that can be sampled easily, we can generate random samples following other distributions, through a complicated non-linear mapping x = f(z).

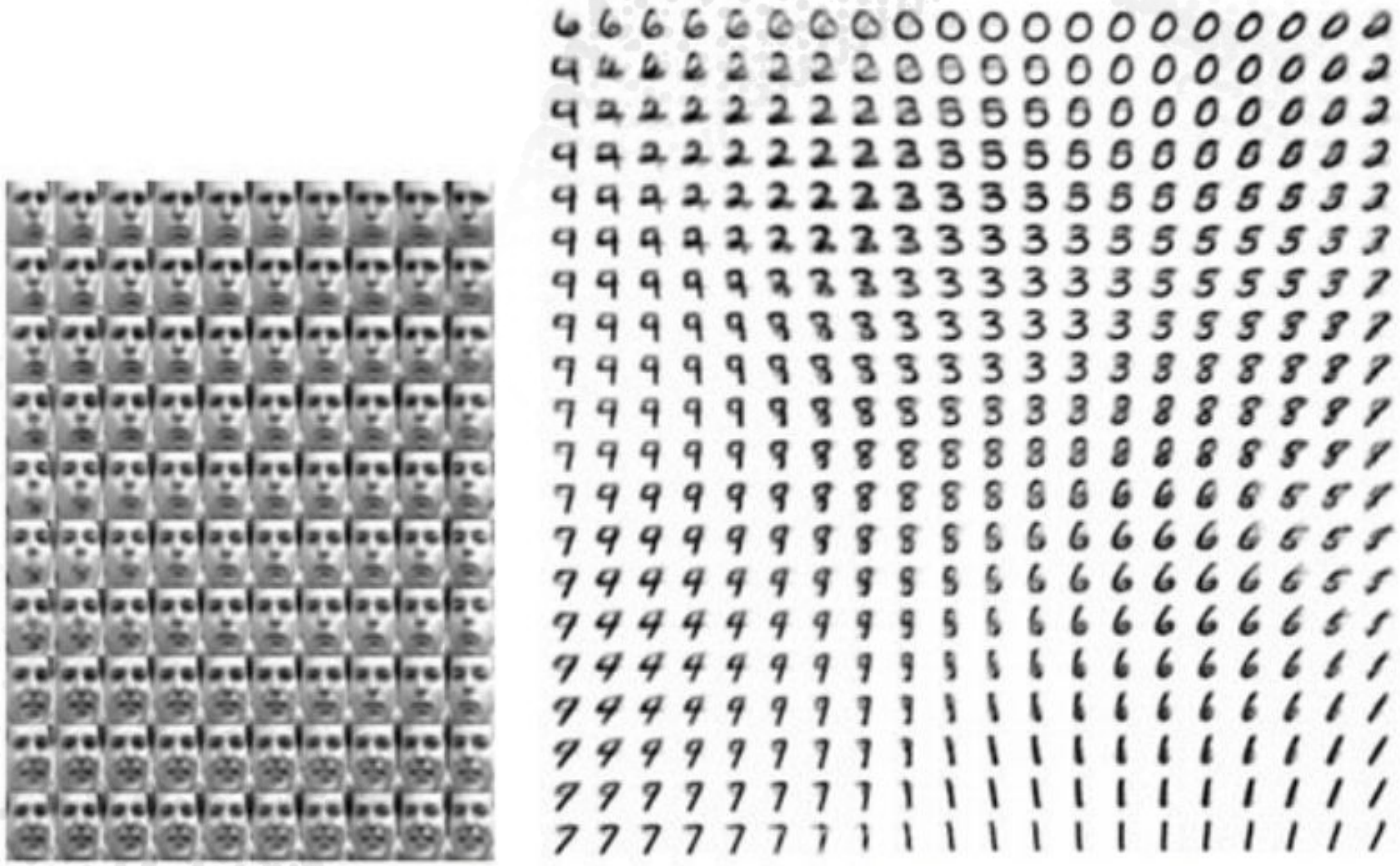


Variational Auto-Encoders



InstaDeep™

Variational Auto-Encoders

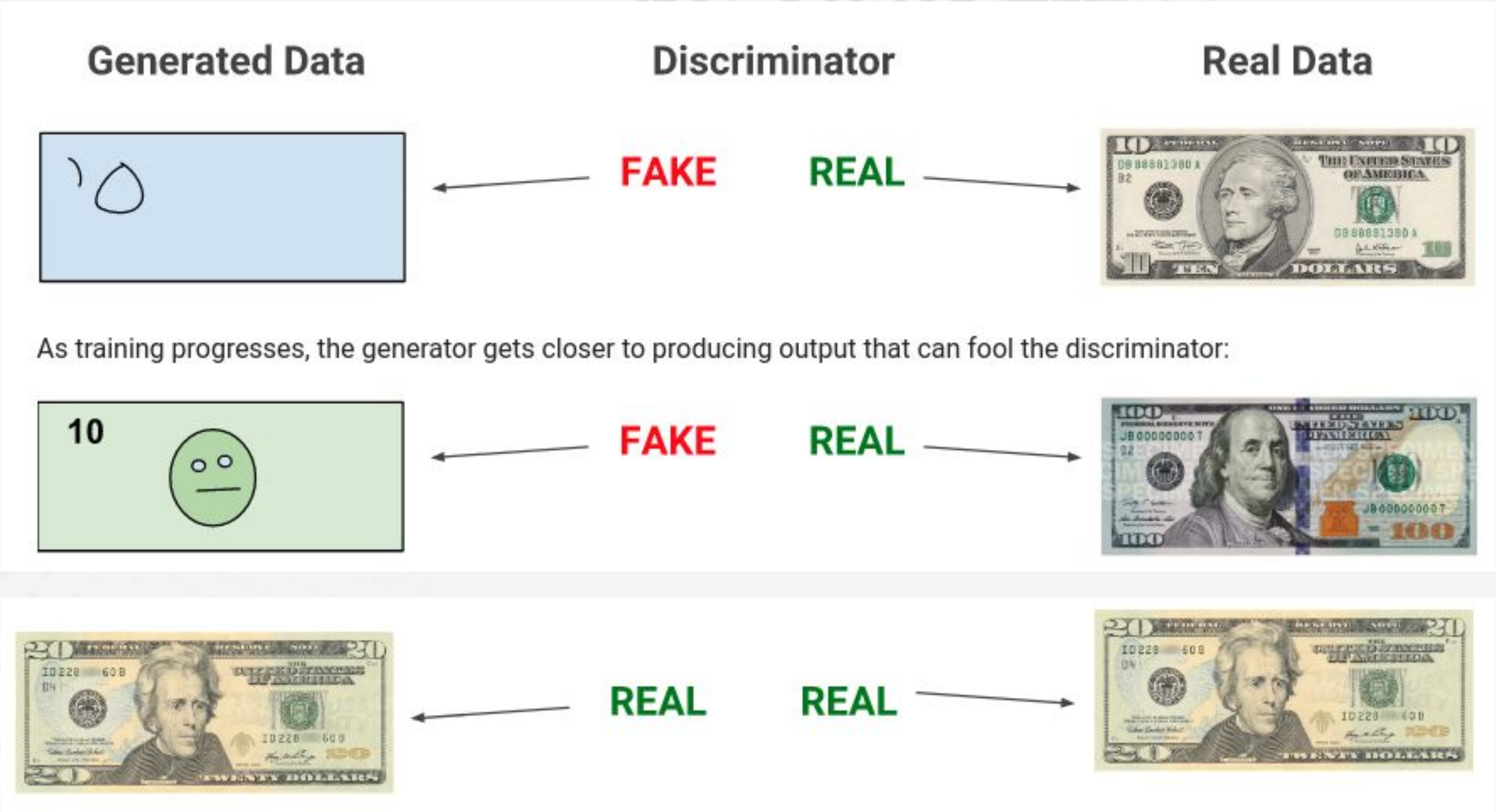


(a) Learned Frey Face manifold (b) Learned MNIST manifold

Generative Adversarial Networks - History

GAN

Overview of GAN Structure



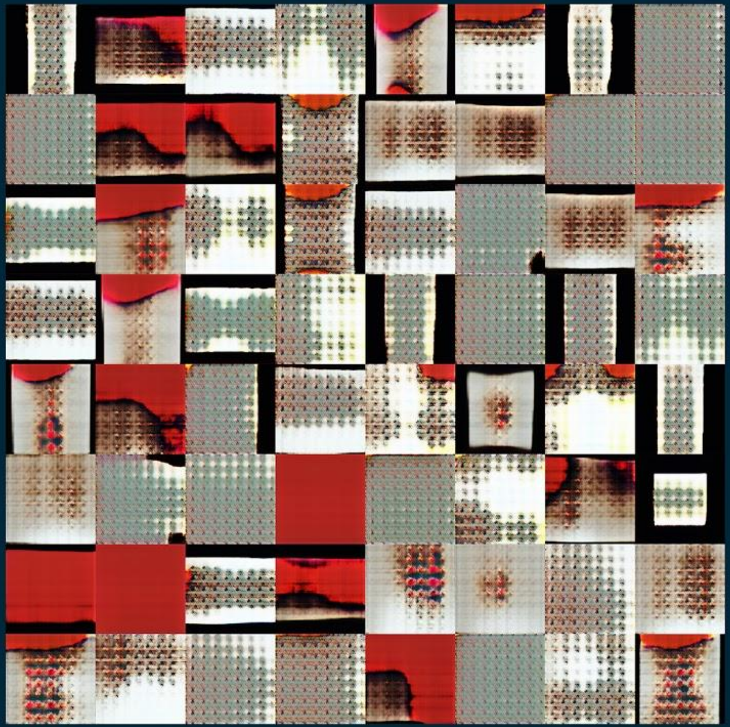
InstaDeep™

Generative Adversarial Networks (GAN)

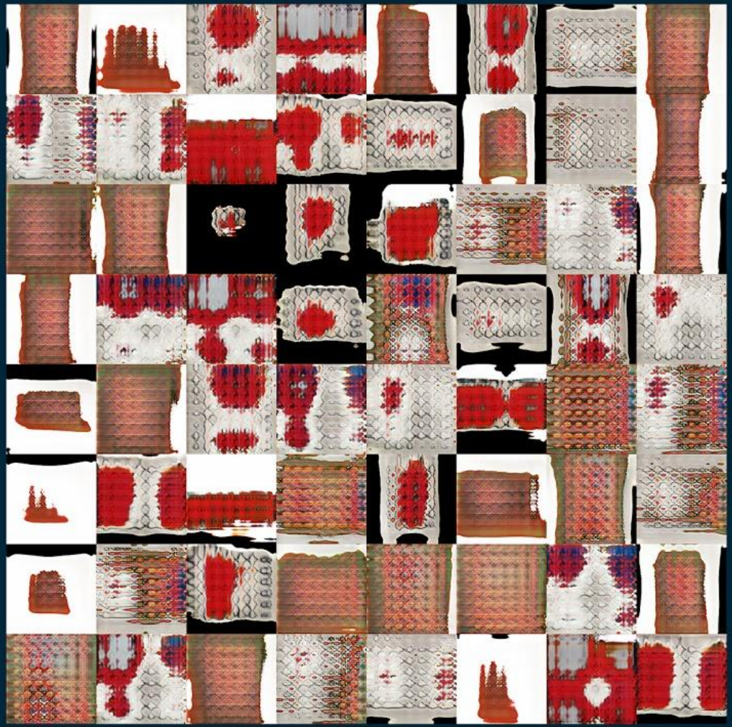
GAN applications

Generate Pictures

Generate time series

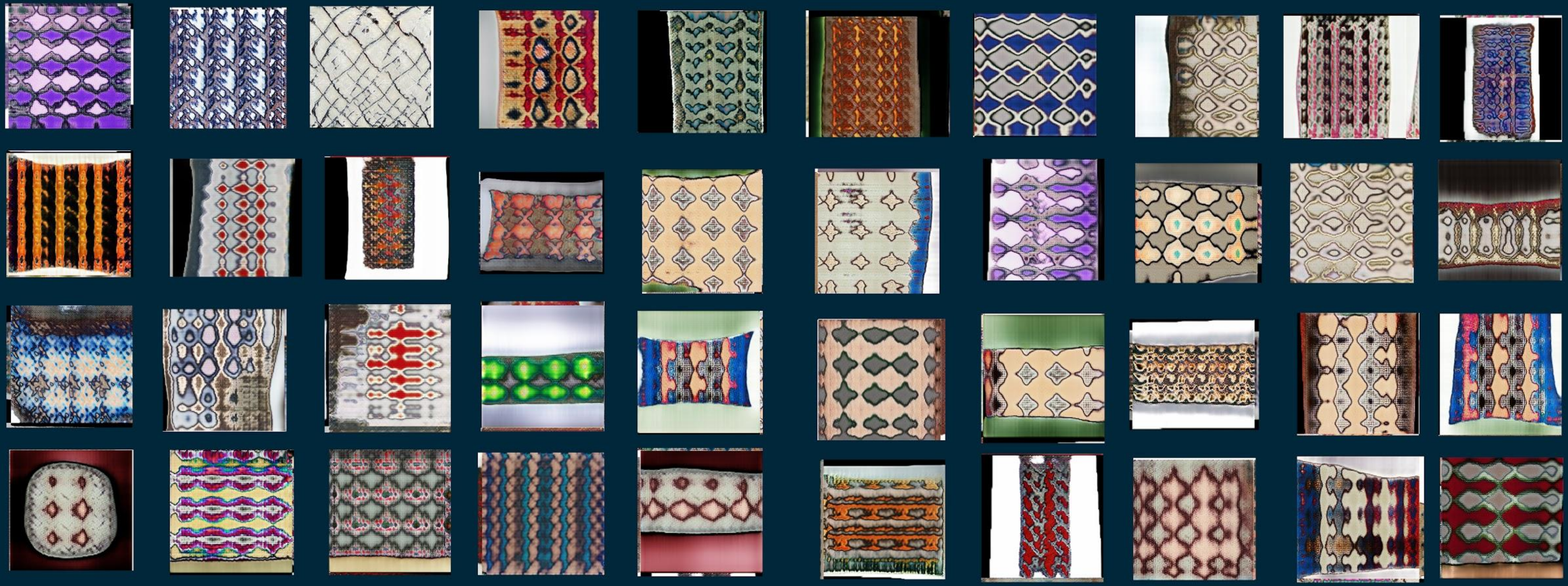


02/03/2019

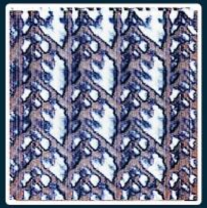


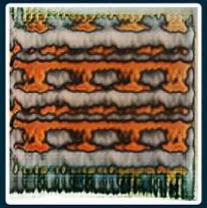
14/03/2019

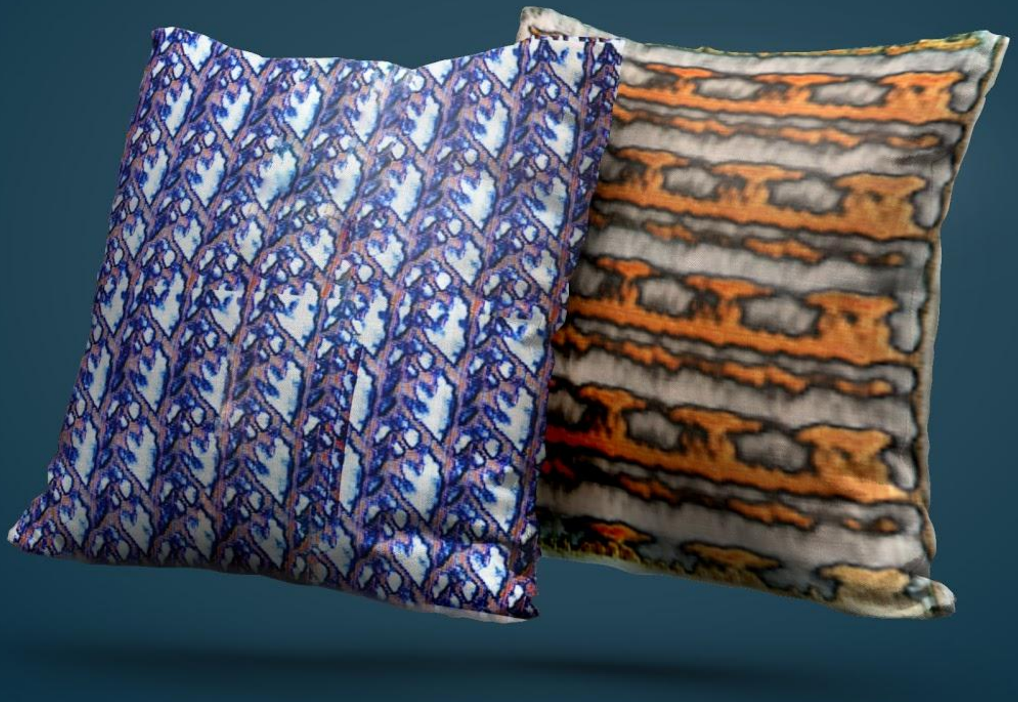
10/04/2019

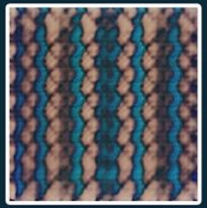


AI generated patterns

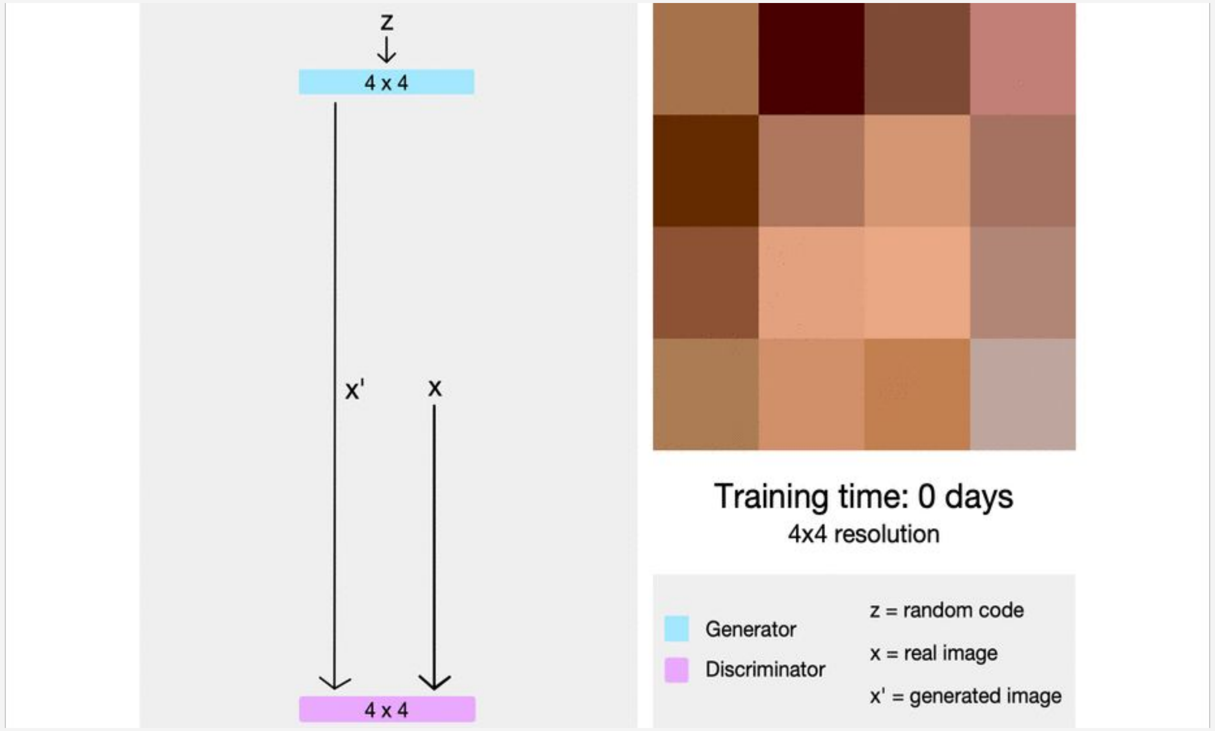


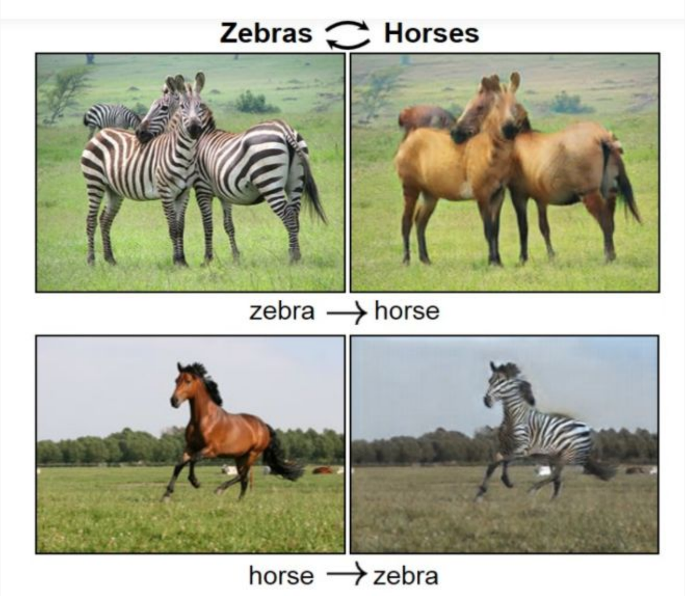






GAN Types





InstaDeep™

Generating high quality realistic photos

- With **GAN**, we can generate photos of imaginary people that we can't distinguish from real photos.

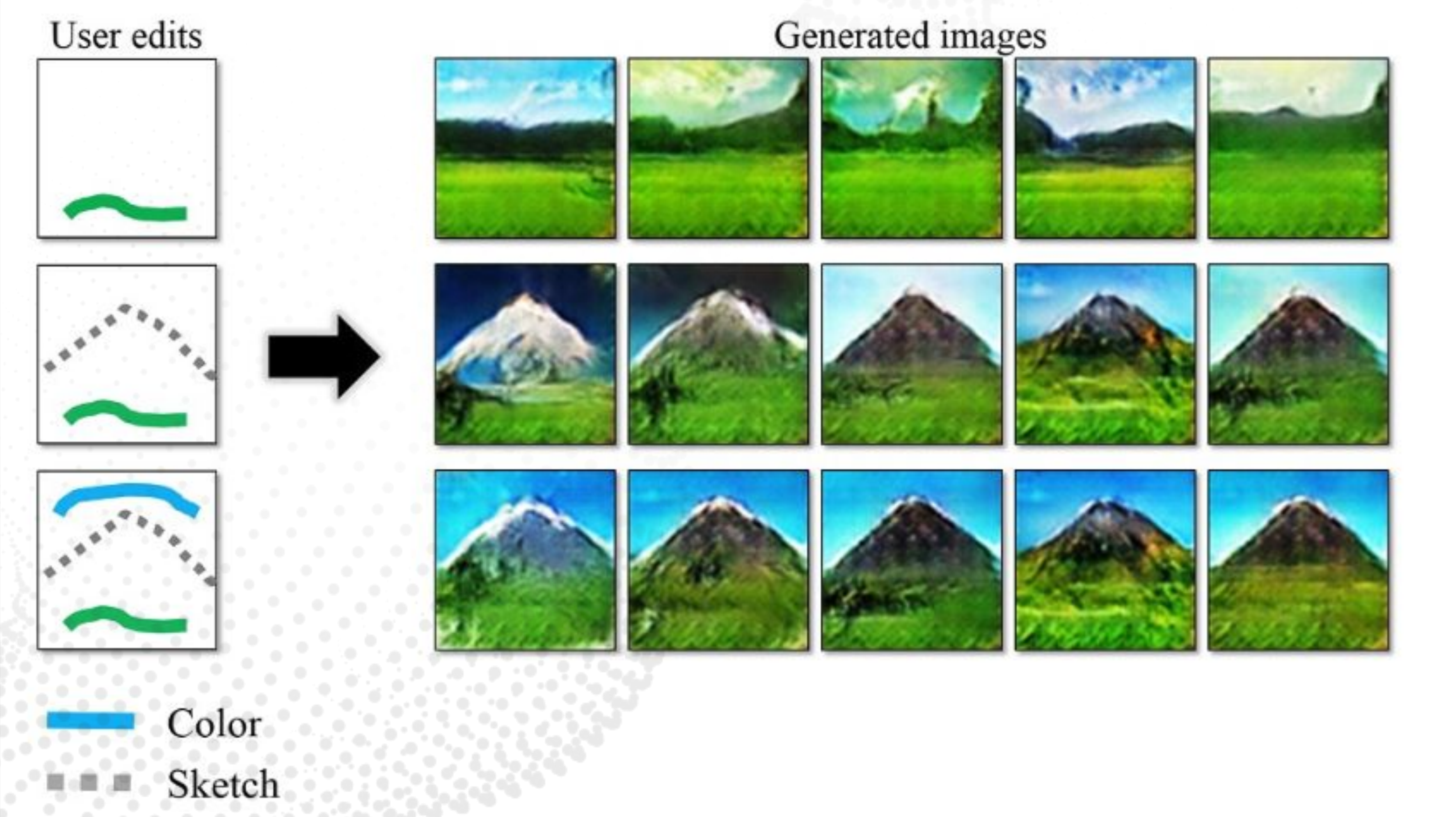
Style transfer

- **GAN** can learn to translate between unpaired sets of images.

InstaDeep™

Image edition

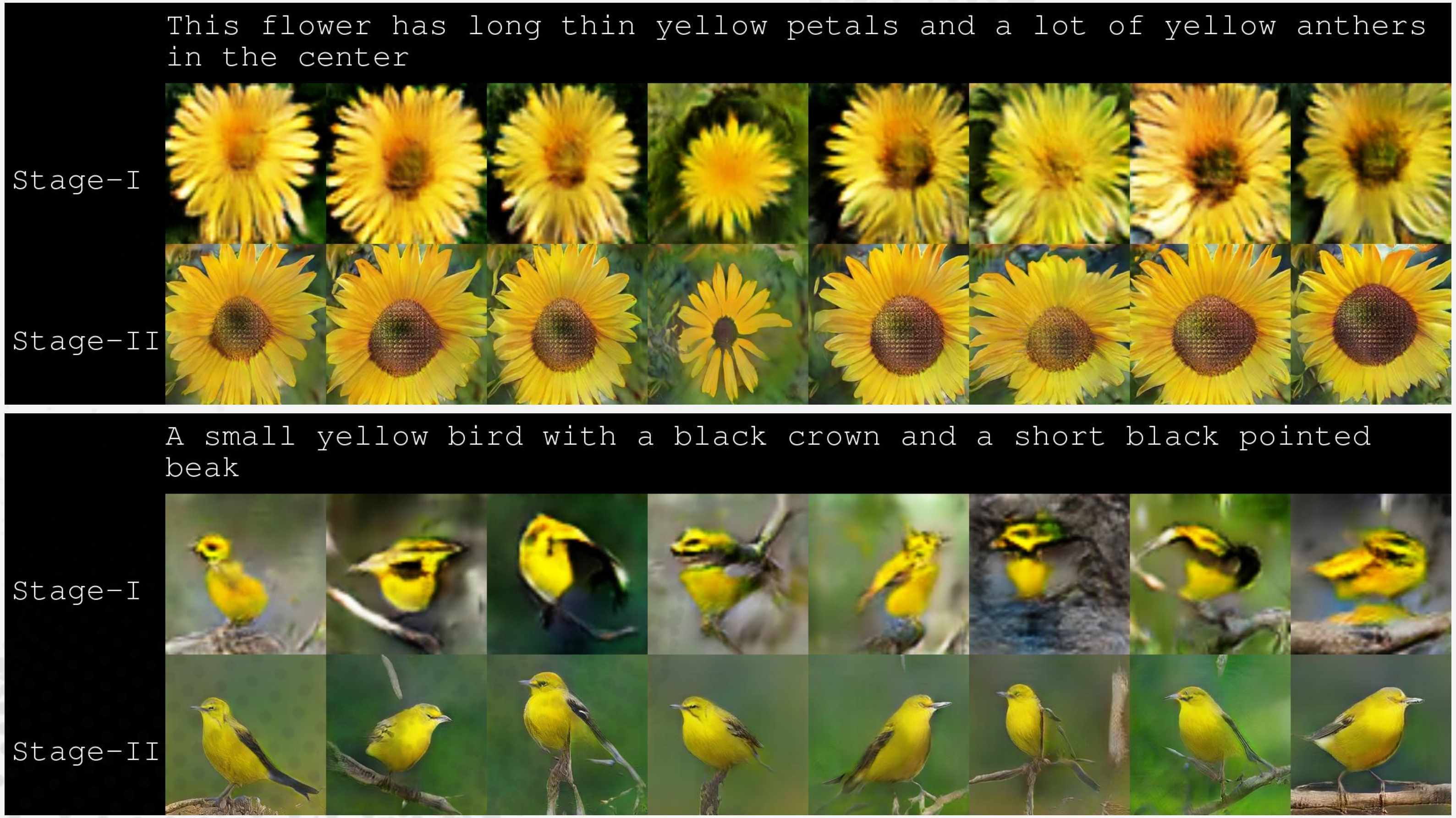
- **GAN** can help you draw complex images with sketches.



InstaDeep™

Text to Image generation

- **GAN** can be used to generate images from a text description.

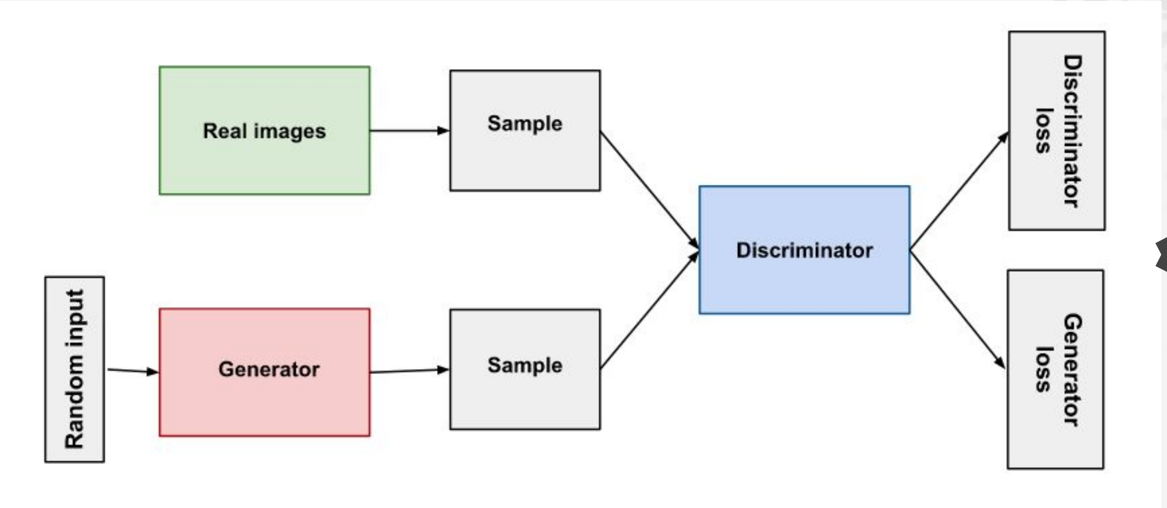


And many other applications

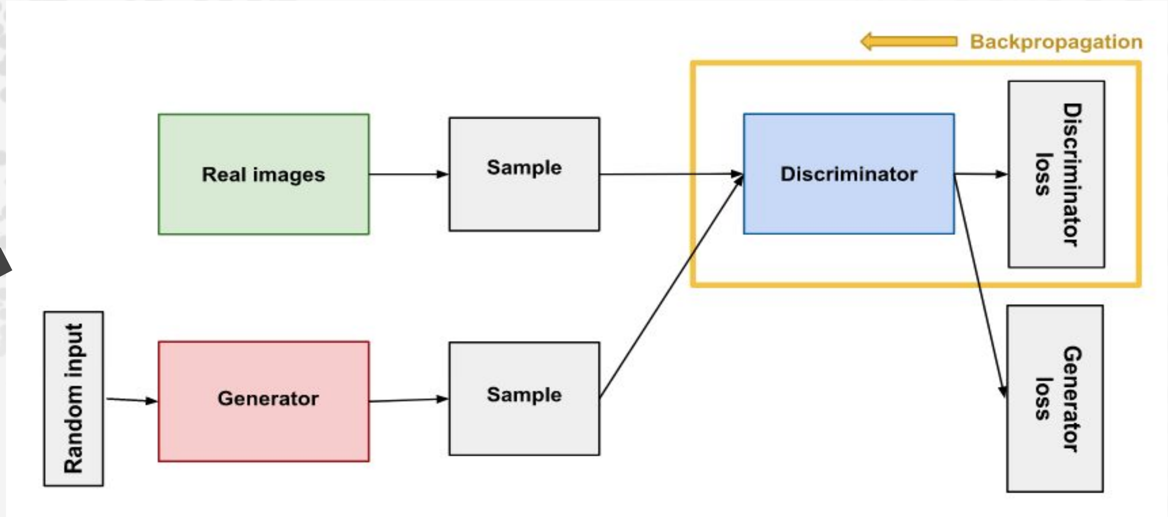
- Generating candidates for cancer drug molecules.
- Augmenting medical records data to help research and overcome regulation in medical data usage.
- Data compression/decompression

InstaDeep™

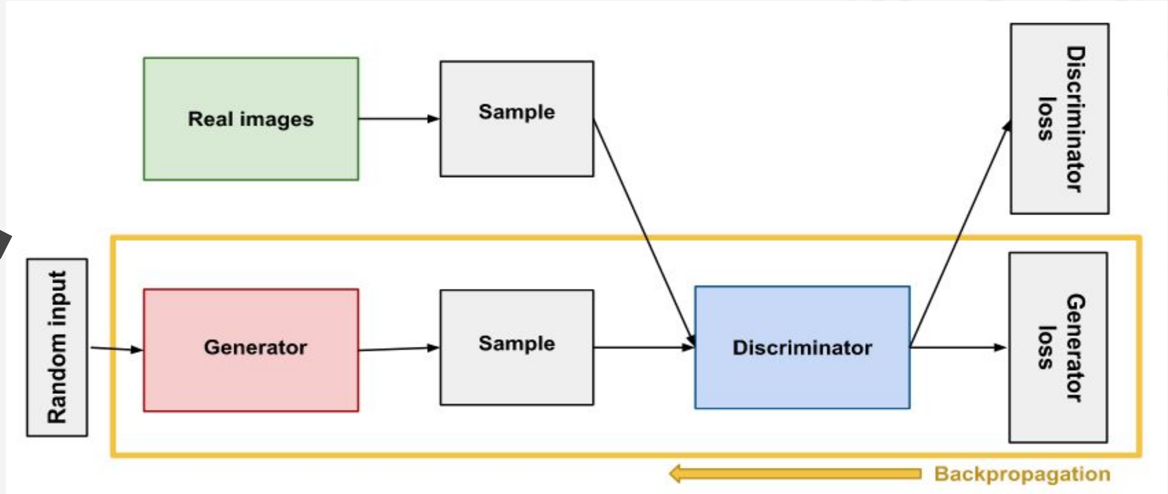
Overview of GAN Structure



GAN



Discriminator



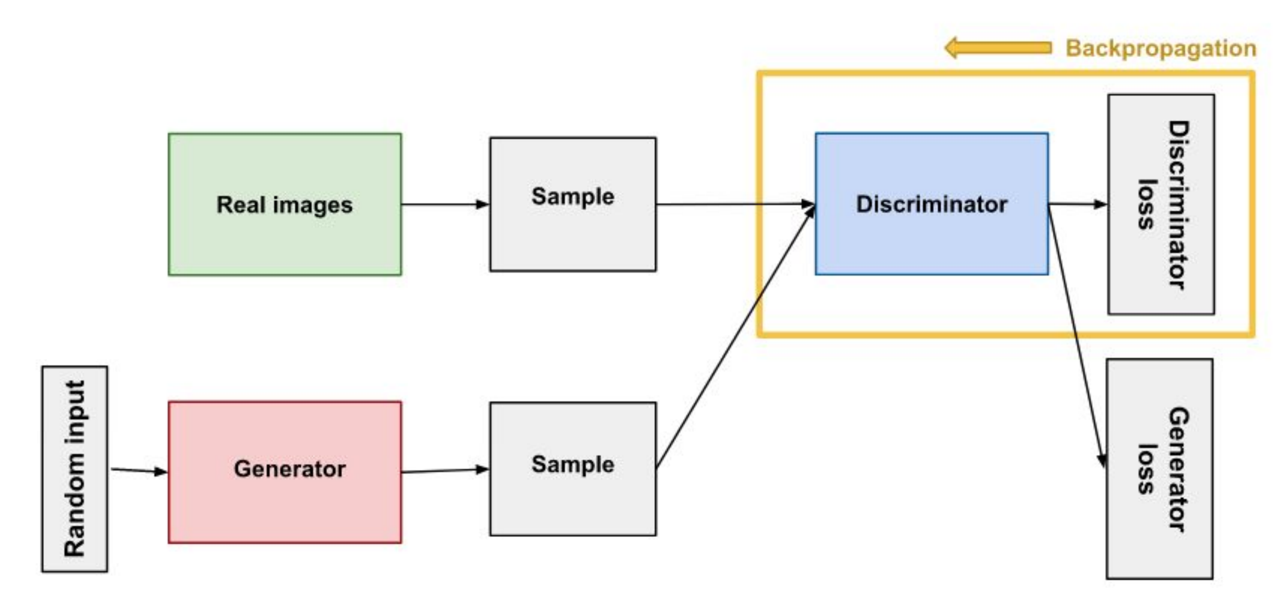
Generator

InstaDeep™

Discriminator

Input: Real data and fake data

- During discriminator training, the discriminator ignores the generator loss and just uses the discriminator loss.
- The discriminator classifies both real data and fake data from the generator.
- The discriminator loss penalizes the discriminator for misclassifying a real instance as fake or a fake instance as real.
- The discriminator updates its weights through backpropagation from the discriminator loss through the discriminator network.

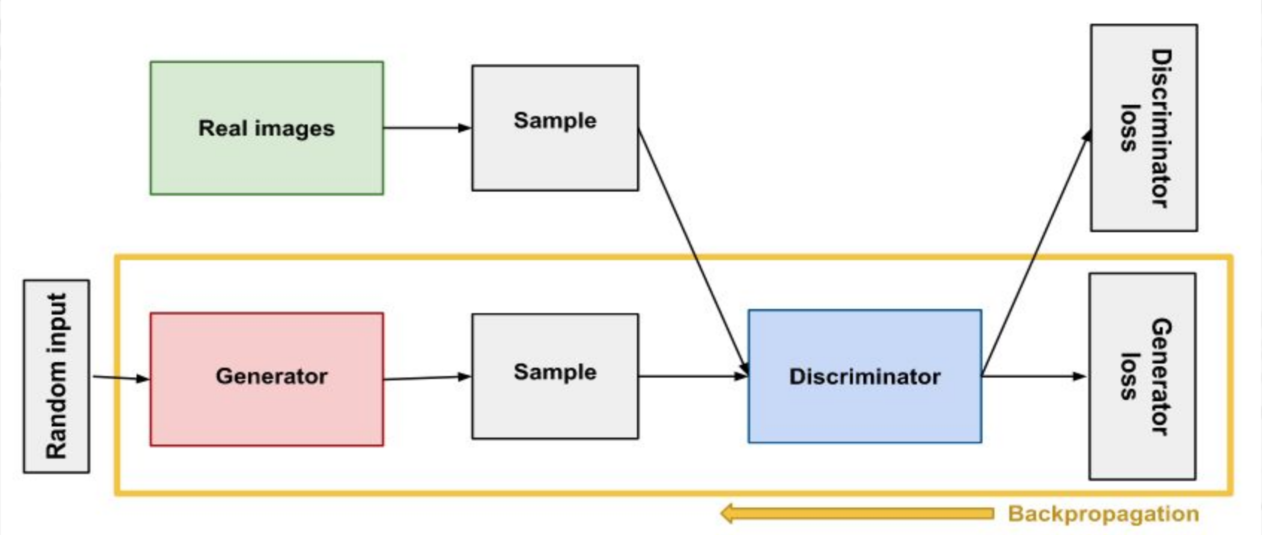


Discriminator

Generator

Input: Noise (inference)

- Sample random noise.
- Produce generator output from sampled random noise.
- Get discriminator "Real" or "Fake" classification for generator output.
- Calculate loss from discriminator classification.
- Backpropagate through both the discriminator and generator to obtain gradients.
- Use gradients to change only the generator weights.



Generator

Loss Functions - MiniMax Loss

GAN goal?

Nash equilibrium, state in game theory, is the stage where the discriminator is unable to distinguish between real and fake artworks.

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

z → Noise vector G(z) → Generator's output → x_{fake}

x → training sample → x_{real}

D (x) → Discriminator's output for x_{real} → P(y | x_{real}) → {0,1}

D (G(z)) → Discriminator's output for x_{fake} → P(y | x_{fake}) → {0,1}

at Discriminator D	at Generator G
D (x) → should be maximized D (G(z)) → should be minimized	D (G(z)) → should be maximized

Common Problems

Vanishing Gradient

Cause : Discriminator is too good.

Optimal solution: discriminator doesn't provide enough information for the generator to make progress.

Mode Collapse

Generate the same pictures using different inference.

Cause: discriminator gets stuck in a local minimum and doesn't find the best strategy.

Failure to Converge: Find the nash equilibrium

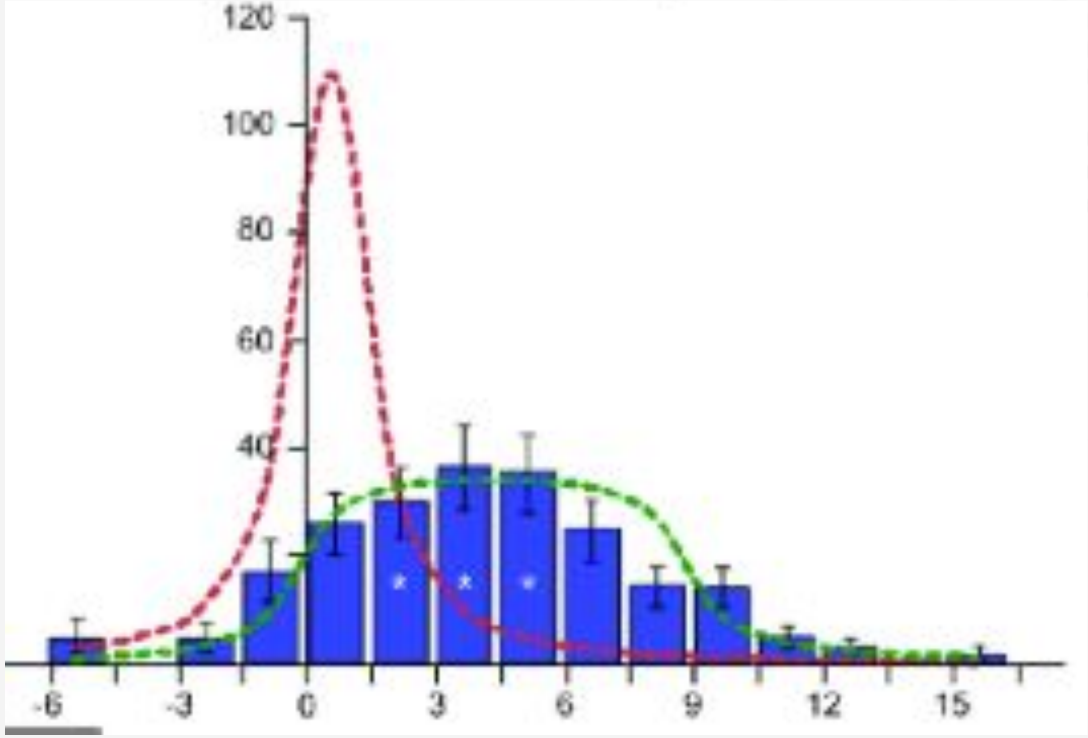
Cause: the discriminator can't easily tell the difference between real and fake. the discriminator is giving completely random feedback, then the generator starts to train on junk feedback, and its own quality may collapse.

For a GAN, convergence is often a fleeting, rather than stable, state.

Wasserstein GAN

The physical idea of EM distance is: how much work you should spend to transport the distribution to another one. As the result, the value is positive and the shape is symmetric. There are two properties that the EM-distance has:

1. The function is continuous anywhere
2. The gradient of the function is almost everywhere



InstaDeep™

Wasserstein GAN

- Wasserstein GANs are less vulnerable to getting stuck than minimax-based GANs, and avoid problems with vanishing gradients.
- The discriminator does no more classify instances.
- The Discriminator tries the score for real instances than for fake instances.

Critic Loss: D(x) - D(G(z))

The discriminator tries to maximize this function. In other words, it tries to maximize the difference between its output on real instances and its output on fake instances.

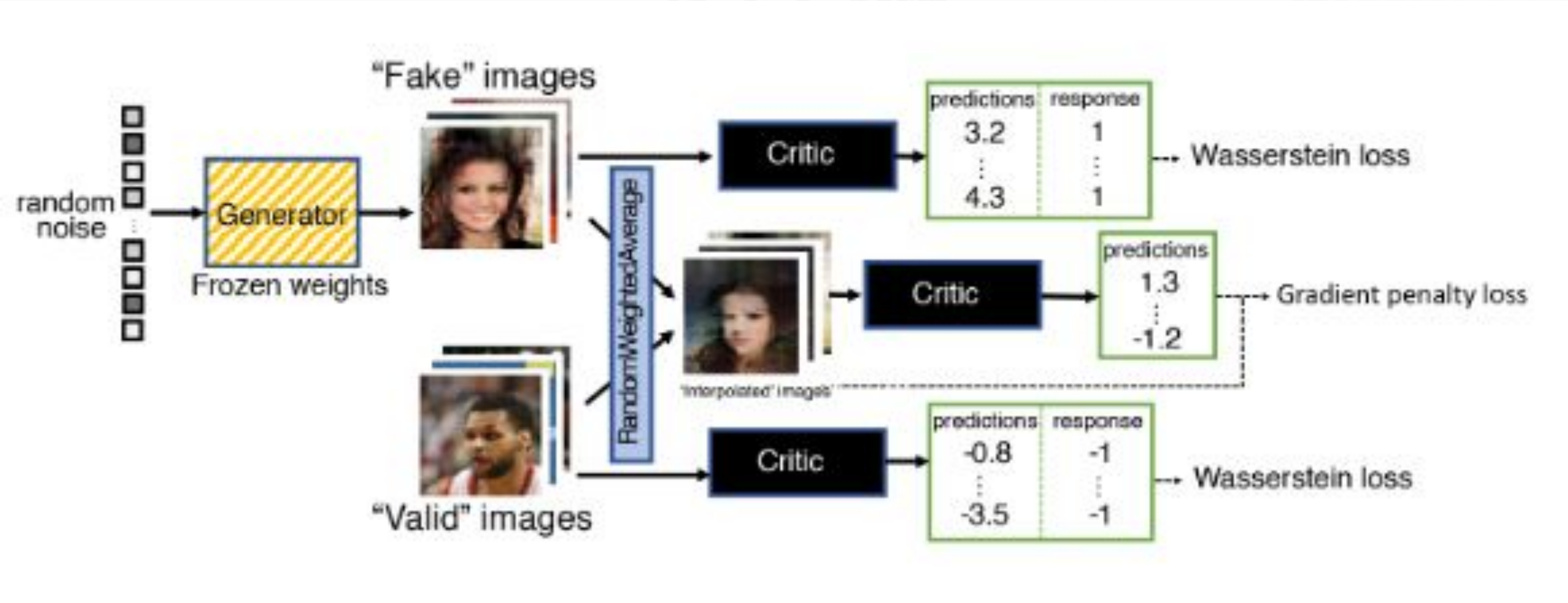
Generator Loss: D(G(z))

The generator tries to maximize this function. In other words, It tries to maximize the discriminator's output for its fake instances.

NB: We should do weight clipping to satisfy the constraint of Lipschitz continuity

Wasserstein GAN with gradient penalty

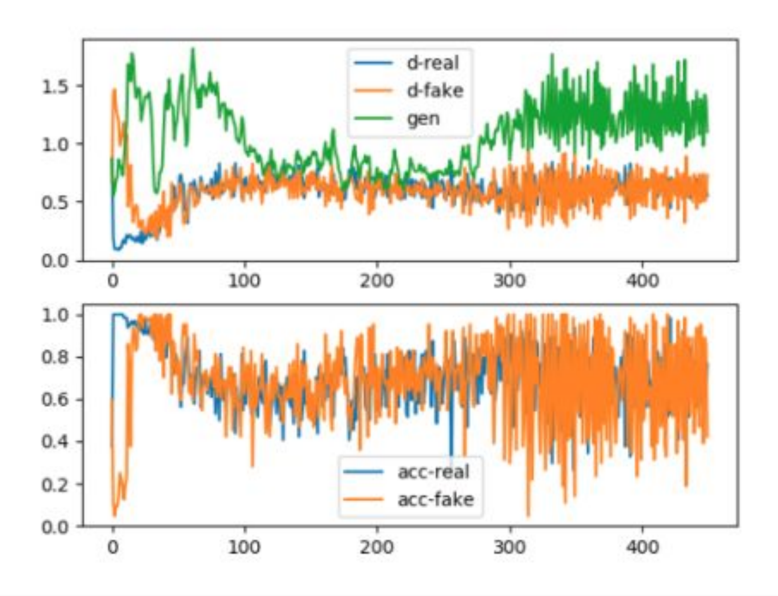
Goal:
Penalizes the model if the gradient norm of the critic deviates from 1.



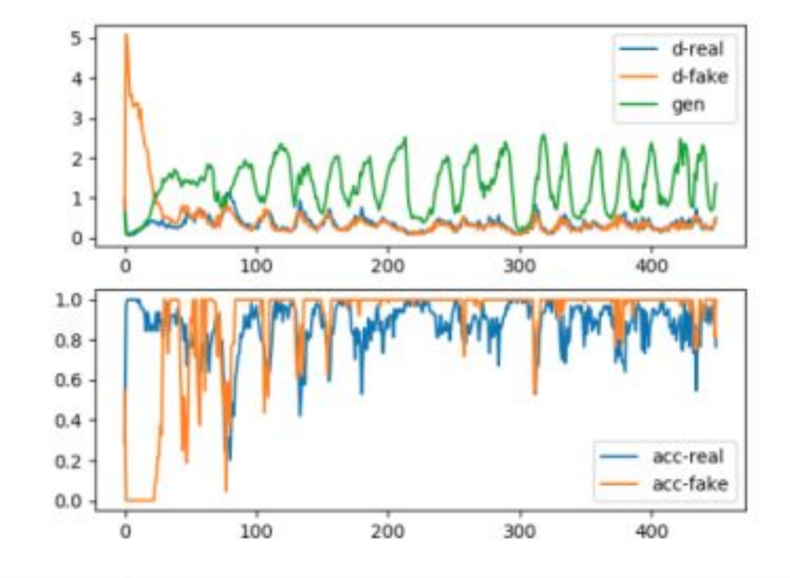
NB:

- Don't clip the weights of the critic.
- Don't use batch normalization layers in the critic.

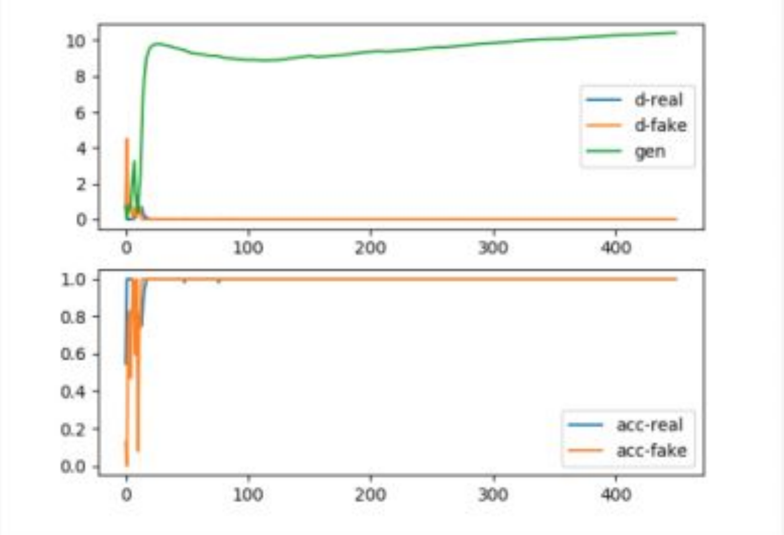
How to debug your code?



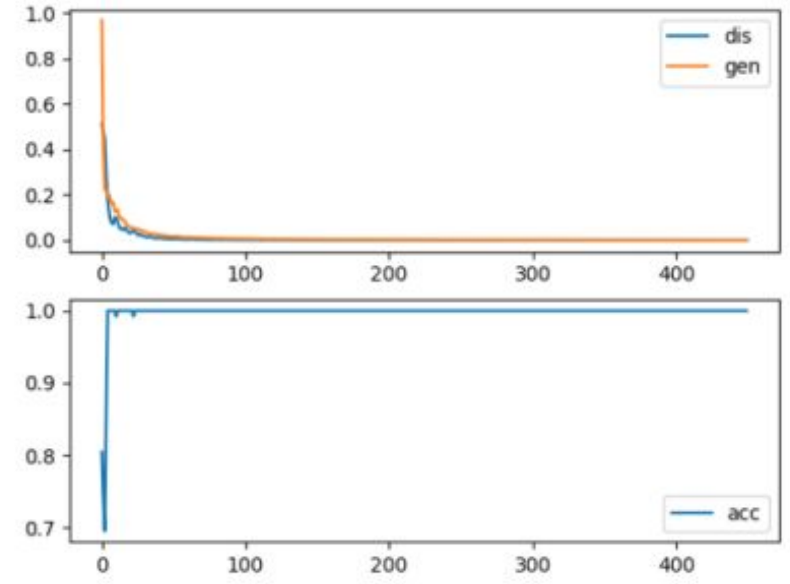
Line Plots of Loss and Accuracy for a Stable Generative Adversarial Network



Line Plots of Loss and Accuracy for a Generative Adversarial Network With Mode Collapse



Sample of 100 Generated Images of a Handwritten Number 8 at Epoch 450 From a GAN That Has a Convergence Failure via Aggressive Optimization.



GANs variations

DCGAN == Deep Convolution GAN

stackGANs ⇒ provide picture based on a text that contain description of a person

cycleGANs => converts photos to painting and the inverse.

3D-GAN ⇒ generating shapes

Age-cGANs ⇒ Face aging has many industry use cases, including cross-age face recognition, finding lost children, and in entertainment.

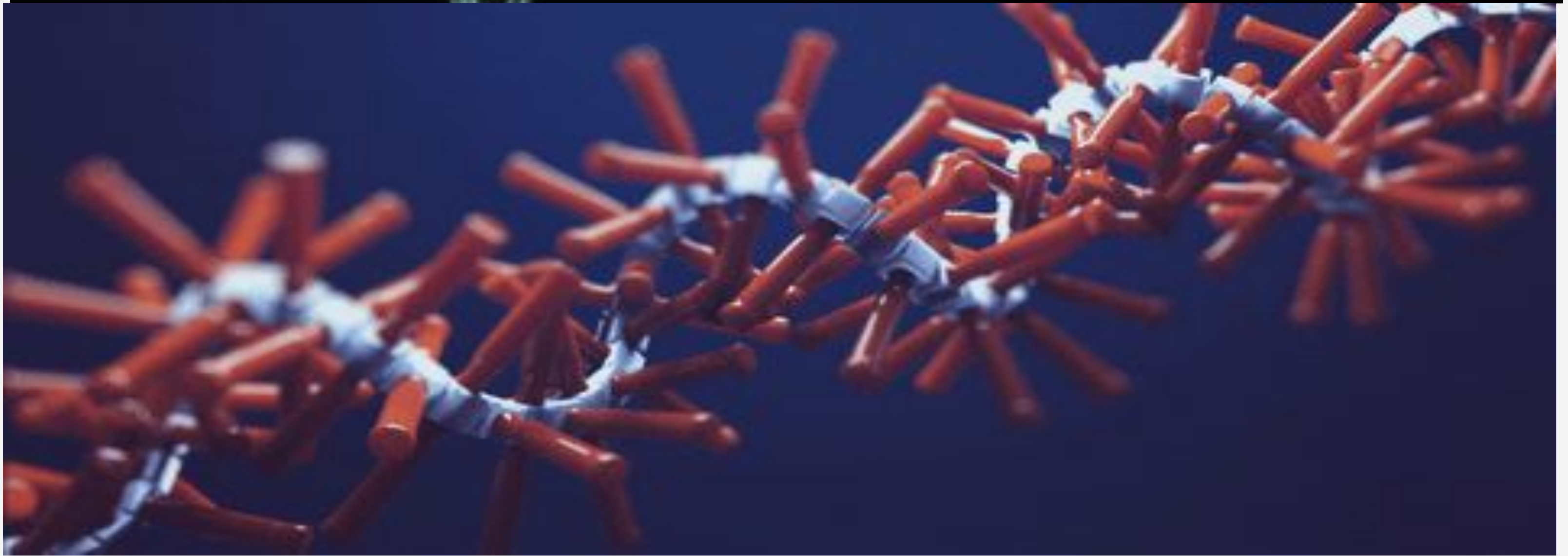
pix2pix ⇒ similar use cases as the cycle GAN. It can convert building labels to pictures of buildings, black and white images to color images, images taken in the day to night images, sketches to photos, and aerial images to map-like images.

LAB:

Github id: **wiemChakroun**

AI


Infrastructure



InstaDeep™

Get in Touch

facebook.com/InstaDeepAI

twitter.com/instadeepai

linkedin.com/company/instadeep

w.chakroun@instadeep.com

www.linkedin.com/in/wiemchakroun/

Thank You

InstaDeep™