
GNOMON®
DIGITAL
GENERATIVE
ADVERSARIAL
NETWORKS
Jiqiong Qiu

Objectif du cours

- Le but de ce cours est l'introduction aux réseaux GAN et Auto-Encoder, nous allons voir la composition et l'architecture de ce genre de réseau, ainsi que les difficultés qu'on peut rencontrer.
- Le cours se focalisera essentiellement sur les aspects suivants :
  - *GAN*
  - *Auto-Encoder*
  - *Introduction to Hugging Face*

C’est quoi un réseau GAN

- Les réseaux antagonistes génératifs (GAN) ont été introduits pour la première fois en 2014 par Ian Goodfellow et. Al.
- Les réseaux GAN sont des réseaux de neurones qui **génèrent du matériel**, tel que des images, de la musique, de la parole ou du texte, similaire à ce que les humains produisent.
- Les réseaux GAN sont composés par **deux réseaux de neurones** qui se font **concurrence** pour devenir plus **précis dans leurs prédictions**.
- Les GAN fonctionnent généralement **sans supervision** et utilisent un cadre de **jeu coopératif à somme nulle** pour apprendre.

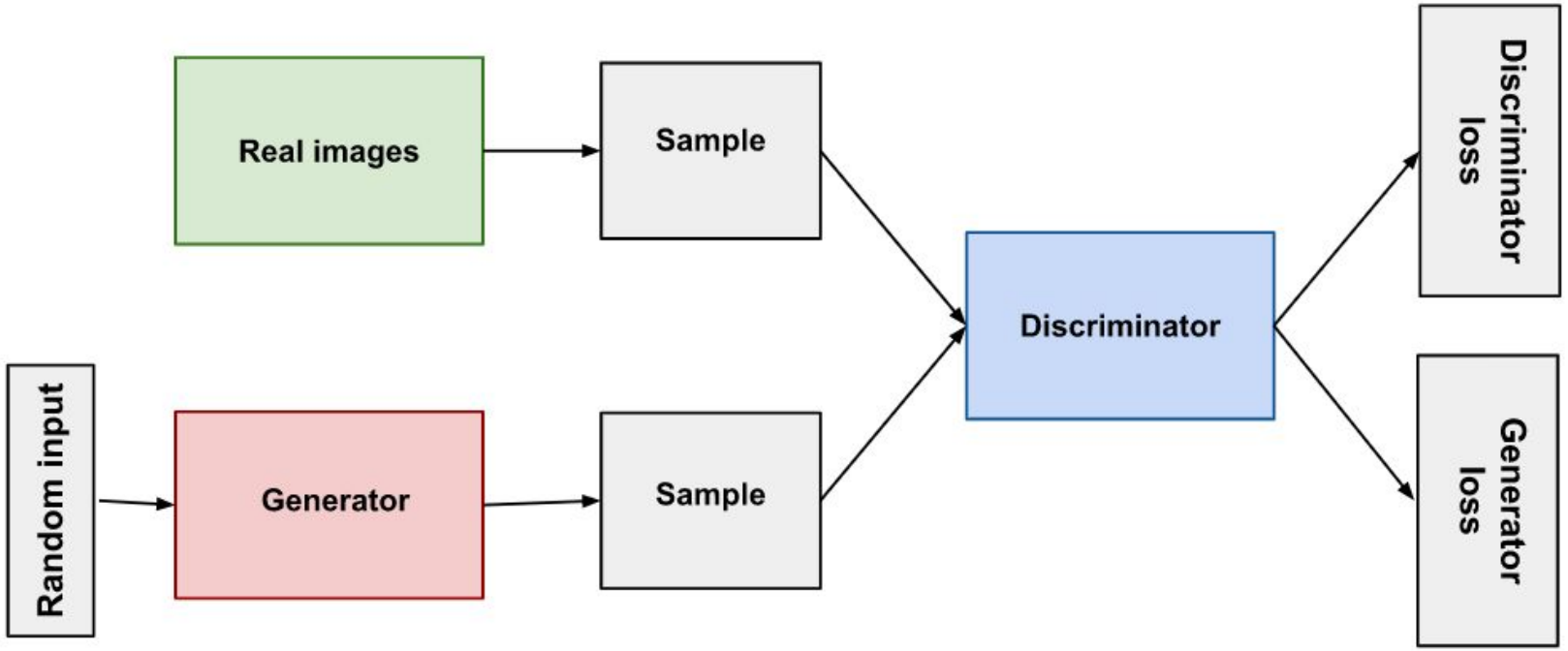
Composition d’un GAN

GNOMON®
DIGITAL

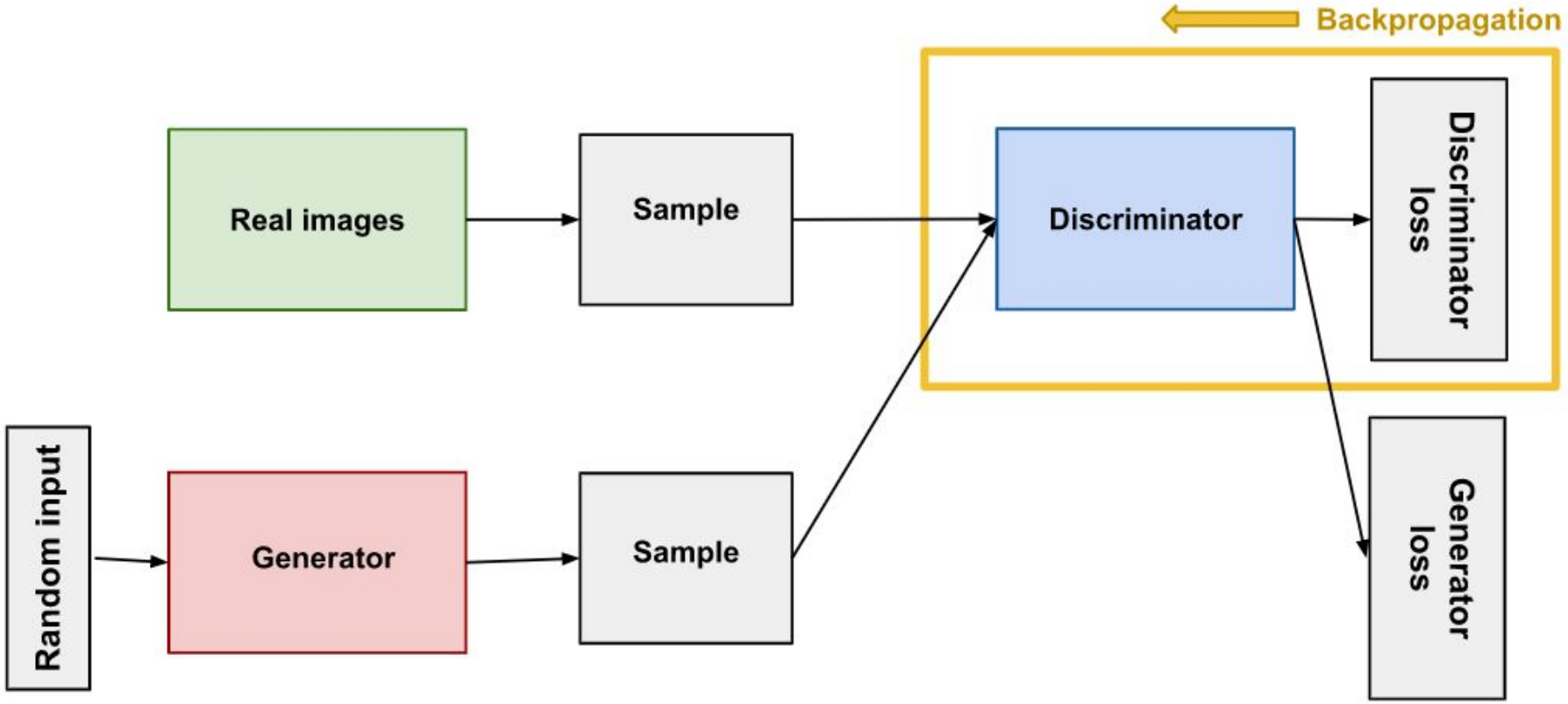
Un réseau GAN se compose de deux parties:

- Le générateur apprend à générer des données plausibles. Les instances générées deviennent des exemples d'entraînement négatifs pour le discriminateur.
- Le discriminateur apprend à distinguer les données fictives des données réelles. Le discriminateur pénalise le générateur pour avoir produit des résultats impossibles à comprendre.

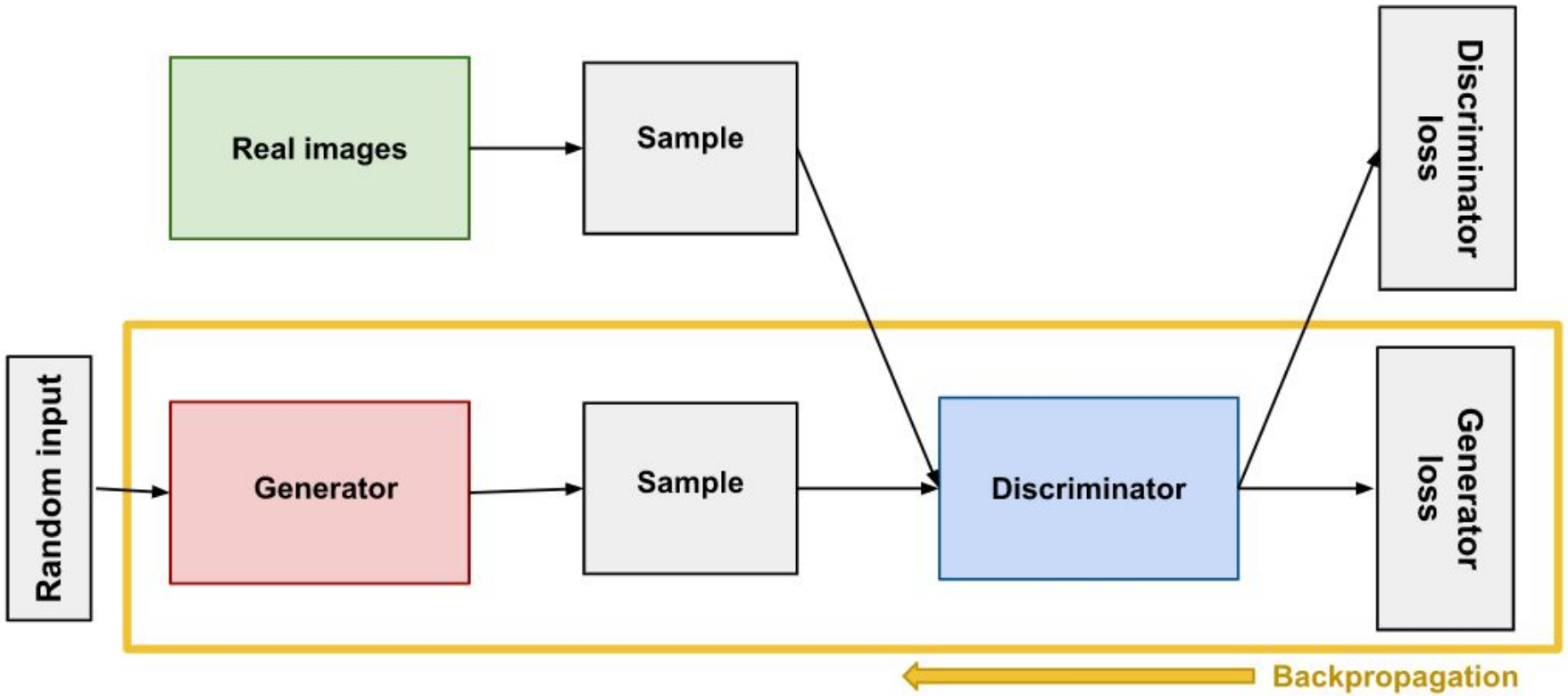
Architecture d’un GAN



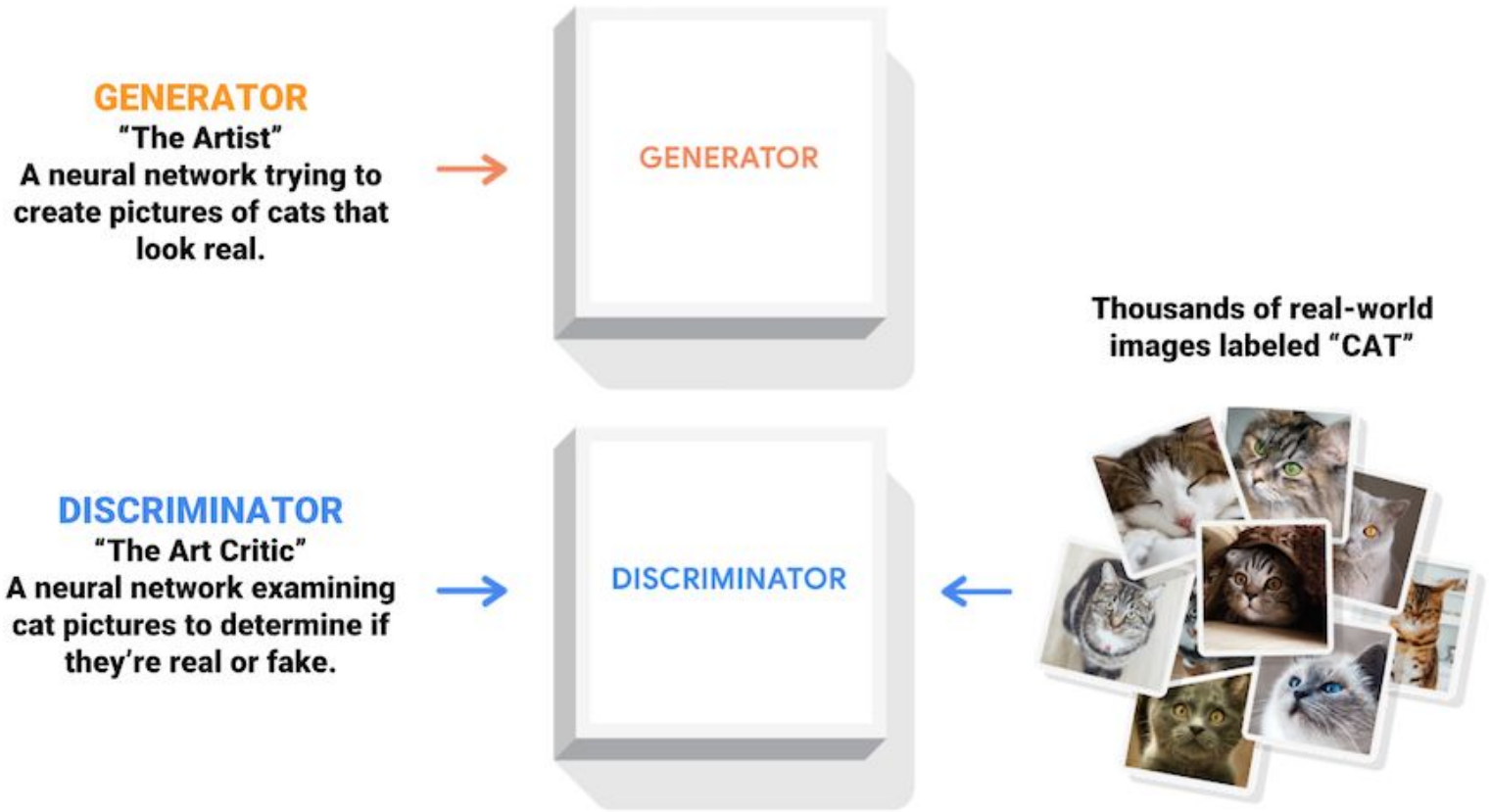
Le discriminateur



Le générateur



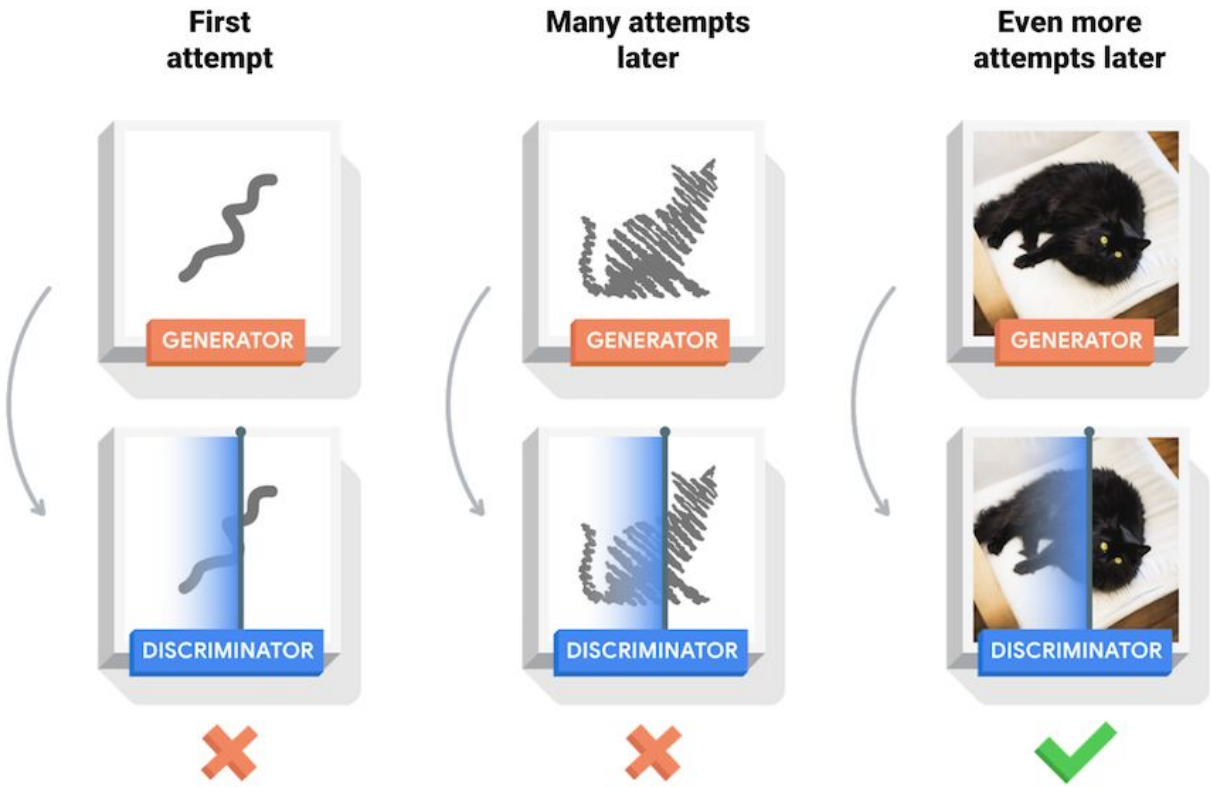
Exemple d’un GAN



Source : https://www.tensorflow.org/

Exemple d’un GAN

Au cours de l'entraînement, le générateur s'améliore progressivement pour créer des images qui semblent réelles, tandis que le discriminateur s'améliore pour les différencier. Le processus atteint l'équilibre lorsque le discriminateur ne peut plus distinguer les images réelles des fausses.



Source : https://www.tensorflow.org/

Complexité des GAN

- Les GAN doivent jongler avec deux types d'entraînement différents (générateur et discriminateur).
- La convergence du GAN est difficile à identifier. Car au bout d’un certain temps d’entrainement, les commentaires du discriminateur deviennent moins significatifs.
- Si le GAN continue de s'entraîner au-delà du moment où le discriminateur envoie des commentaires complètement aléatoires, le générateur commence à s'entraîner sur les commentaires indésirables et sa qualité peut s'effondrer.

Fonctions de perte

GNOMON DIGITAL

- Les GAN tentent de répliquer une distribution de probabilité. Elles doivent donc utiliser des fonctions de perte qui reflètent la distance entre la distribution des données générées par le GAN et la distribution des données réelles.

- Un GAN peut avoir deux fonctions de perte: une pour l'entraînement du générateur et une pour l'entraînement du discriminateur.

- Nous allons aborder deux fonctions de perte : Perte minimax, Perte Wasserstein.

Perte minimale

GNOMON®
DIGITAL

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

- D(x) est une estimation de la probabilité que l'instance de données x réelle soit réelle.
- Ex est la valeur attendue sur toutes les instances de données réelles.
- G(z) est la sortie du générateur lorsqu'il reçoit une valeur de z pour le bruit.
- D(G(z)) est l'estimation de la probabilité qu'une fausse instance soit réelle.
- Ez est la valeur attendue sur toutes les entrées aléatoires du générateur (en effet, il s'agit de la valeur attendue sur toutes les fausses instances générées, G(z)).

Perte de Wasserstein

- Cette fonction de perte dépend d'une modification du schéma du GAN (nommé "Wasserstein GAN" ou "WGAN") dans lequel le discriminateur ne classe pas les instances.
- Perte de critiques: D(x) - D(G(z))
- Le discriminateur tente de maximiser cette fonction. En d'autres termes, il tente de maximiser la différence entre sa sortie sur des instances réelles et sa sortie sur de fausses instances.
- Perte de générateur : D(G(z))
- Le générateur tente de maximiser cette fonction. En d'autres termes, il tente d'optimiser le résultat du discriminateur pour ses fausses instances.

Où:
D(x) est le résultat du critique d'une instance réelle.
G(z) est la sortie du générateur lorsqu'il reçoit une valeur de z pour le bruit.
D(G(z)) est la sortie du critique d'une fausse instance.

Problèmes courants

- Dégradés : Une étude a suggéré que si votre discriminateur est trop bon, l'entraînement du générateur peut échouer en raison de la disparition des gradients. En effet, un discriminateur optimal ne fournit pas suffisamment d'informations au générateur pour progresser.
- Réduction du mode : si un générateur génère une sortie particulièrement plausible, il peut apprendre à produire uniquement cette sortie
- Échec de la convergence

Problèmes courants

- Dégradés : Une étude a suggéré que si votre discriminateur est trop bon, l'entraînement du générateur peut échouer en raison de la disparition des gradients. En effet, un discriminateur optimal ne fournit pas suffisamment d'informations au générateur pour progresser.
- Réduction du mode : si un générateur génère une sortie particulièrement plausible, il peut apprendre à produire uniquement cette sortie
- Échec de la convergence

Déclinaisons des GAN

GNOMON®
DIGITAL

En quelques années, les chercheurs ont proposés des améliorations au réseau GAN classiques, afin de répondre à des problématiques bien spécifiques:

- DCGAN : Deep Convolutional Generative Adversarial Network
- CycleGAN
- Conditional GAN (CGAN)

Deep Convolutional Generative Adversarial Network

GNOMON®
DIGITAL

- Remplacer toutes les couches de mise en commun par des convolutions à foulée (discriminateur) et des convolutions à foulée fractionnaire (générateur).
- Utilisation de batchnorm dans le générateur et le discriminateur.
- Suppression des couches cachées entièrement connectées pour des architectures plus profondes.
- Utilisation de l'activation ReLU dans le générateur pour toutes les couches à l'exception de la sortie, qui utilise tanh.
- Utilisation de l'activation LeakyReLU dans le discriminateur pour toutes les couches.

CycleGAN

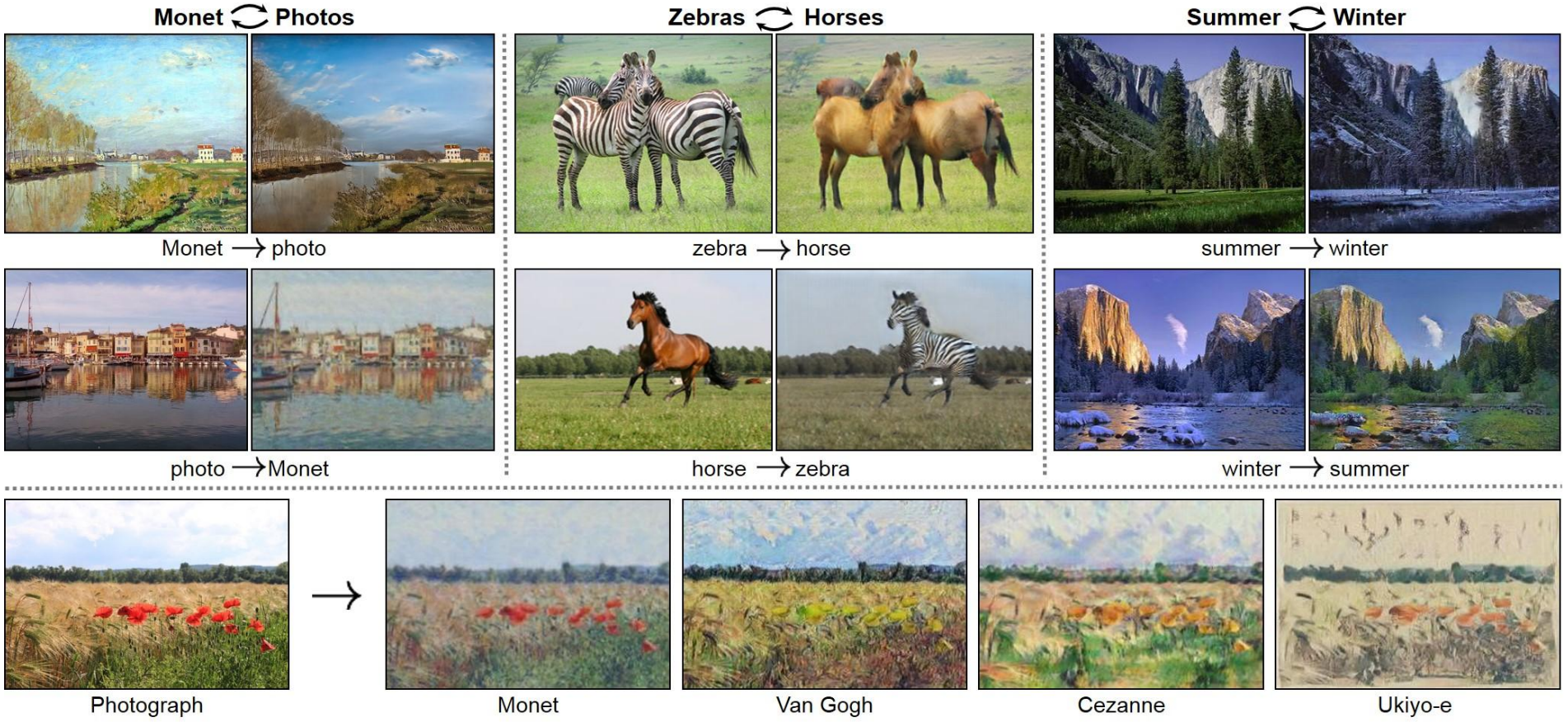
CycleGAN est une architecture GAN très populaire principalement utilisée pour apprendre la transformation entre des images de styles différents.

A titre d'exemple, ce type de formulation peut apprendre :

- une carte entre images artistiques et réalistes
- une transformation entre des images de cheval et de zèbre
- une transformation entre image d'hiver et image d’été

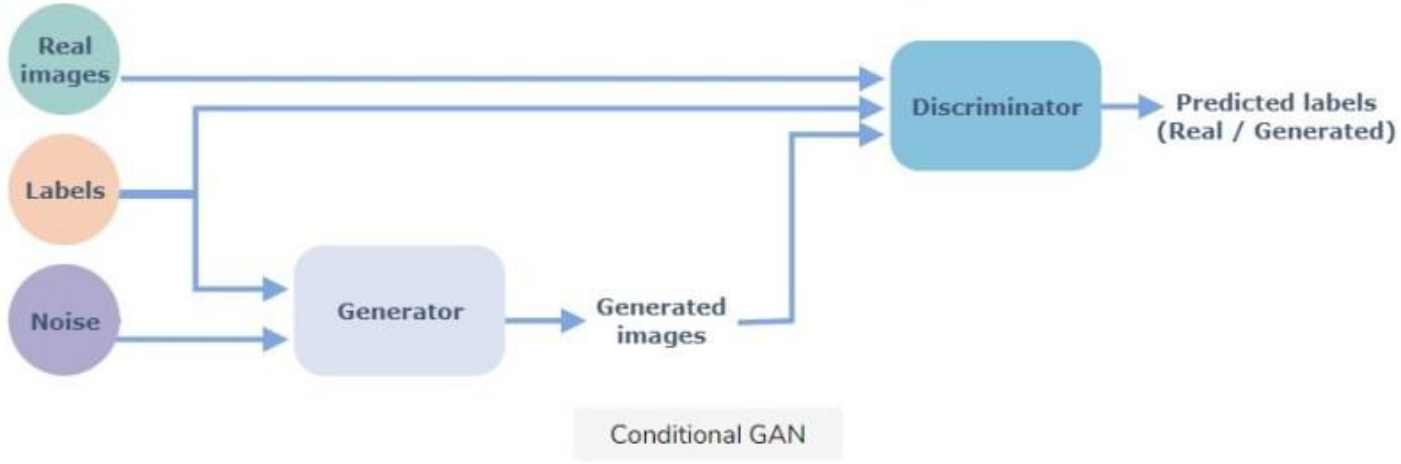
FaceApp est l'un des exemples les plus populaires de CycleGAN où les visages humains sont transformés en différents groupes d'âge.

CycleGAN


Monet ⇄ Photos
Zebras ⇄ Horses
Summer ⇄ Winter
Monet → photo
zebra → horse
summer → winter
photo → Monet
horse → zebra
winter → summer
Photograph
Monet
Van Gogh
Cezanne
Ukiyo-e

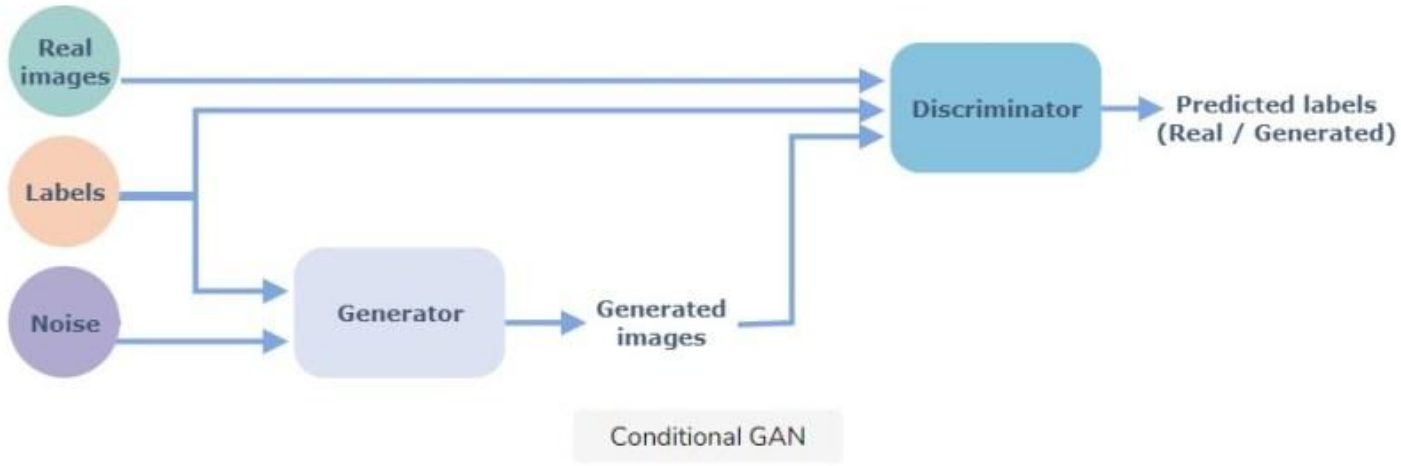
CycleGAN

- Avec un « **conditional gan** », il est possible d'envoyer des informations **plus précises** (données labélisées), au générateur et au discriminateur pour cadrer leur production de données. Ces informations vont permettre de préciser les données produites par le générateur et le discriminateur afin qu'ils arrivent plus rapidement au résultat voulu.
- Au lieu de produire des images de vêtements, par exemple, il produira des images de pantalons, de vestes, ou de chaussettes selon le label qu'on lui fourni.
- Les domaines d'application des cGAN sont très viarés:
  - La traduction d'image à image
  - Création d'images à partir de texte
  - La génération de vidéo
  - Génération de visages

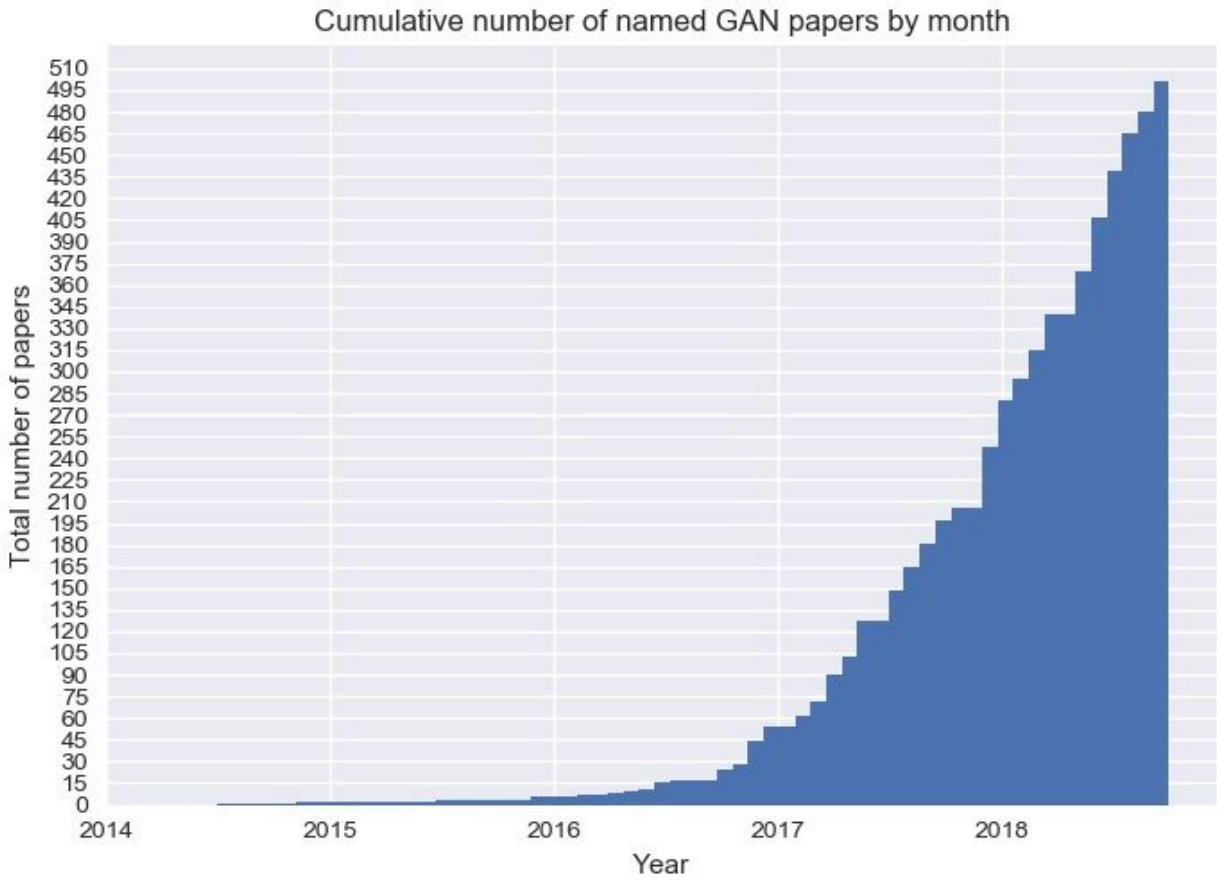


CycleGAN

- Avec un « **conditional gan** », il est possible d'envoyer des informations **plus précises** (données labélisées), au générateur et au discriminateur pour cadrer leur production de données. Ces informations vont permettre de préciser les données produites par le générateur et le discriminateur afin qu'ils arrivent plus rapidement au résultat voulu.
- Au lieu de produire des images de vêtements, par exemple, il produira des images de pantalons, de vestes, ou de chaussettes selon le label qu'on lui fourni.
- Les domaines d'application des cGAN sont très viarés:
  - La traduction d'image à image
  - Création d'images à partir de texte
  - La génération de vidéo
  - Génération de visages



GAN ZOO



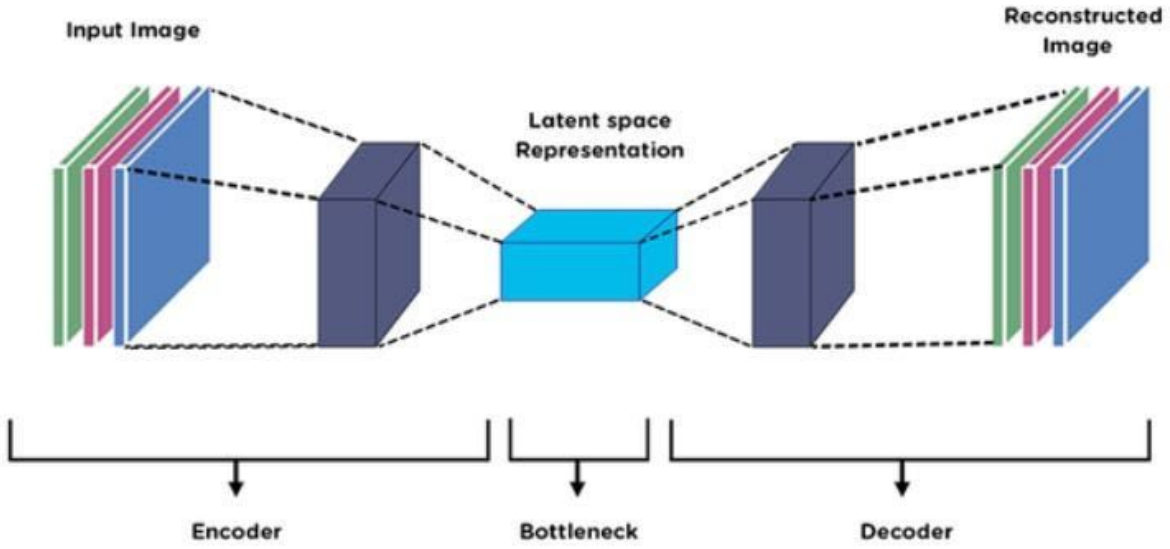
Very popular until ChatGPT

https://github.com/hindupuravinash/the-gan-zoo

Auto-Encoder

GNOMON DIGITAL

Pourquoi Auto-encoder?



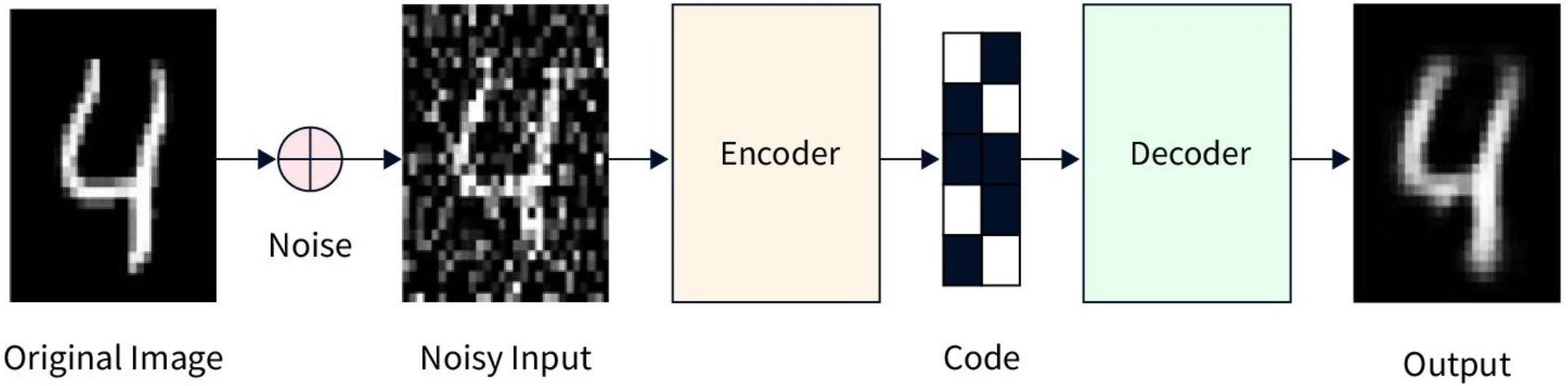
Reducing the Dimensionality of Data with Neural Networks

G. E. HINTON AND R. R. SALAKHUTDINOV Authors Info & Affiliations

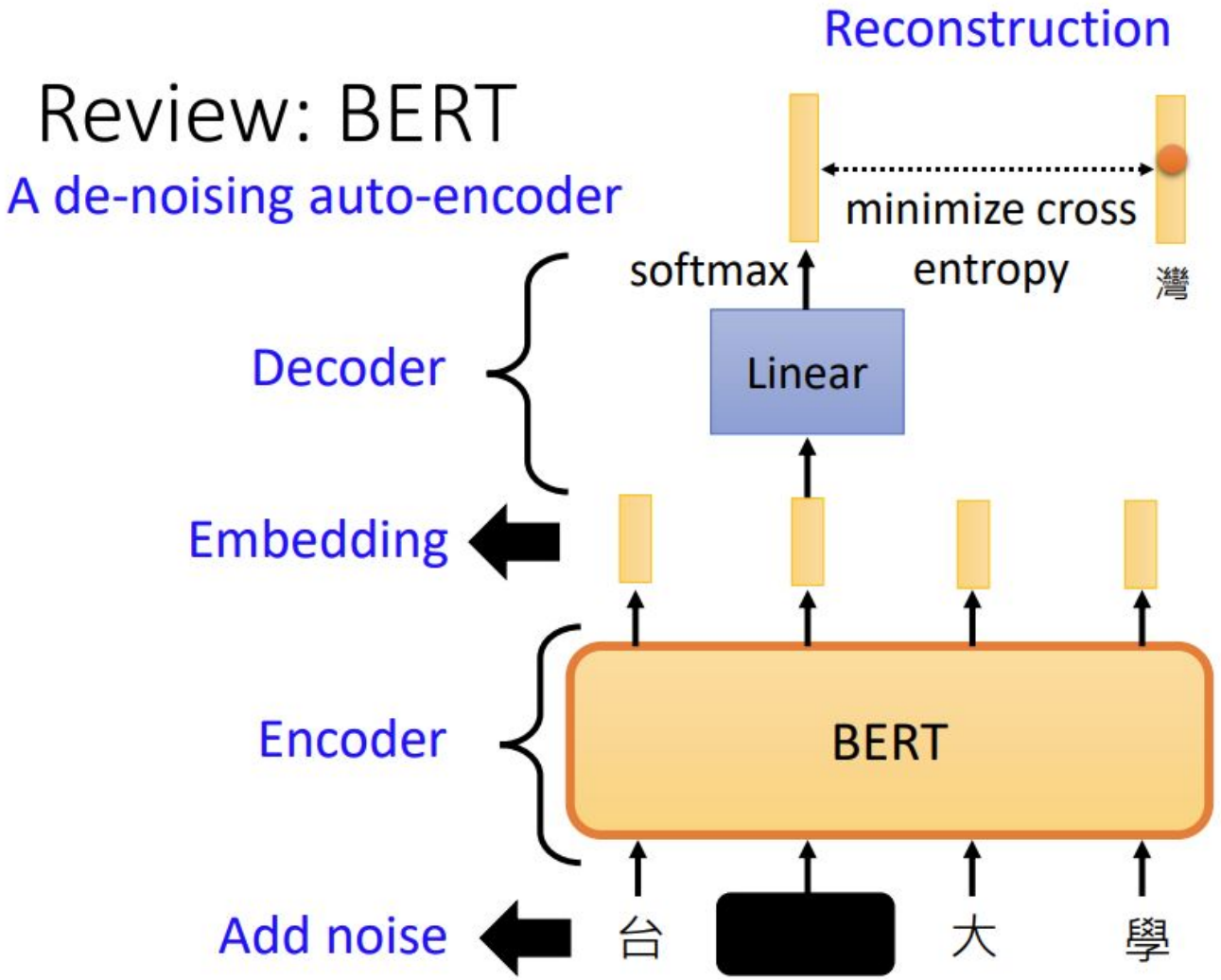
SCIENCE • 28 Jul 2006 • Vol 313, Issue 5786 • pp. 504-507 • DOI: 10.1126/science.1127647

Auto-Encoder

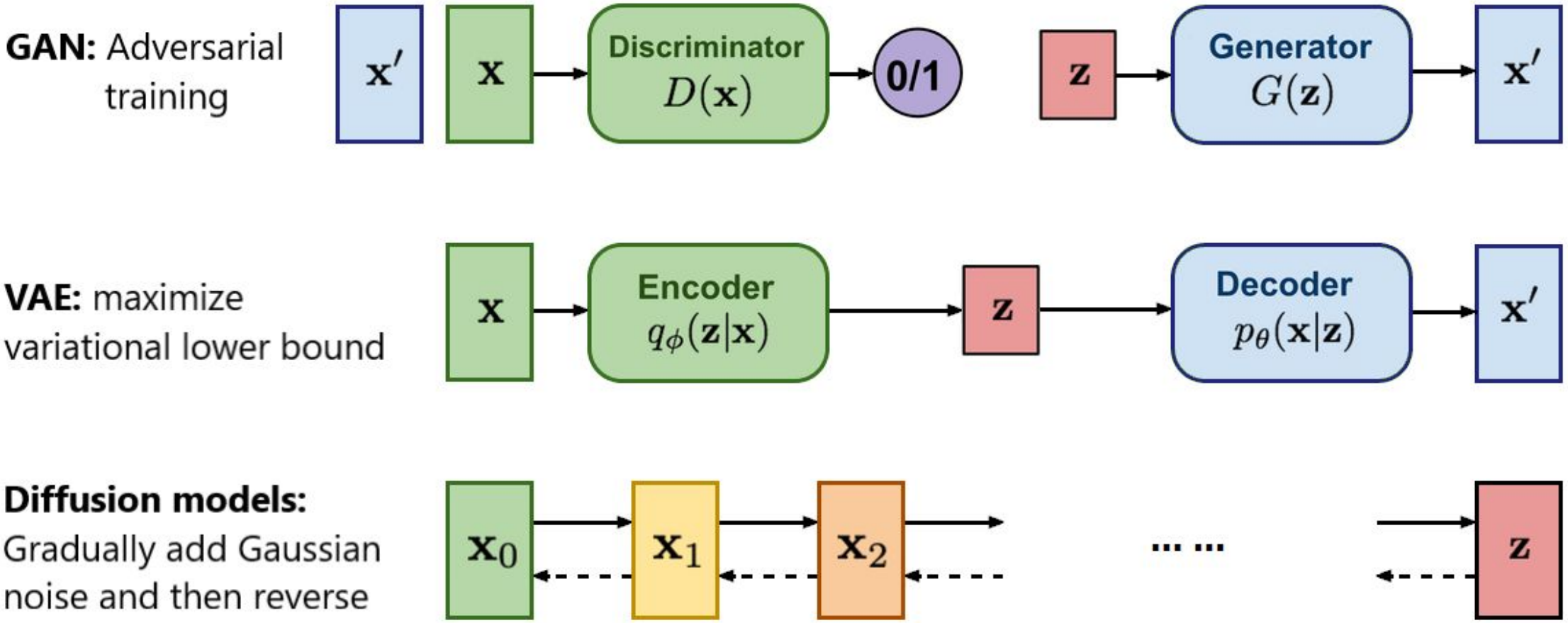
De-noising Auto-encoder



Auto-Encoder

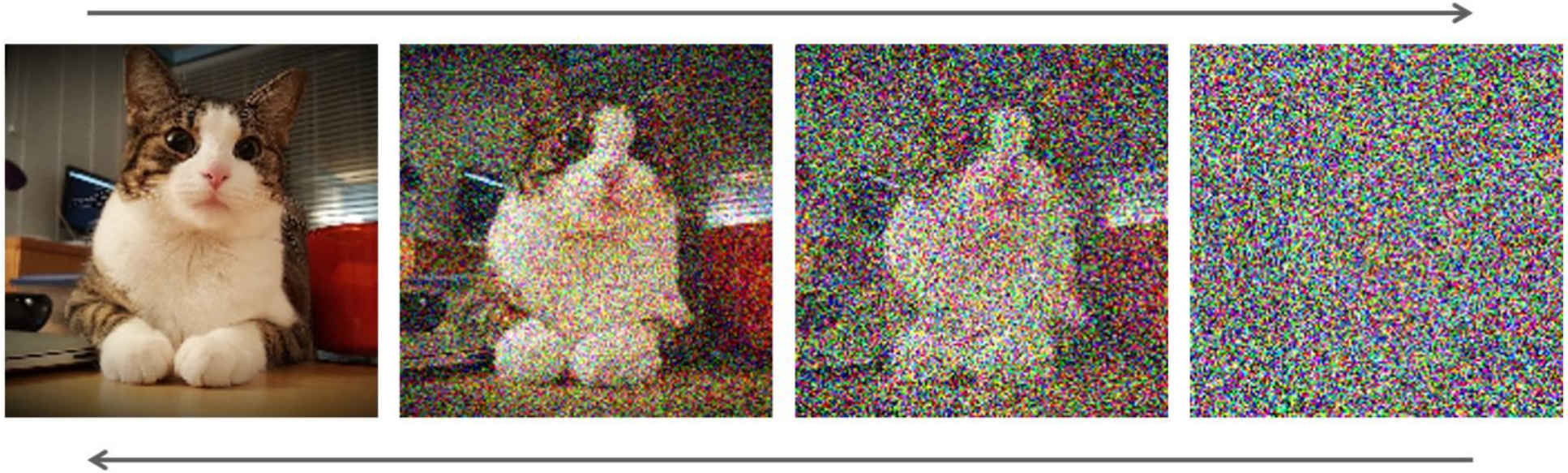


Auto-Encoder



Auto-Encoder

On peut donc générer des images à partir un vecteur random
Avec quelques modifications, on a VAE (variational auto-encoder)



Auto-Encoder

GNOMON DIGITAL

Application

Training Data:

anomaly

Training Data:

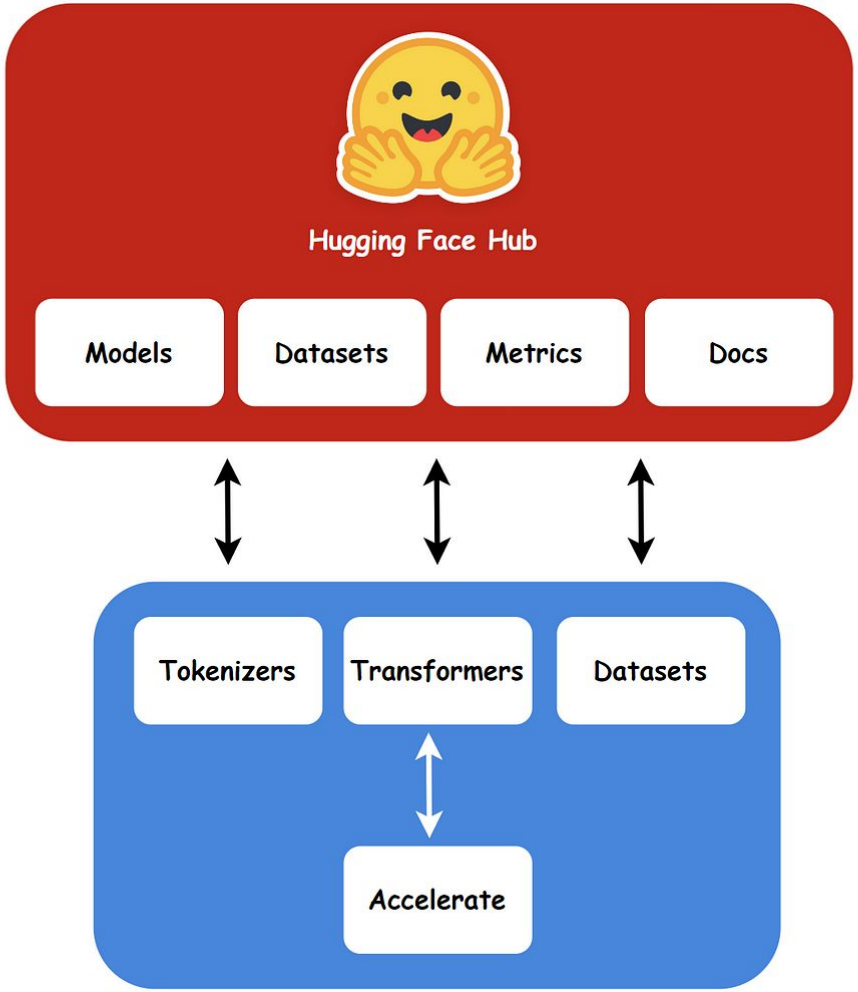
anomaly

Training Data:

anomaly

Hugging FACE

Oublie Keras, TF, Pytorch, hugging face is all you need



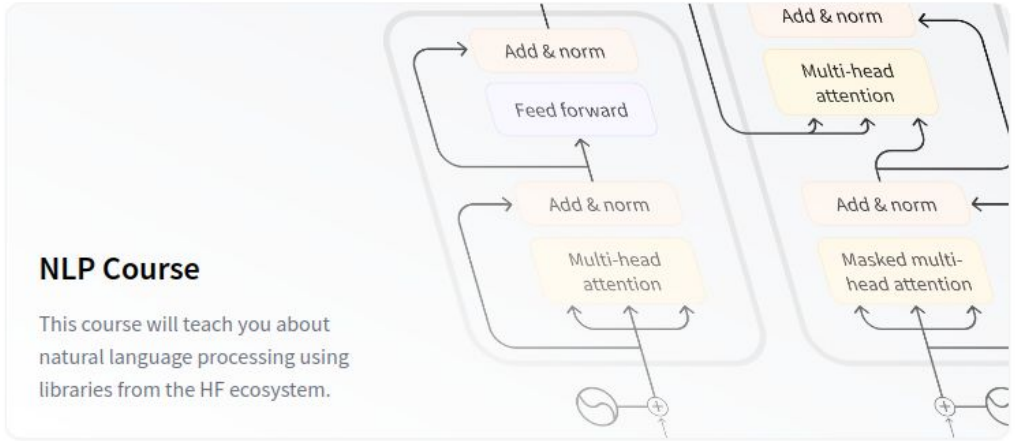
Démocratiser l'accès aux modèles d'IA

Hugging FACE

GNOMON DIGITAL

https://huggingface.co/learn

Learn

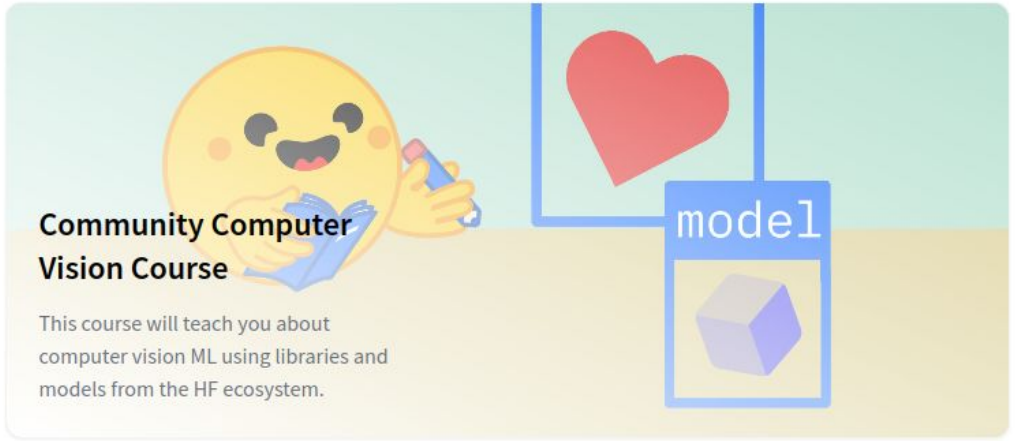


NLP Course

This course will teach you about natural language processing using libraries from the HF ecosystem.

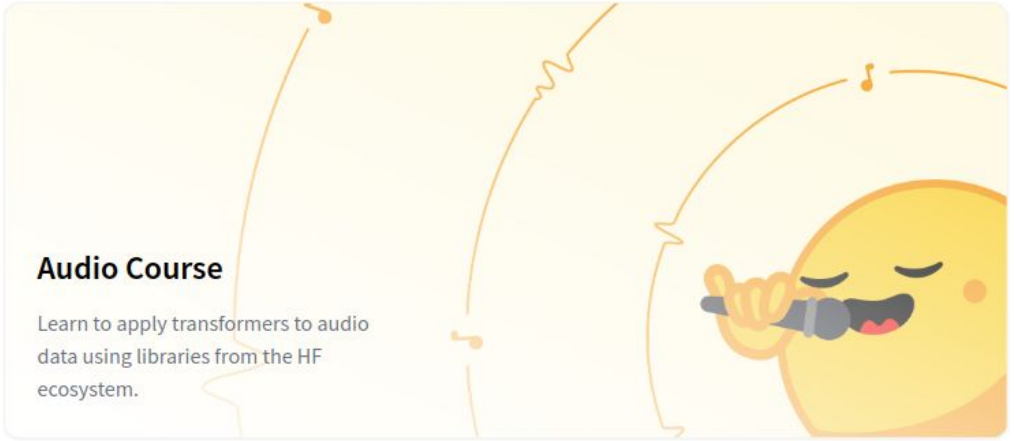
Deep RL Course

This course will teach you about deep reinforcement learning using libraries from the HF ecosystem.



Community Computer Vision Course

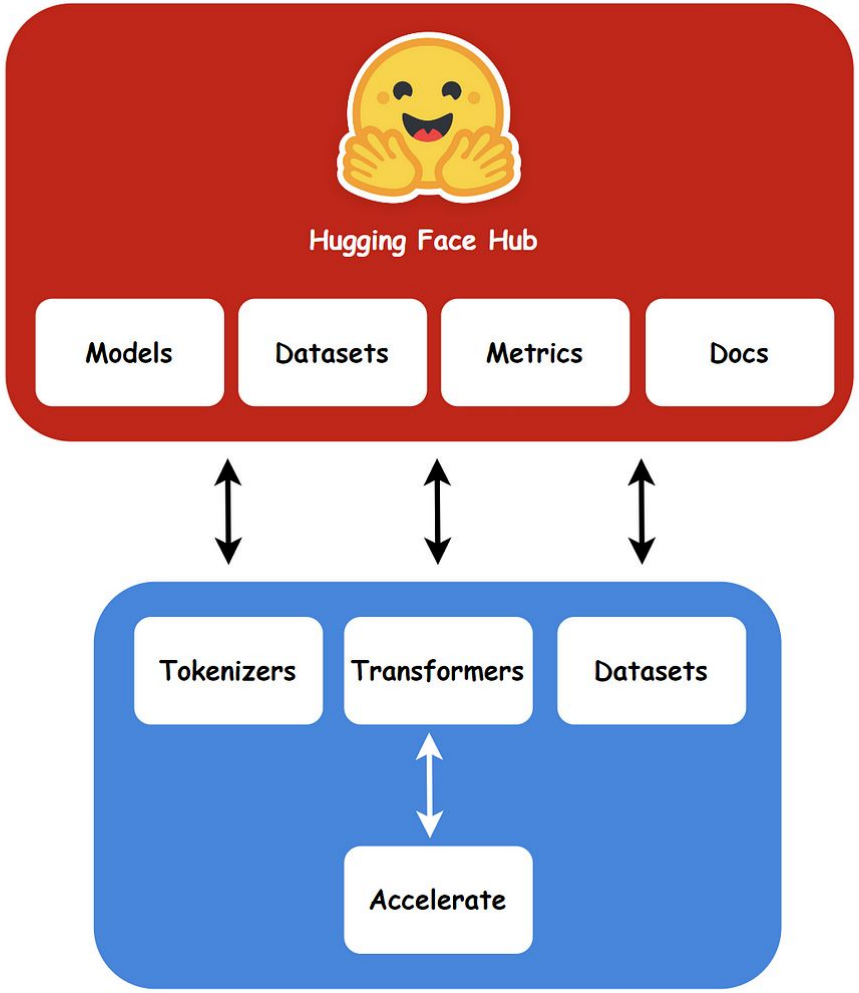
This course will teach you about computer vision ML using libraries and models from the HF ecosystem.



Audio Course

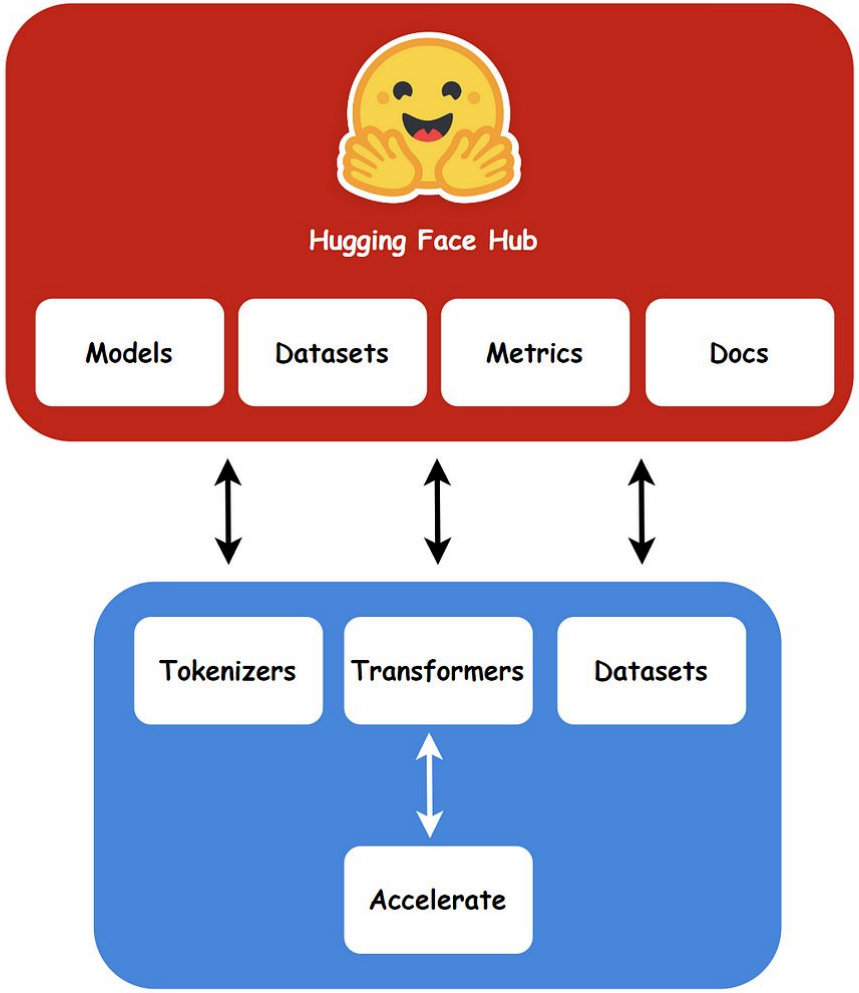
Learn to apply transformers to audio data using libraries from the HF ecosystem.

Hugging FACE



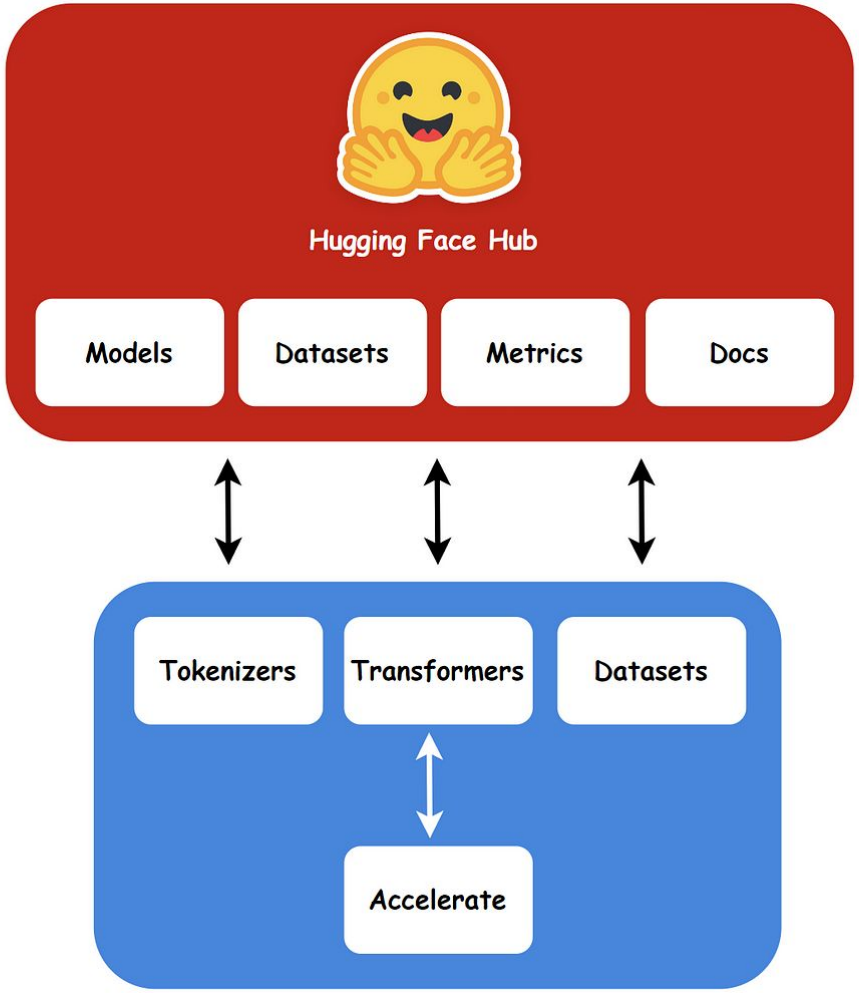
https://huggingface.co/docs/tokenizers/quicktour

Hugging FACE



https://huggingface.co/docs/datasets/quickstart

Hugging FACE



https://huggingface.co/docs/transformers/quicktour