Mini introduction à l'IA : l'apprentissage profond.

Thierry Dumont

8 avril 2024

Plan

- Vers l'apprentissage : les neurones (artificiels...).

Plan

- Vers l'apprentissage : les neurones (artificiels...).
- Apprentissage profond.
  - Réseaux de neurones denses.
  - Réseaux convolutifs.

  Exemples (et démonstrations).

Plan

- Vers l'apprentissage : les neurones (artificiels...).
- Apprentissage profond.
  - Réseaux de neurones denses.
  - Réseaux convolutifs.

  Exemples (et démonstrations).
- Auto encodeurs.

Plan

- Vers l'apprentissage : les neurones (artificiels...).
- Apprentissage profond.
  - Réseaux de neurones denses.
  - Réseaux convolutifs.

  Exemples (et démonstrations).
- Auto encodeurs.
- Traîtement du langage naturel.

Plan

- Vers l'apprentissage : les neurones (artificiels...).
- Apprentissage profond.
  - Réseaux de neurones denses.
  - Réseaux convolutifs.

  Exemples (et démonstrations).
- Auto encodeurs.
- Traîtement du langage naturel.
- Calcul :
  - entraîner = minimiser.
  - ce qui rend les calculs possibles (« heureuses coincidences »).
  - logiciels.

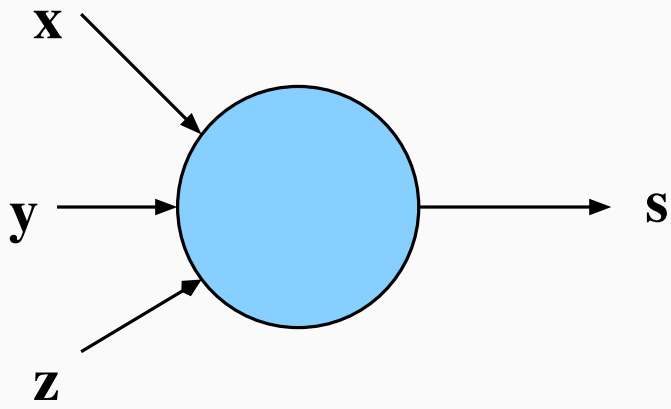
- Dangers.
- Pourquoi est-ce que ça fonctionne ?
- Intelligence ? vraiment ?

- Dangers.
- Pourquoi est-ce que ça fonctionne ?
- Intelligence ? vraiment ?
- Références, lectures plus ou moins recommandables.

Neurones

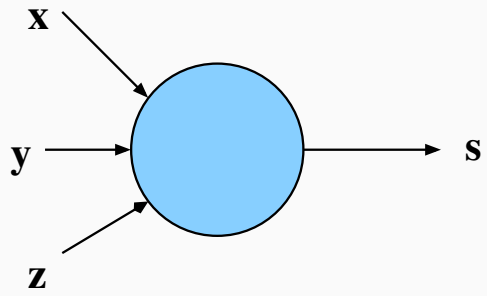
Neurones artificiels

What the frog's eye tells the frog's brain (Warren McCulloch et Walter Pitts, 1959).



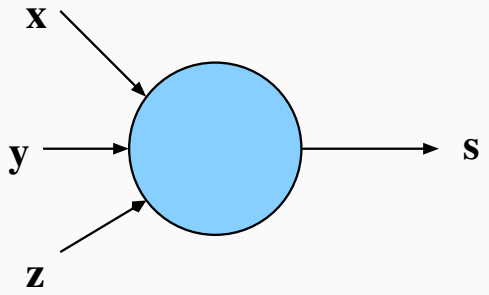
- Mauvaise idée : modèle *linéaire* :

$$s = a_x \times x + a_y \times y + a_z \times z.$$



- Mauvaise idée : modèle *linéaire* :

$$s = a_x \times x + a_y \times y + a_z \times z.$$

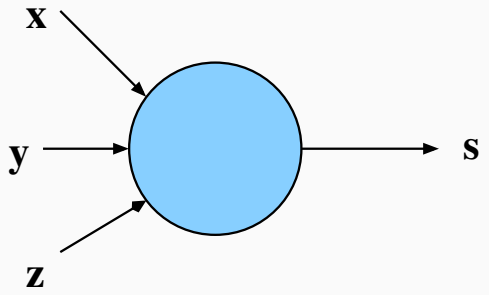


- Autre mauvaise idée : modèle *affine* :

$$s = a_x \times x + a_y \times y + a_z \times z - b.$$

- Mauvaise idée : modèle *linéaire* :

$$s = a_x \times x + a_y \times y + a_z \times z.$$



- Autre mauvaise idée : modèle *affine* :

$$s = a_x \times x + a_y \times y + a_z \times z - b.$$

Il faut une *non-linéarité.*

Le Relu

Rectified Linear Unit

Un modèle très général

$$s = (a_x \times x + a_y \times y + a_z \times z - b)^+$$

Le Relu

Rectified Linear Unit

Un modèle très général

$$s = (a_x \times x + a_y \times y + a_z \times z - b)^+$$

soit encore :

$$s = \max(0, a_x \times x + a_y \times y + a_z \times z - b).$$

Le Relu

Rectified Linear Unit

Un modèle très général

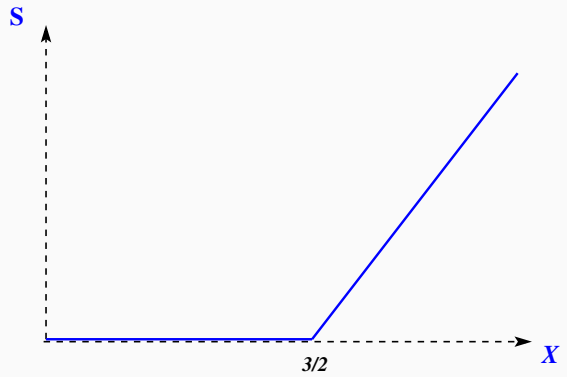
$$s = (a_x \times x + a_y \times y + a_z \times z - b)^+$$

soit encore :

$$s = \max(0, a_x \times x + a_y \times y + a_z \times z - b).$$

Exemple (avec une seule entrée) :

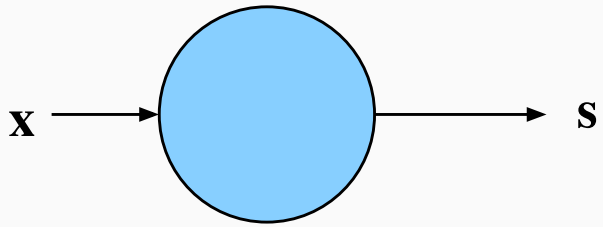
$s = (2 \times x - 3)^+$



C'est **LA** brique de base de tout ! Pourquoi ?

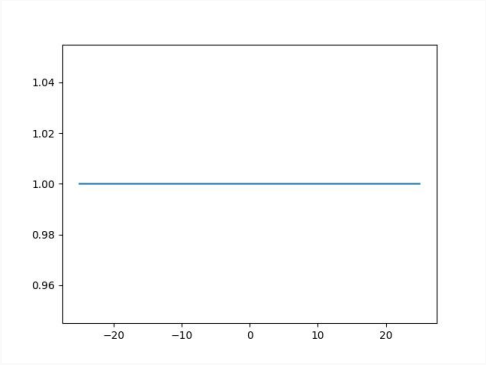
C'est **LA** brique de base de tout ! Pourquoi ?

Regardons des neurones avec une seule entrée :

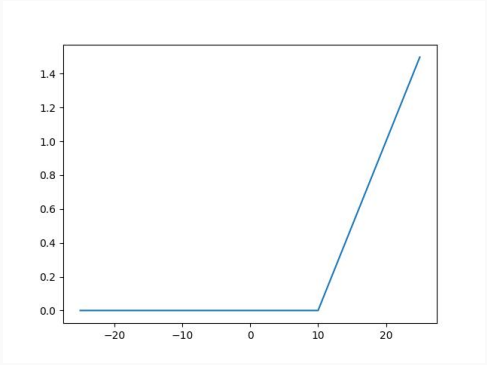


Une certaine universalité

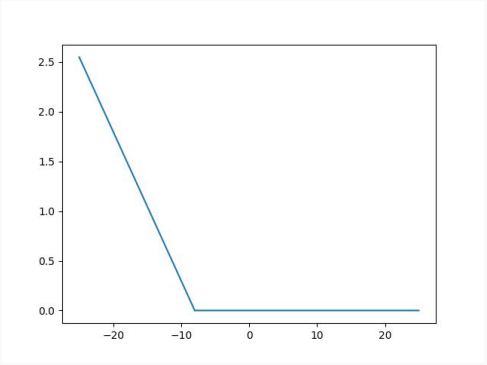
Toute fonction continue (qui peut être tracée sans lever la main) peut être approchée d'aussi près qu'on veut par une combinaison de neurones (ReLU) (Cybenko et al, 1989 ->).



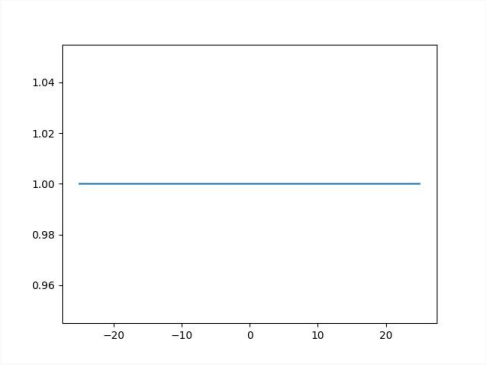
(A) $0\ x + 1$



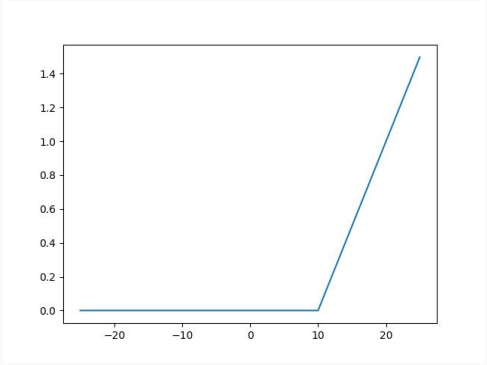
(B) $0.1\ x - 1$



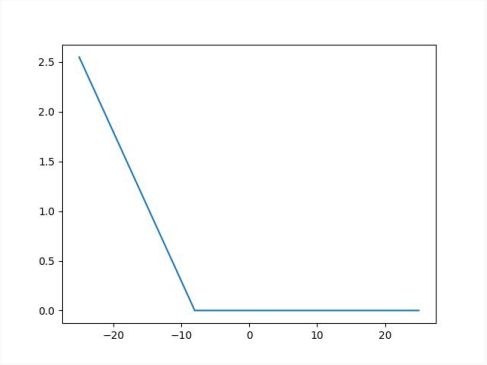
(C) $-0.15\ x - 1.2$



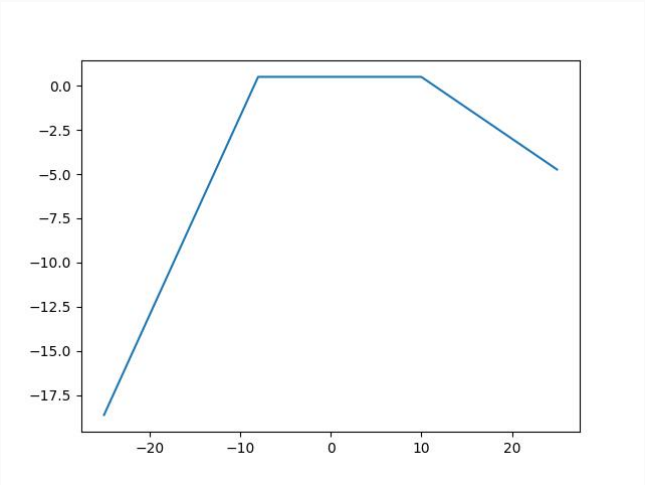
(A) $0\ x + 1$



(B) $0.1\ x - 1$



(C) $-0.15\ x - 1.2$

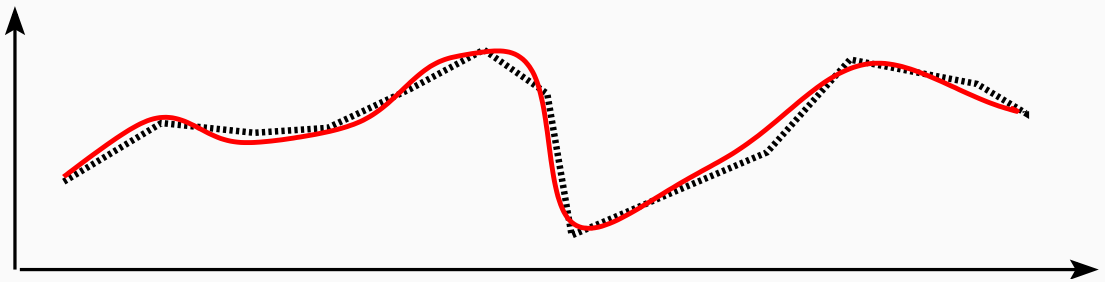


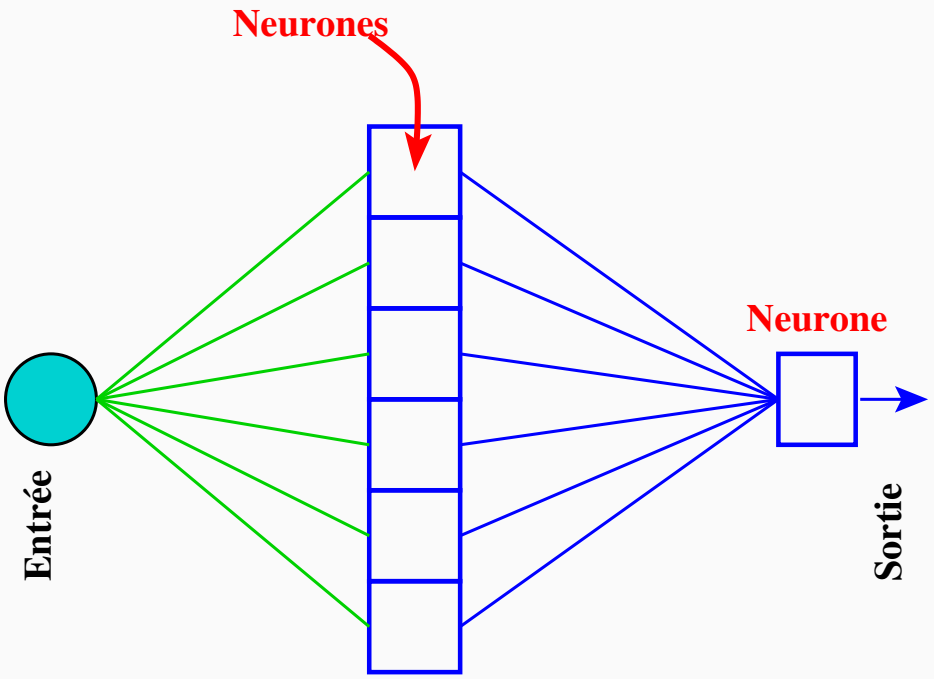
$0.5 \times (A) - 3.5 \times (B) - 7.5 \times (C)$

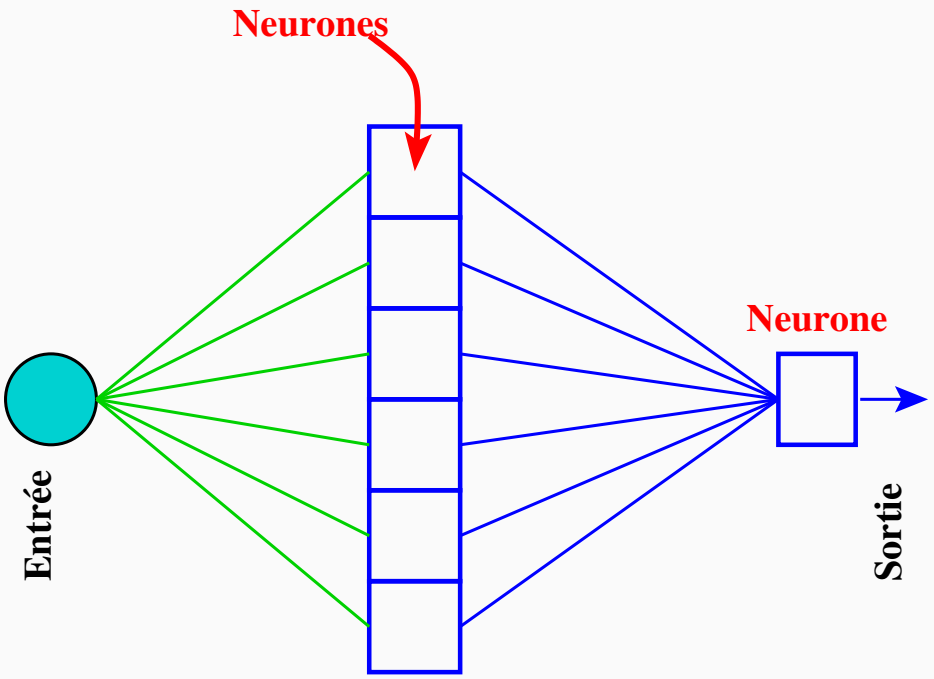
Difficile de fabriquer la courbe qu'on veut !

Combiner les neurones (ReLU) permet de fabriquer toutes les lignes brisées qu'on veut...

Combiner les neurones (ReLU) permet de fabriquer toutes les lignes brisées qu'on veut...




Neurones
Neurone
Entrée
Sortie

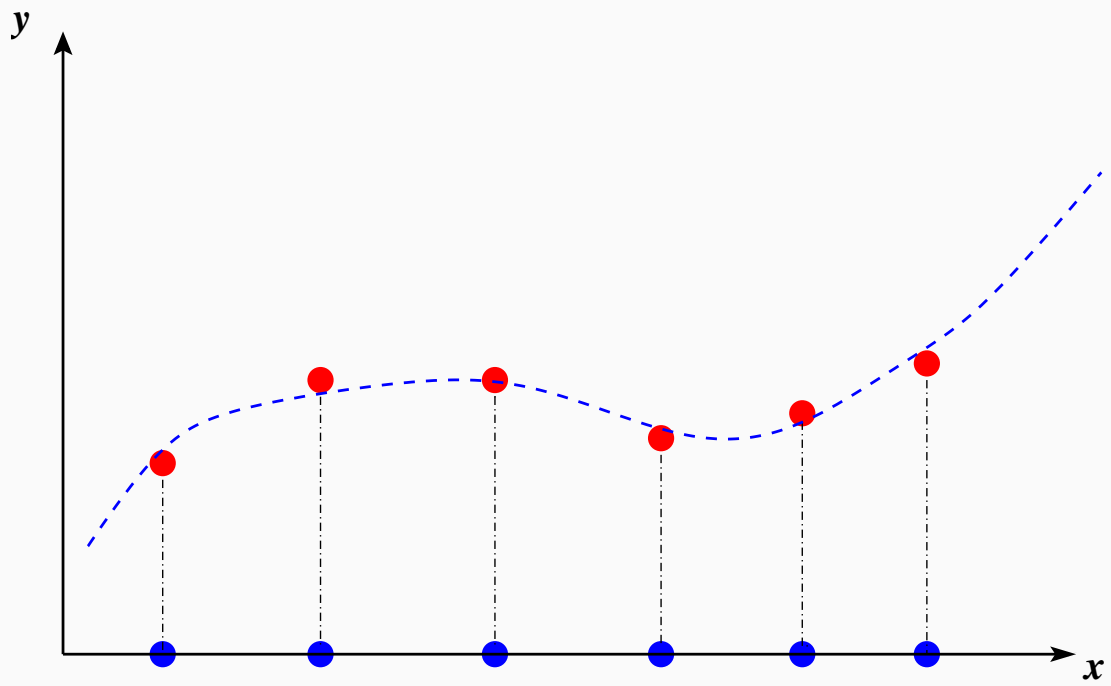


Nombre de coefficients :

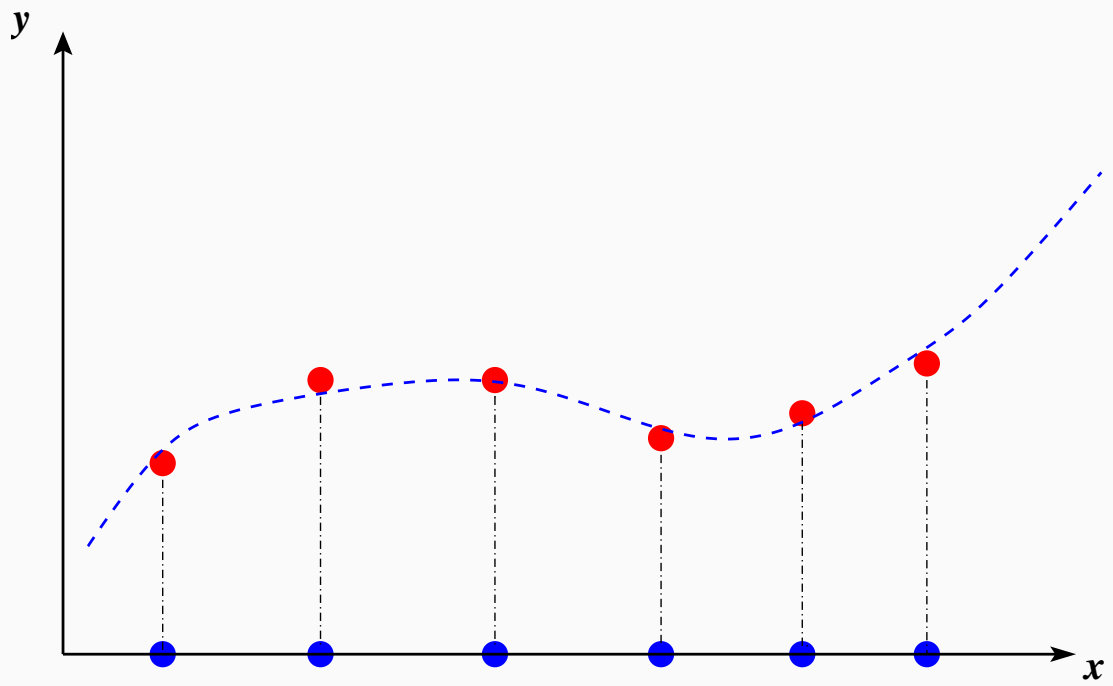
- Première couche : $6 \times 2 = 12$.
- Sortie : 6 (pas de ReLu).

soit en tout 18 coefficients.

« Apprendre une courbe »

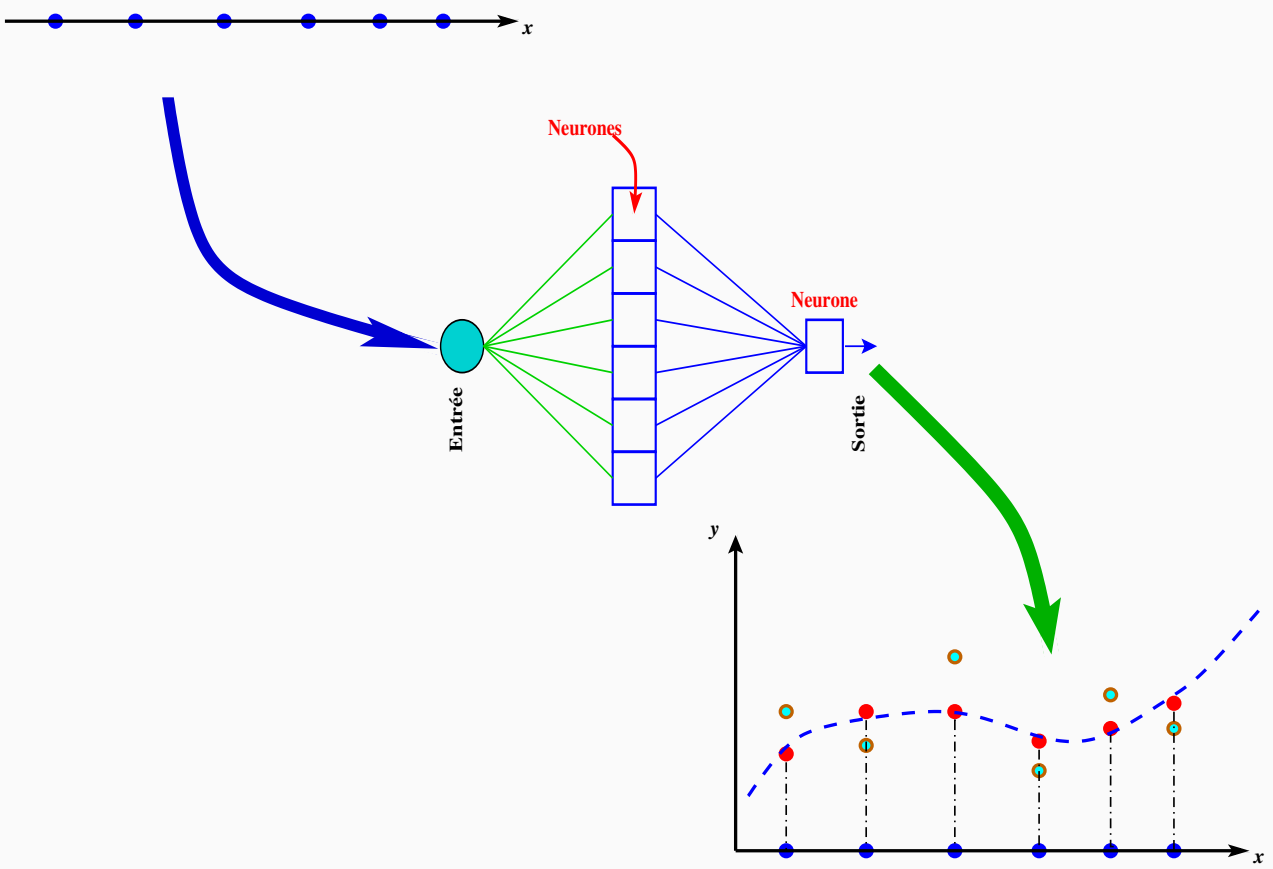


« Apprendre une courbe »

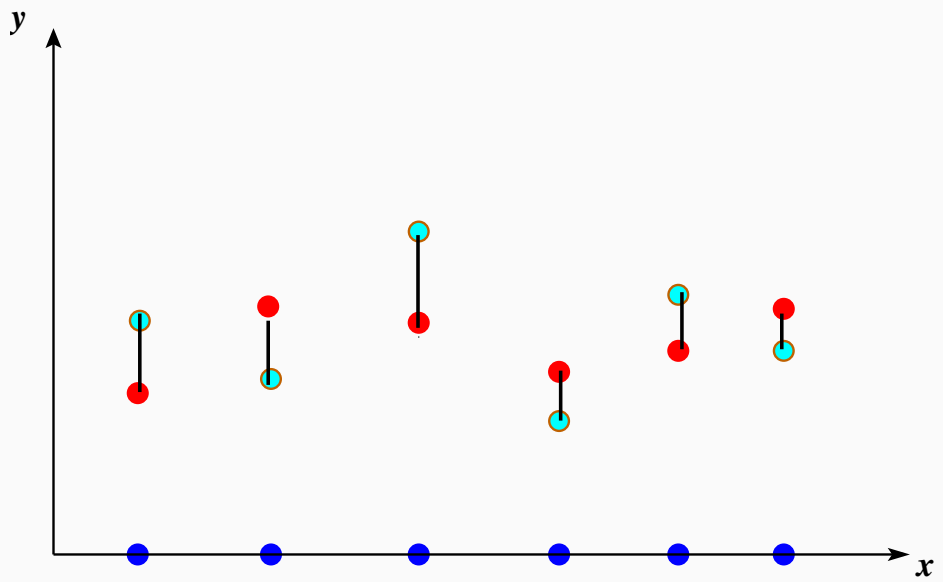


Données : des couples (x, y).

Apprendre *par les données.*

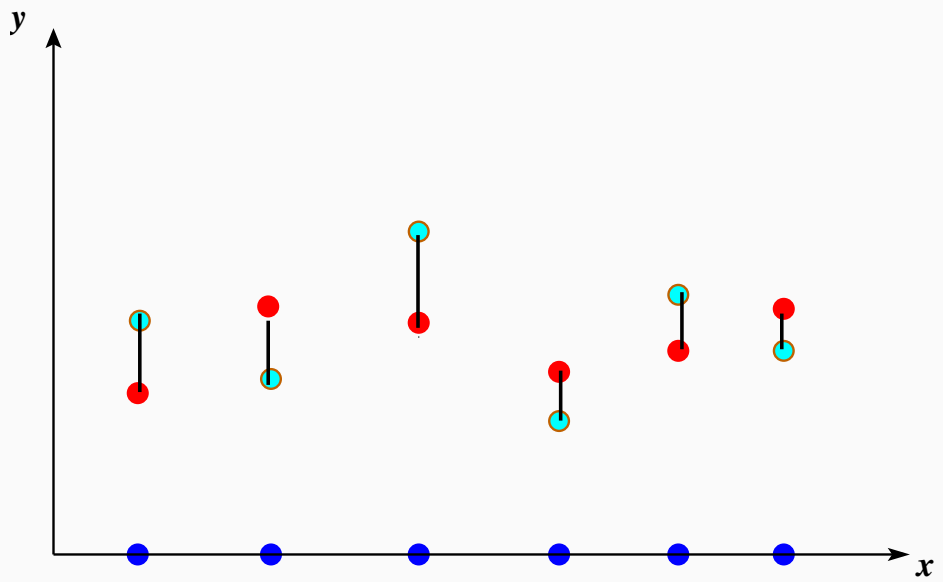


Points prévus, points observés



Apprendre =

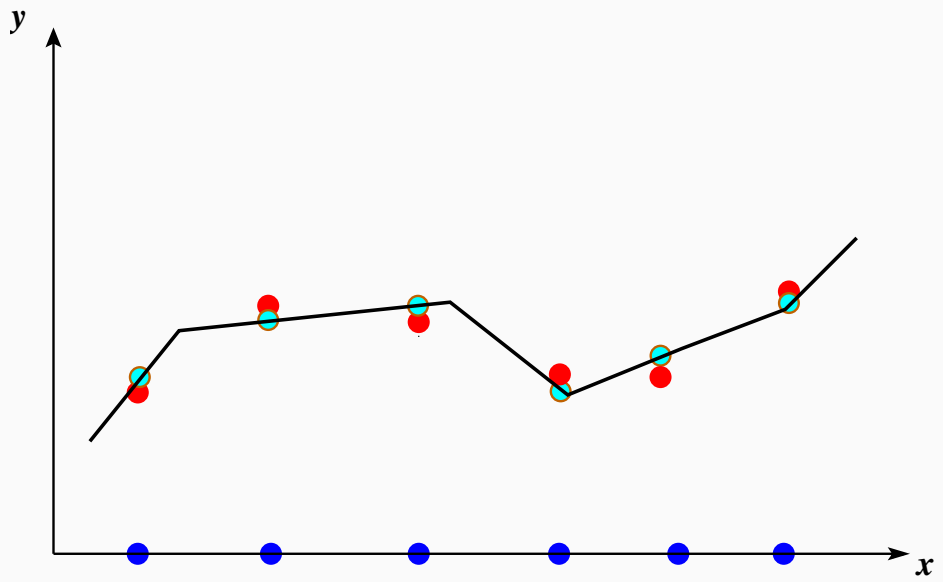
- Choisir les paramètres des neurones pour minimiser une distance entre points prévus et points observés.



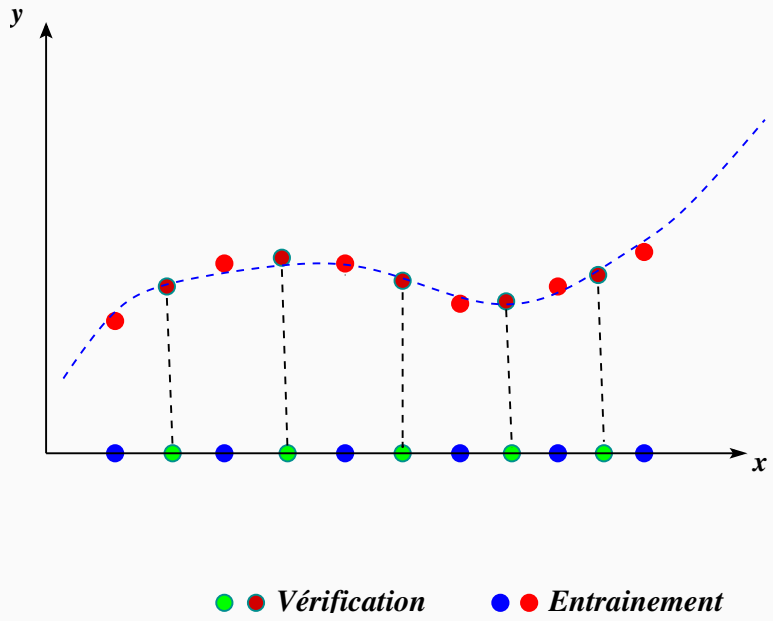
Apprendre =

- Choisir les paramètres des neurones pour minimiser une distance entre points prévus et points observés.

Exemple de distance : la somme des carrés des distances entre points prévus et points observés.


y
x

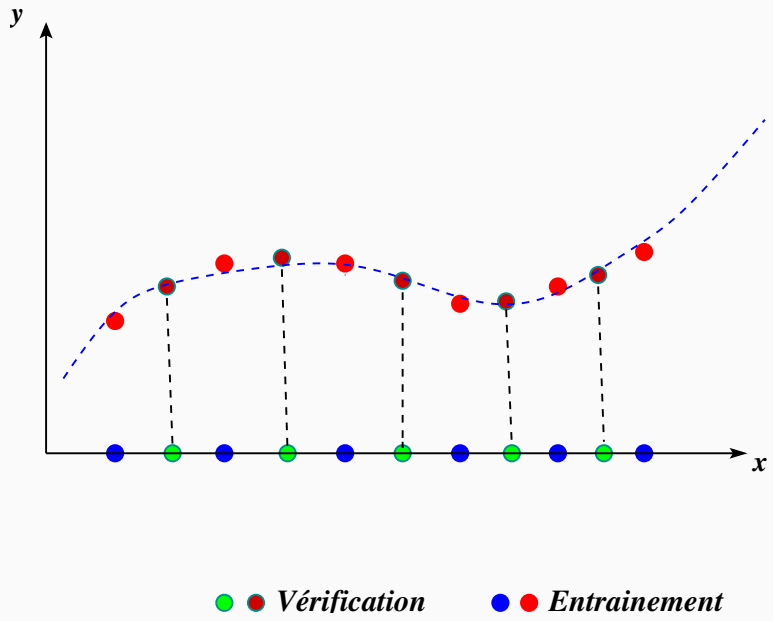
Vérifier ?



Il faut un ensemble de couples $(x, y) =$ (donnée, résultat), partitionné en :

1. des couples pour l'apprentissage,
2. des couples pour vérifier.

Vérifier ?



Il faut un ensemble de couples $(x, y) =$ (donnée, résultat), partitionné en :

1. des couples pour l'apprentissage,
2. des couples pour vérifier.

Apprentissage **supervisé**.

Questions importantes :

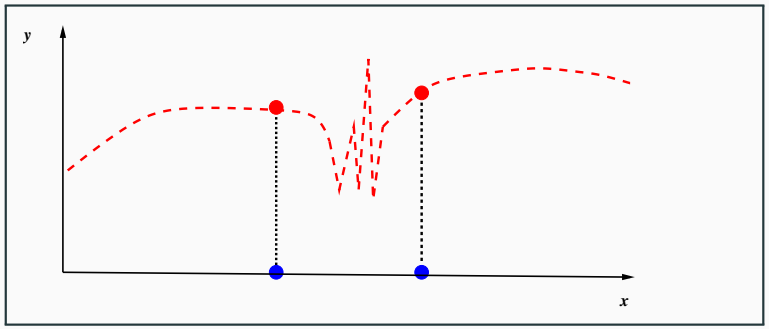
- Comment choisir le nombre de neurones ?

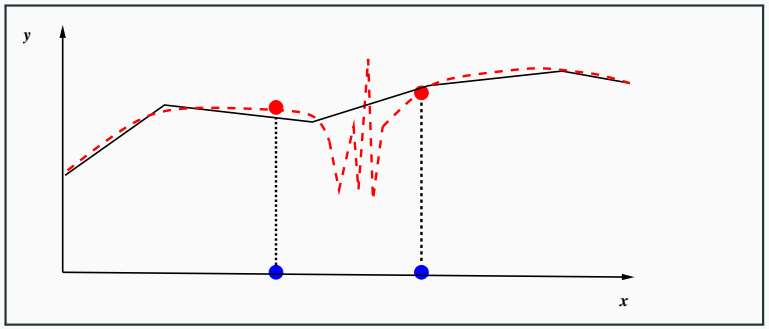
Questions importantes :

- Comment choisir le nombre de neurones ?
- Fiabilité (de la vérification) ? (question de régularité).

Questions importantes :

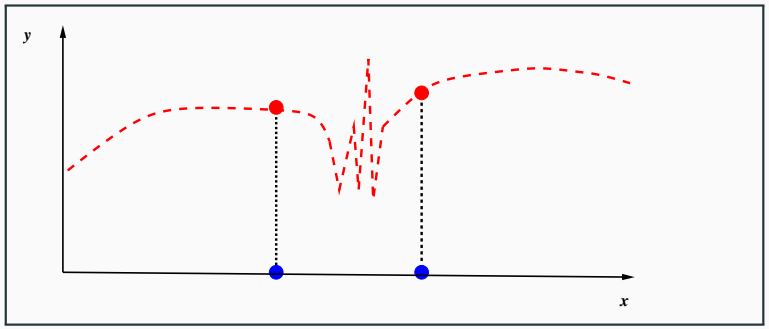
- Comment choisir le nombre de neurones ?
- Fiabilité (de la vérification) ? (question de régularité).

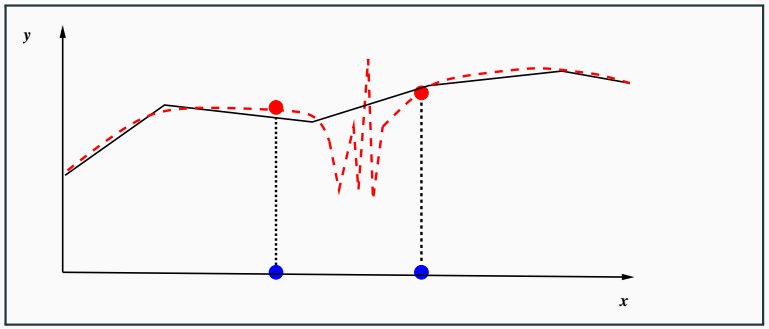




Questions importantes :

- Comment choisir le nombre de neurones ?
- Fiabilité (de la vérification) ? (question de régularité).

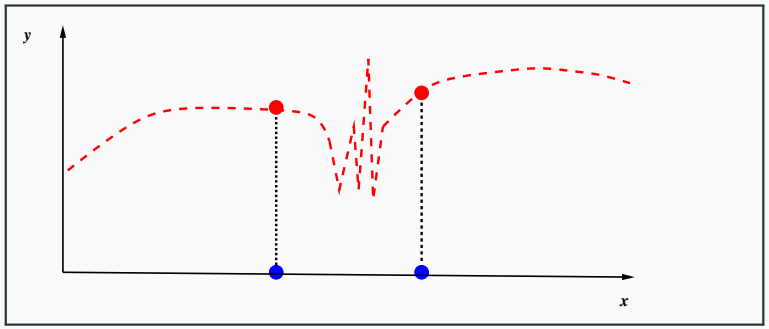


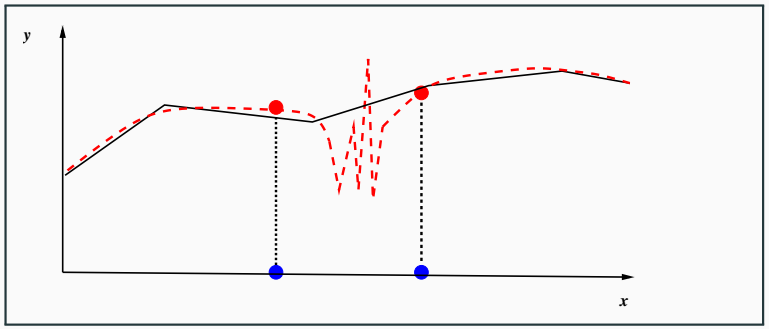


- Extrapoler est sûrement dangereux !

Questions importantes :

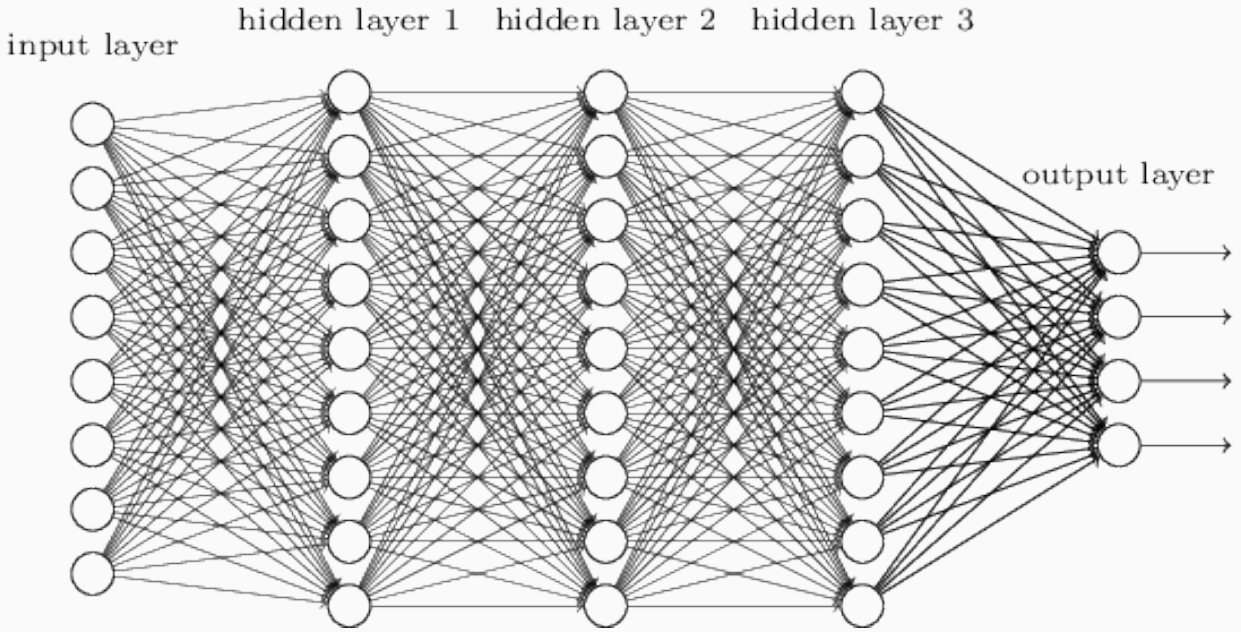
- Comment choisir le nombre de neurones ?
- Fiabilité (de la vérification) ? (question de régularité).



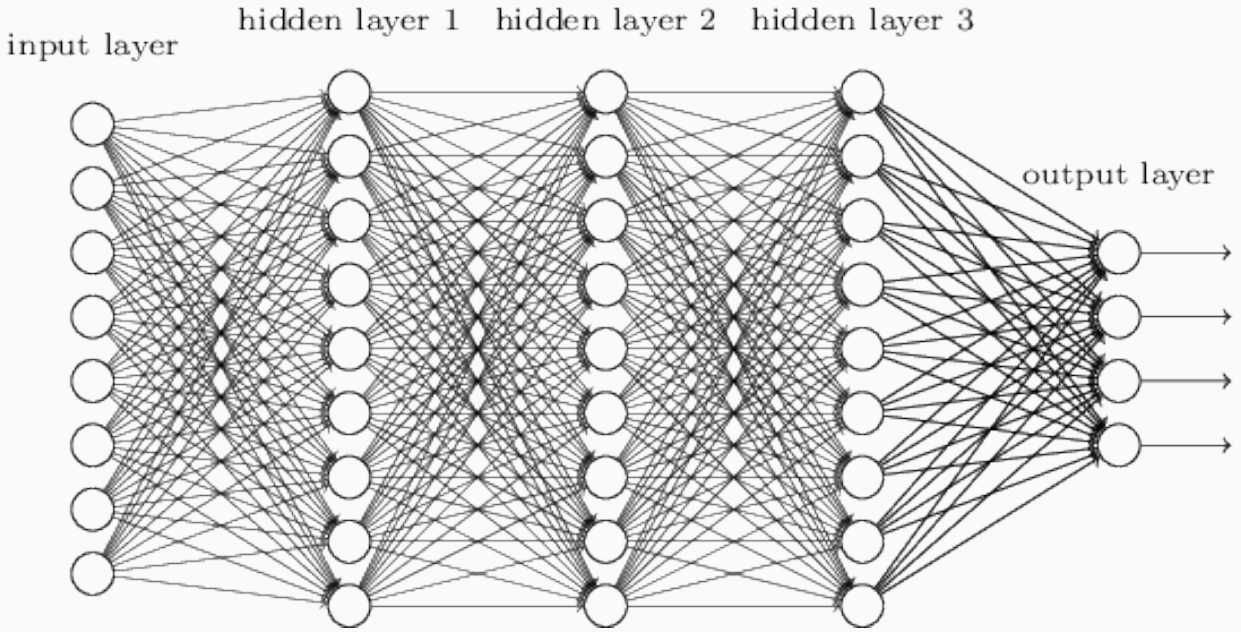


- Extrapoler est sûrement dangereux !
- Comment calculer les paramètres des neurones qui minimisent la distance (calcul/observation) ?

Apprentissage profond



Réseau de neurones profond (3 couches cachées), *dense*.



Réseau de neurones profond (3 couches cachées), *dense*.

C'est l'apprentissage profond (ajouter une ou plusieurs couches de neurones) qui a tout changé !

Un exemple œnologique

- 1598 vins sont testés et notés (Q) de 0 à 10.
- 12 paramètres sont mesurés.

n	ac.	v.ac.	cita.	sugar	chlor.	sulf.	totalsul.	dens.	pH	sulph.	alc.	Q
1589	6.60	0.72	0.20	7.80	0.07	29.00	79.00	1.00	3.29	0.54	9.20	5.00
1590	6.30	0.55	0.15	1.80	0.08	26.00	35.00	0.99	3.32	0.82	11.60	6.00
1591	5.40	0.74	0.09	1.70	0.09	16.00	26.00	0.99	3.67	0.56	11.60	6.00
1592	6.30	0.51	0.13	2.30	0.08	29.00	40.00	1.00	3.42	0.75	11.00	6.00
1593	6.80	0.62	0.08	1.90	0.07	28.00	38.00	1.00	3.42	0.82	9.50	6.00
1594	6.20	0.60	0.08	2.00	0.09	32.00	44.00	0.99	3.45	0.58	10.50	5.00
1595	5.90	0.55	0.10	2.20	0.06	39.00	51.00	1.00	3.52	0.76	11.20	6.00
1596	6.30	0.51	0.13	2.30	0.08	29.00	40.00	1.00	3.42	0.75	11.00	6.00
1597	5.90	0.65	0.12	2.00	0.07	32.00	44.00	1.00	3.57	0.71	10.20	5.00
1598	6.00	0.31	0.47	3.60	0.07	18.00	42.00	1.00	3.39	0.66	11.00	6.00

Peut-on réaliser un système qui prévoit la qualité (Q) du vin en fonction des 12 paramètres ?

Le réseau

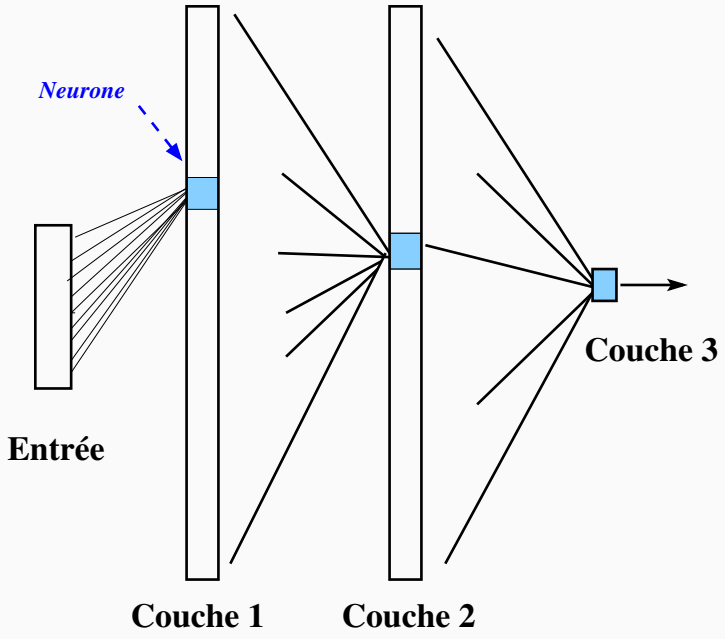
Réseau à 2 couches cachées, dense.

1. Entrée : 12 valeurs.
2. 1re couche : 64 neurones.
3. 2e couche : 64 neurones.
4. 3e couche : 1 neurone,
   pas de ReLu.

Le réseau

Réseau à 2 couches cachées, dense.

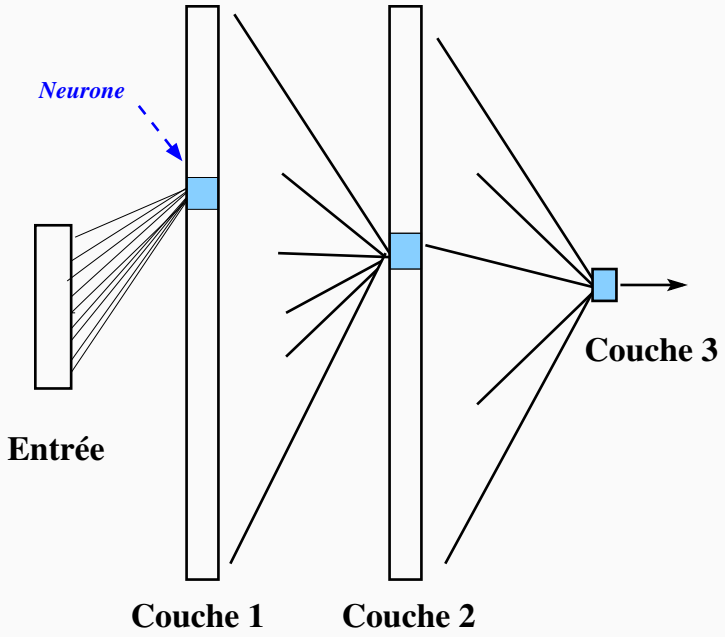
1. Entrée : 12 valeurs.
2. 1re couche : 64 neurones.
3. 2e couche : 64 neurones.
4. 3e couche : 1 neurone, pas de ReLu.



Le réseau

Réseau à 2 couches cachées, dense.

1. Entrée : 12 valeurs.
2. 1re couche : 64 neurones.
3. 2e couche : 64 neurones.
4. 3e couche : 1 neurone, pas de ReLu.

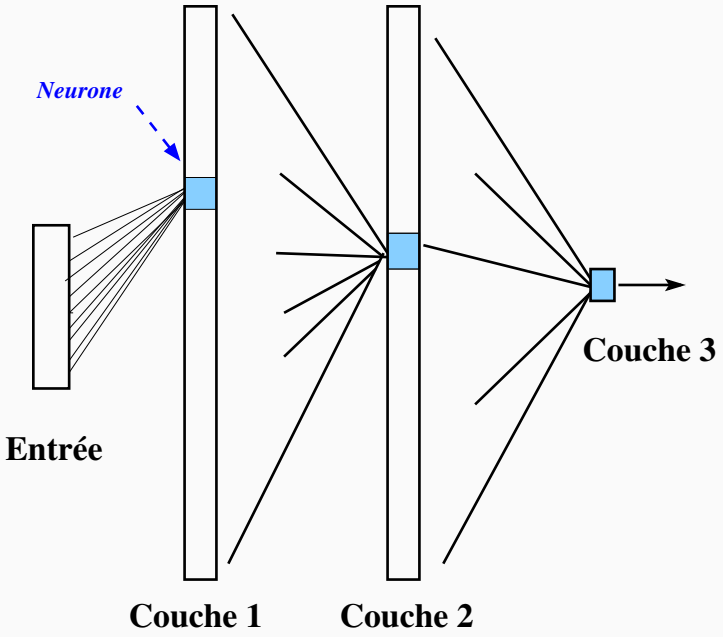


- Chaque neurone de la couche 1 : couplé aux 12 entrées.

Le réseau

Réseau à 2 couches cachées, dense.

1. Entrée : 12 valeurs.
2. 1re couche : 64 neurones.
3. 2e couche : 64 neurones.
4. 3e couche : 1 neurone, pas de ReLu.

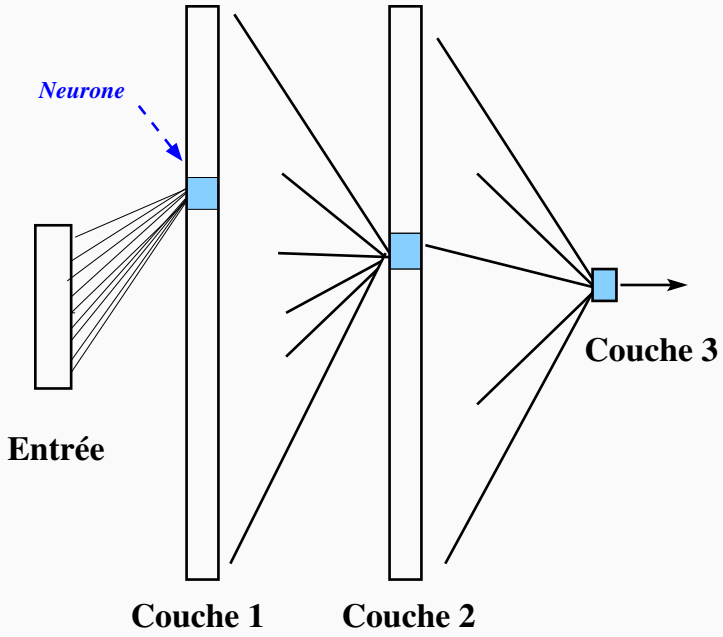


- Chaque neurone de la couche 1 : couplé aux 12 entrées.
- Chaque neurone de la couche 2 : couplé aux 64 sorties de la couche 1.

Le réseau

Réseau à 2 couches cachées, dense.

1. Entrée : 12 valeurs.
2. 1re couche : 64 neurones.
3. 2e couche : 64 neurones.
4. 3e couche : 1 neurone, pas de ReLu.



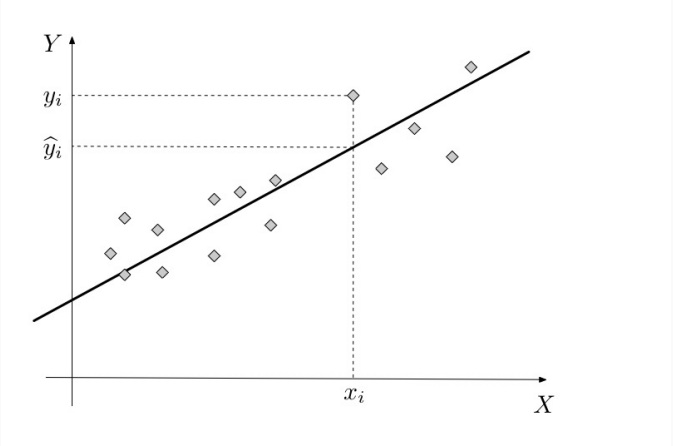
- Chaque neurone de la couche 1 : couplé aux 12 entrées.
- Chaque neurone de la couche 2 : couplé aux 64 sorties de la couche 1.
- Le neurone de la couche 3 est couplé aux 64 sorties de la couche 2.

Soit :

$64 \times 12 + 64 \times 65 + 65 = 4993$ paramètres à ajuster.

Apprentissage = optimisation

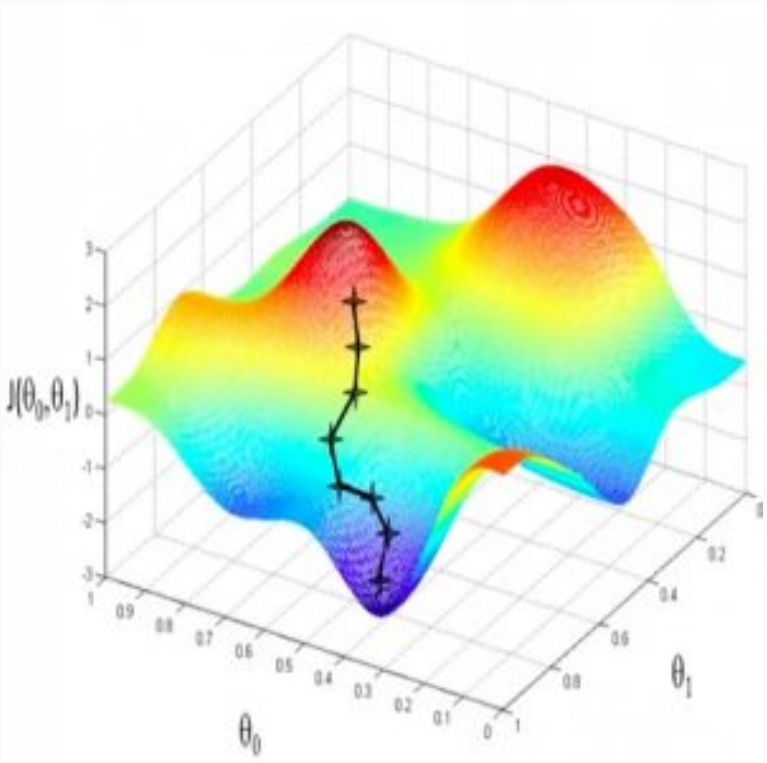
Un problème beaucoup plus difficile qu'un ajustement linéaire aux moindres carrés (droite aux moindres carrés par exemple).



Algorithme glouton : descente de gradient

Algorithme glouton : descente de gradient

On progresse par pas dans la direction de la plus grande pente :



Aucune garantie d'atteindre le vrai minimum.

- Calcul du gradient (direction de la plus forte pente) : un problème d'une complexité redoutable ! (la quantité de calculs à effectuer est énorme).

- Calcul du gradient (direction de la plus forte pente) : un problème d'une complexité redoutable ! (la quantité de calculs à effectuer est énorme).

  La difficulté provient principalement de la non-linéarité (ReLu ou autre).

- Gradient stochastique : à chaque pas on tire au hasard *quelques paramètres (2 au moins)*, et on *oublie* les autres.

- Gradient stochastique : à chaque pas on tire au hasard *quelques paramètres (2 au moins)*, et on *oublie* les autres.

Convergence *très* lente.

Important ! Propriété fondamentale !

- L'entraînement (= optimisation) est une opération extrêmement coûteuse.
- Mais l'utilisation du réseau entraîné est peu coûteuse.

Important ! Propriété fondamentale !

- L'entraînement (= optimisation) est une opération extrêmement coûteuse.
- Mais l'utilisation du réseau entraîné est peu coûteuse.

Le coût en nombre d'opérations de l'utilisation est de l'ordre du nombre de paramètres du réseau.

Il est temps de regarder notre exemple œnologique

Réseaux de neurones convolutifs

Problème : une image = plusieurs millions de pixels => réseau de neurones « classique » inutilisable.

Problème : une image = plusieurs millions de pixels => réseau de neurones « classique » inutilisable.

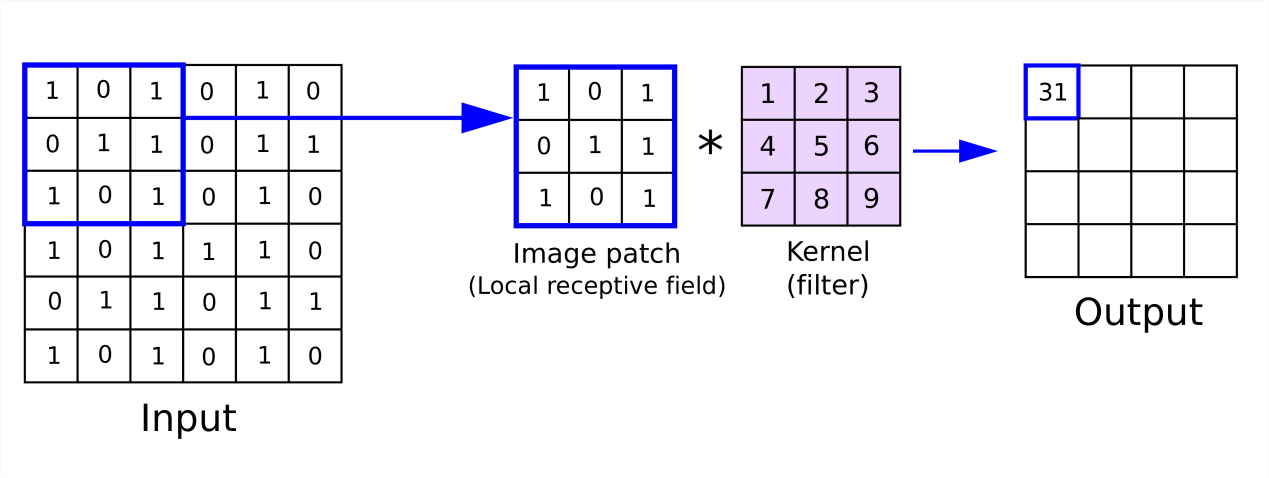
Remèdes :

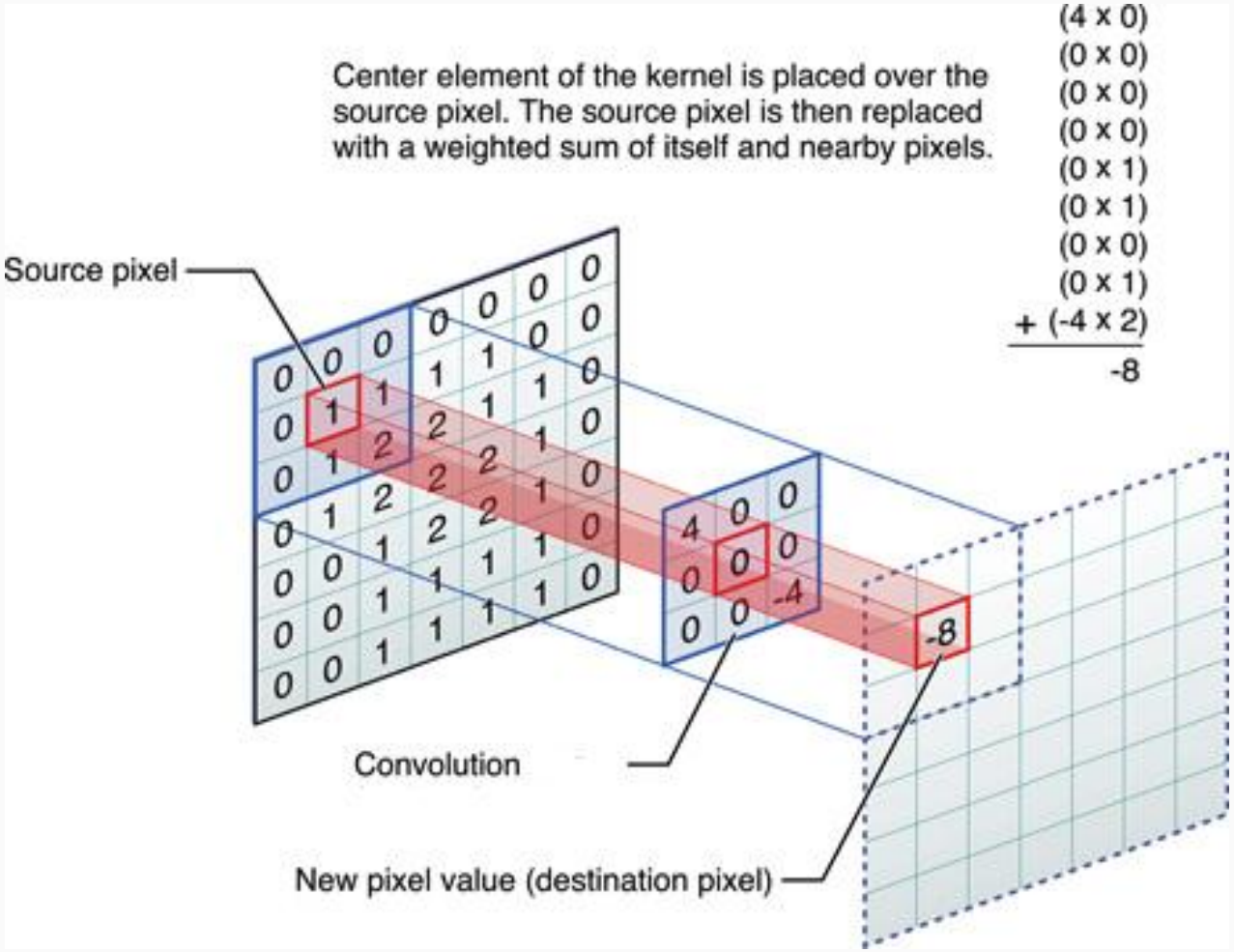
1. simplifier : couche de neurones :
   - faiblement liés aux entrées,
   - partageant les mêmes paramètres.
2. utiliser les acquis du traitement d'images :

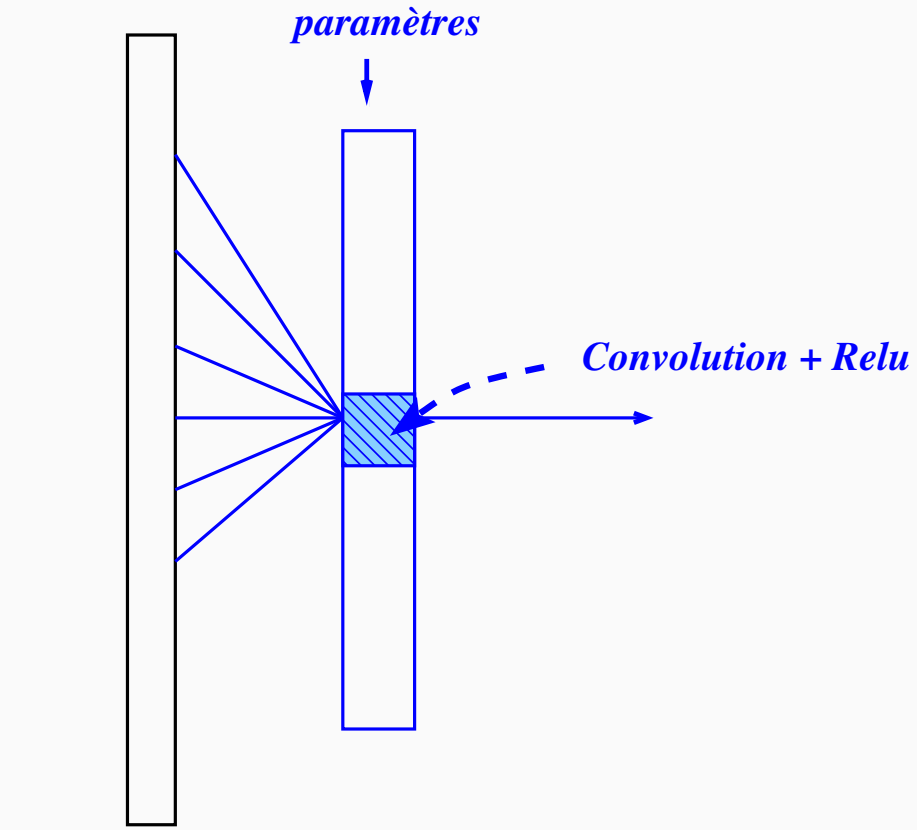
Problème : une image = plusieurs millions de pixels => réseau de neurones « classique » inutilisable.

Remèdes :

1. simplifier : couche de neurones :
   - faiblement liés aux entrées,
   - partageant les mêmes paramètres.
2. utiliser les acquis du traitement d'images : traitement *local*, identique en chaque point de l'image.


1 0 1 0 1 0
0 1 1 0 1 1
1 0 1 0 1 0
1 0 1 1 1 0
0 1 1 0 1 1
1 0 1 0 1 0
Input
1 0 1
0 1 1
1 0 1
Image patch
(Local receptive field)
*
1 2 3
4 5 6
7 8 9
Kernel
(filter)
31
Output


Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8
Source pixel
Convolution
New pixel value (destination pixel)


paramètres
Convolution + Relu
image -> vecteur

Rappel : pour l'apprentissage, les coefficients des noyaux de convolution sont des inconnues.

Un exemple : reconnaissance de chiffres manuscrits

Un peu simplifié ! Images 28×28, N&B.

```
model.add( keras.layers.Input((28,28,1)) )

model.add( keras.layers.Conv2D(8, (3,3),  activation='relu') )
model.add( keras.layers.MaxPooling2D((2,2)))
model.add( keras.layers.Dropout(0.2))

model.add( keras.layers.Conv2D(16, (3,3), activation='relu') )
model.add( keras.layers.MaxPooling2D((2,2)))
model.add( keras.layers.Dropout(0.2))

model.add( keras.layers.Flatten())
model.add( keras.layers.Dense(100, activation='relu'))
model.add( keras.layers.Dropout(0.5))

model.add( keras.layers.Dense(10, activation='softmax'))
```

- 3 couches internes (2 convolutions, 1 dense).
- MaxPooling et Dropout.
- Softmax.

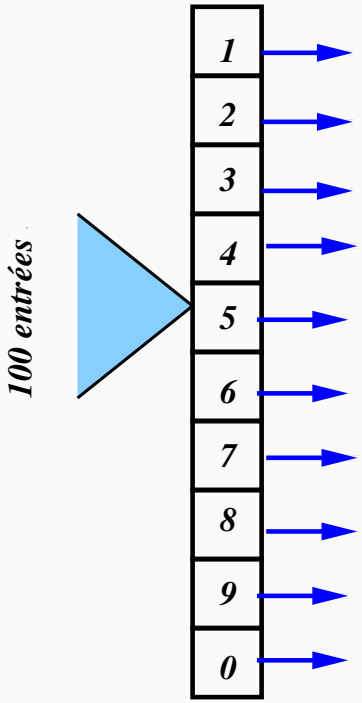
Softmax

La dernière couche : 10 neurones qui fournissent 10 valeurs.

```
model.add( keras.layers.Dense(10, activation='softmax'))
```

Les autres couches avaient un *ReLu*, ici on a un *softmax.*

La dernière couche



On voudrait que :

1. quand on entre une image de “5”, le neurone 5 fournisse 1 et les autres 0.
2. ... et en tout cas que tous fournissent des valeurs entre 0 et 1 (un score).

Exemple, en sortie :

$z = 0.1,\ 0.3,\ 0.2,\ 0.6,\ 0.01,\ 5.20,\ 0.3,\ -0.2,\ 0.05,\ -0.1$

Exemple, en sortie :

$z = 0.1,\ 0.3,\ 0.2,\ 0.6,\ 0.01,\ 5.20,\ 0.3,\ -0.2,\ 0.05,\ -0.1$

$$\mathrm{Softmax}(z_i) = \frac{\exp z_i}{\sum_i \exp z_i}.$$

Exemple, en sortie :

$z = 0.1,\ 0.3,\ 0.2,\ 0.6,\ 0.01,\ 5.20,\ 0.3,\ -0.2,\ 0.05,\ -0.1$

$$\mathrm{Softmax}(z_i) = \frac{\exp z_i}{\sum_i \exp z_i}.$$

- $\exp(x)$ $(= e^x)$ est toujours positive !
- $0 \leq \mathrm{Softmax}(z_i) \leq 1$.
- $\sum_i \mathrm{Softmax}(z_i) = 1$.
- On éxagère les grandes valeurs (absolues).

Exemple, en sortie :

$z = 0.1,\ 0.3,\ 0.2,\ 0.6,\ 0.01,\ 5.20,\ 0.3,\ -0.2,\ 0.05,\ -0.1$

$$\mathrm{Softmax}(z_i) = \frac{\exp z_i}{\sum_i \exp z_i}.$$

- $\exp(x)$ $(= e^x)$ est toujours positive !
- $0 \leq \mathrm{Softmax}(z_i) \leq 1$.
- $\sum_i \mathrm{Softmax}(z_i) = 1$.
- On éxagère les grandes valeurs (absolues).

Ici, $\mathrm{Softmax}$ donne :

$0.00575893, 0.00703398, 0.0063646, 0.00949487, 0.00526327,$

$0.94459097, 0.00703398, 0.00426632, 0.00547807, 0.00471501.$

Attention !

$0.00575893, 0.00703398, 0.0063646, 0.00949487, 0.00526327,$

$0.94459097, 0.00703398, 0.00426632, 0.00547807, 0.00471501.$

qu'on assimile à une probabilité.

Attention !

$0.00575893, 0.00703398, 0.0063646, 0.00949487, 0.00526327,$

$0.94459097, 0.00703398, 0.00426632, 0.00547807, 0.00471501.$

qu'on assimile à une probabilité.

> **À priori, aucun résultat d'un processus de classification n'est sûr à 100 % !**

Un autre exemple : l'OCR *Tesseract*

Un autre exemple : l'OCR *Tesseract*

Principe :

- Caractère = nombre en machine
  (Exemple : LATIN CAPITAL LETTER A -> 65).

Un autre exemple : l'OCR *Tesseract*

Principe :

- Caractère = nombre en machine
  (Exemple : LATIN CAPITAL LETTER A -> 65).
- On prend des textes en docx, pdf, html, etc. et on rend les caractères sous forme d'images avec différentes polices de caractères.

Un autre exemple : l'OCR *Tesseract*

Principe :

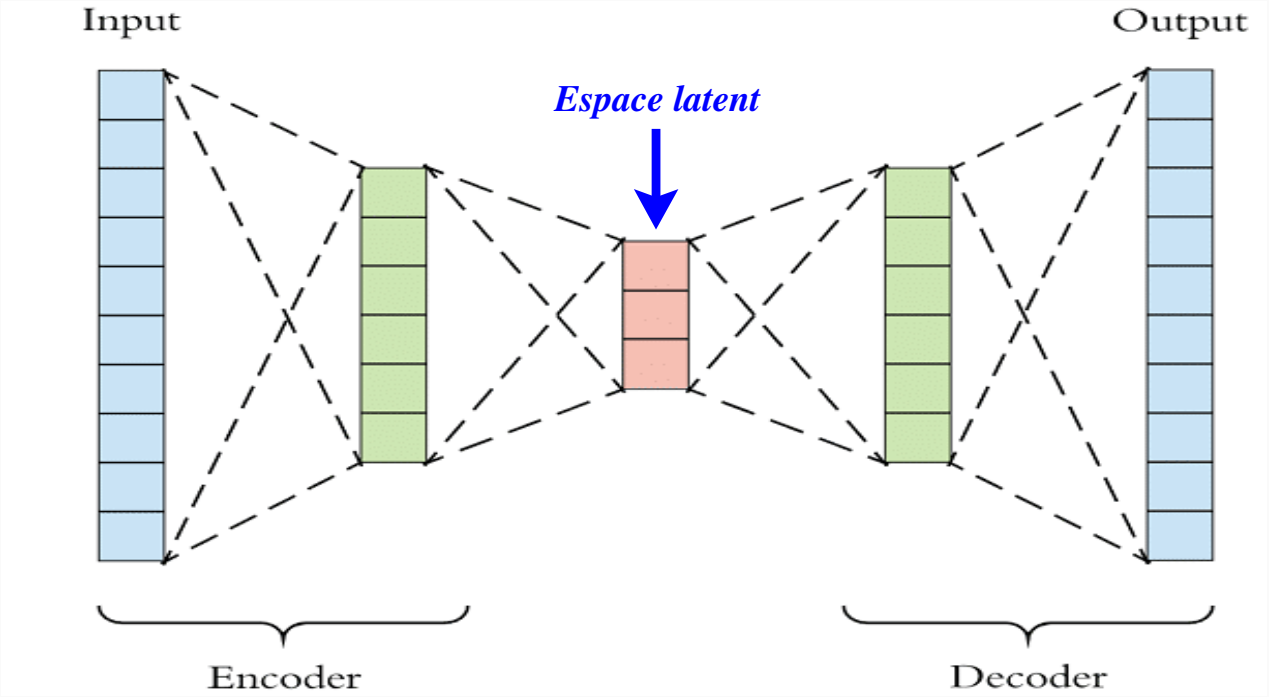
- Caractère = nombre en machine
  (Exemple : LATIN CAPITAL LETTER A -> 65).
- On prend des textes en docx, pdf, html, etc. et on rend les caractères sous forme d'images avec différentes polices de caractères.
- On a donc un grand nombre de correspondances :
  (image) -> (nombre = caractère)
  pour entraîner l'OCR.

Entraînement :

- 400 000 lignes de texte,
- 4500 polices de caractères.

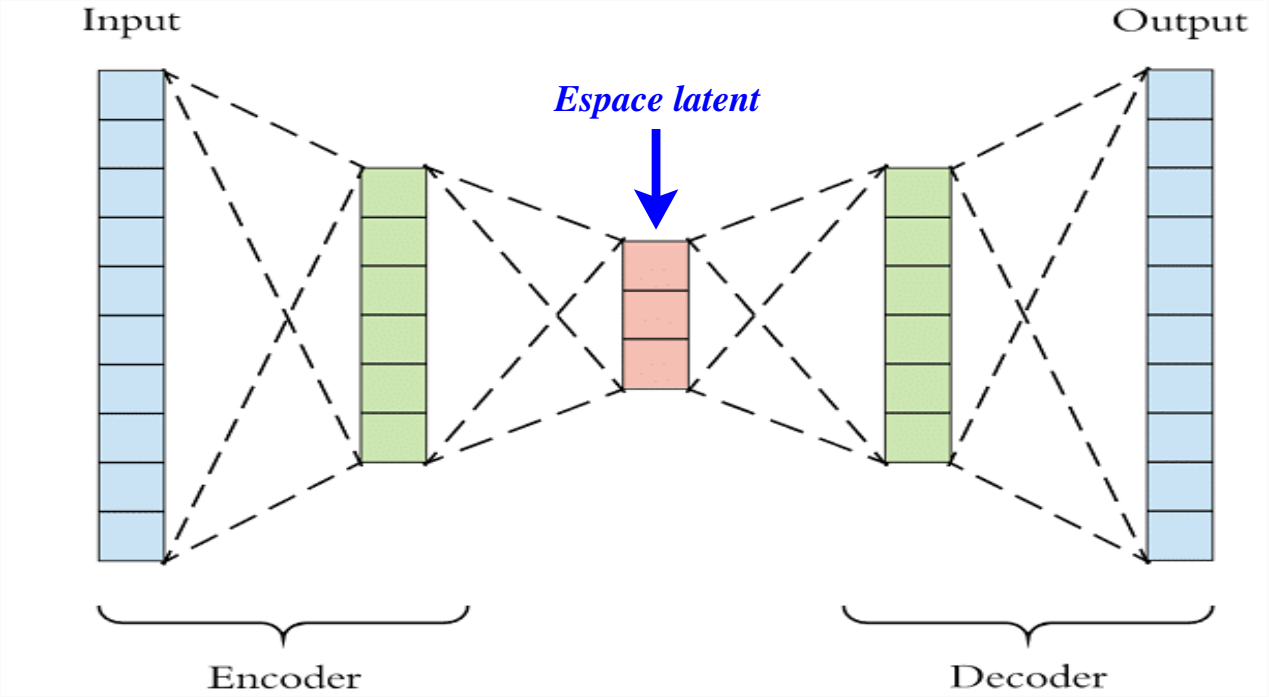
- *de l'ordre de 10 milliards d'entrées pour l'entraînement et la vérification !*
- *Plusieurs jours (semaines) de calcul !*

Auto encodeurs

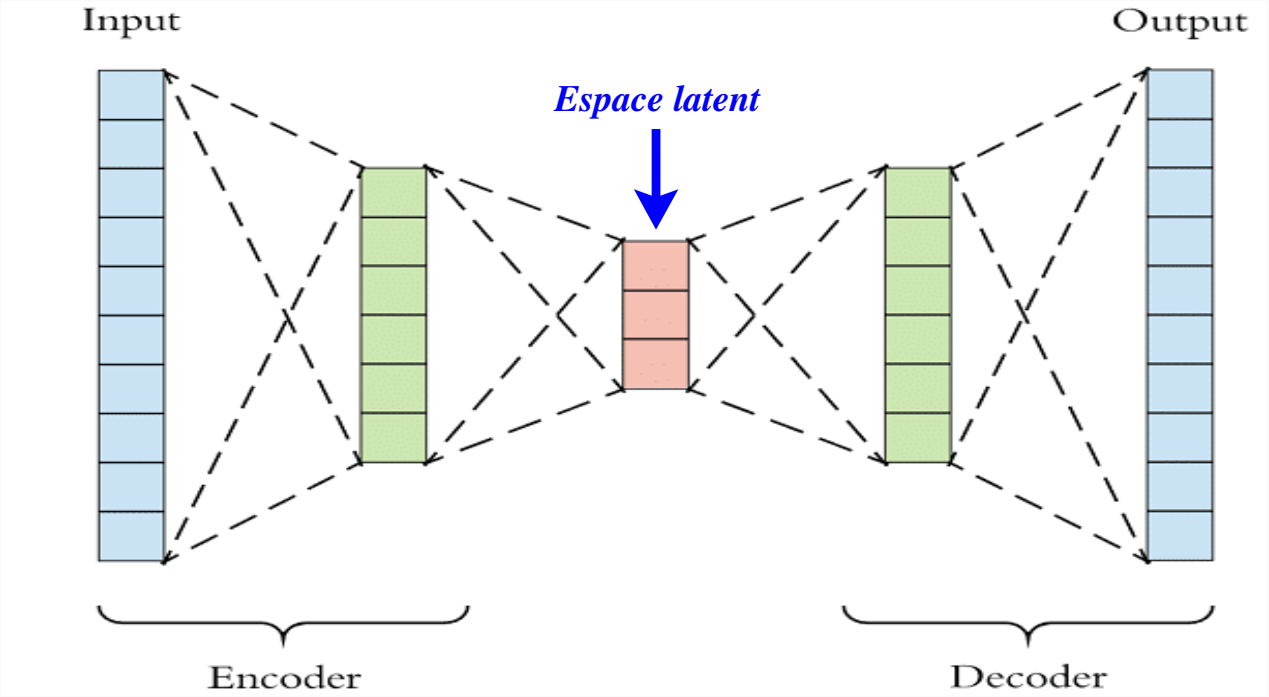


- En entrée : disons, une image.
- En sortie : retrouver *presque* la même image.

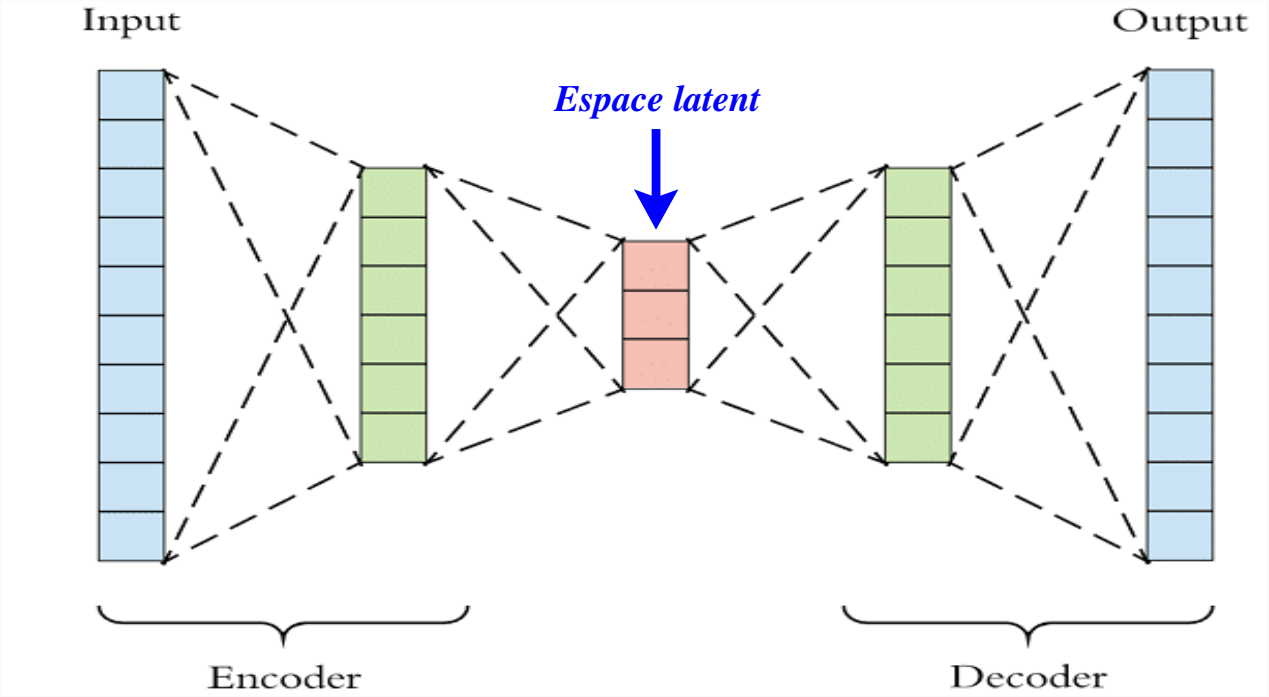
en passant par un espace de petite dimension.



- Image : millions de pixels = millions de valeurs.

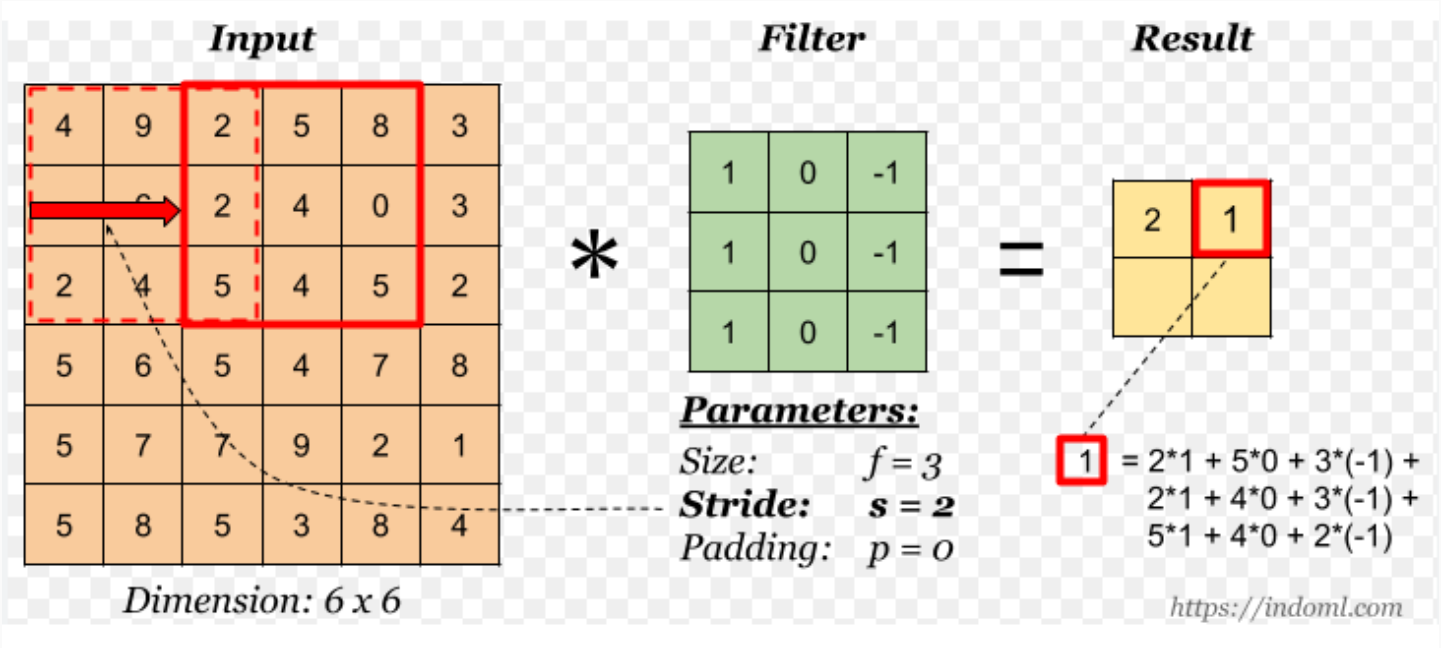


- Image : millions de pixels = millions de valeurs.
- Image = un point dans un espace à quelques millions de dimensions.

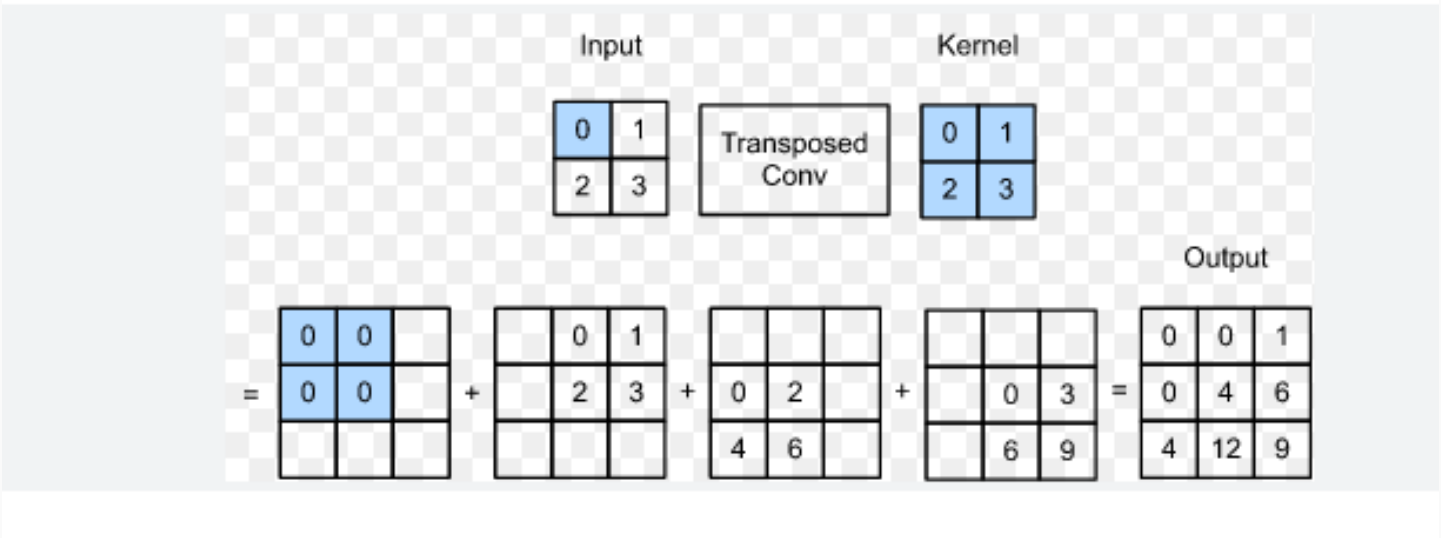


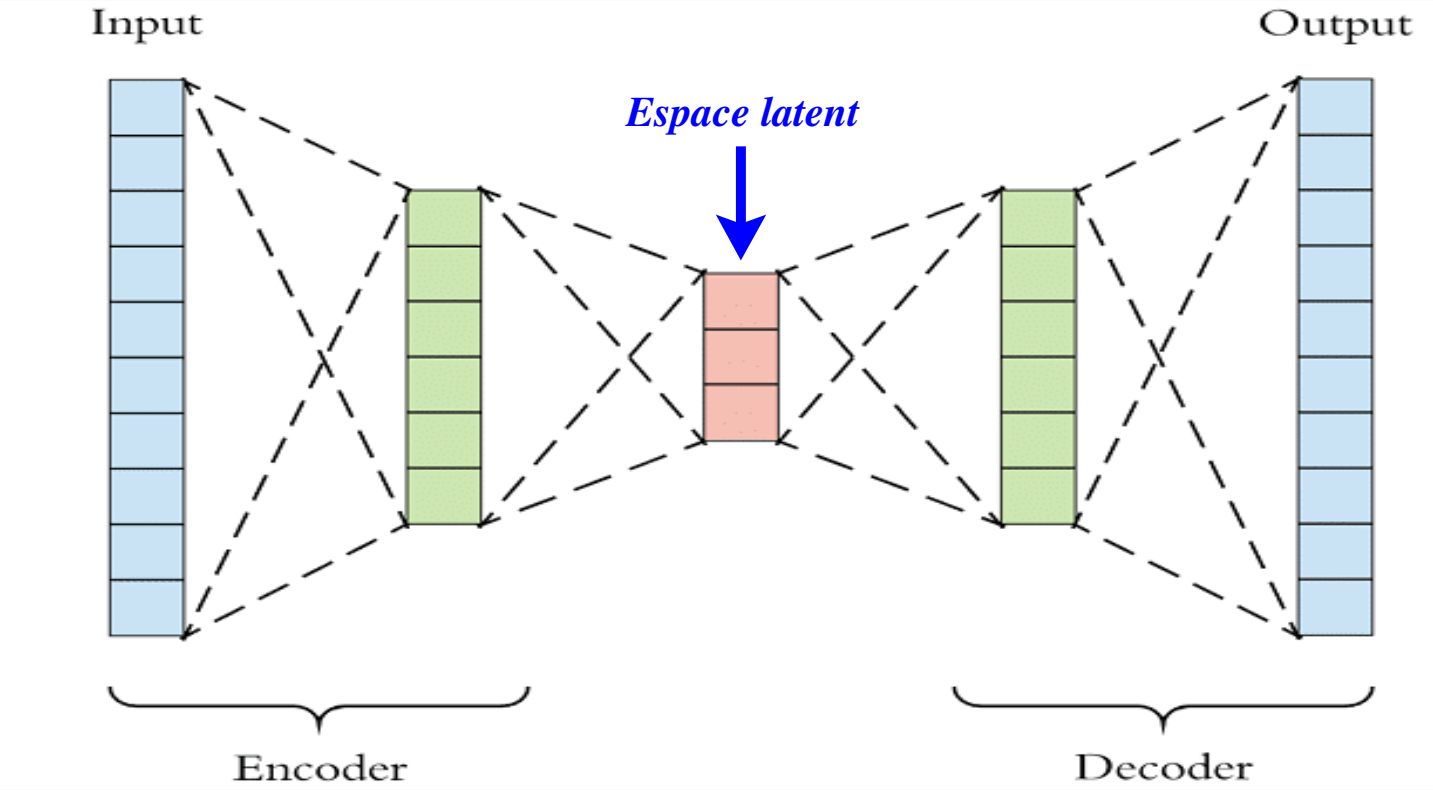
- Image : millions de pixels = millions de valeurs.
- Image = un point dans un espace à quelques millions de dimensions.
- Espace latent : petite dimension (2...10).

Comment compresser l'image ? Une possibilité : convolution avec une foulée (stride) > 1 :



Et en sens inverse (reconstruire une image plus détaillée) :





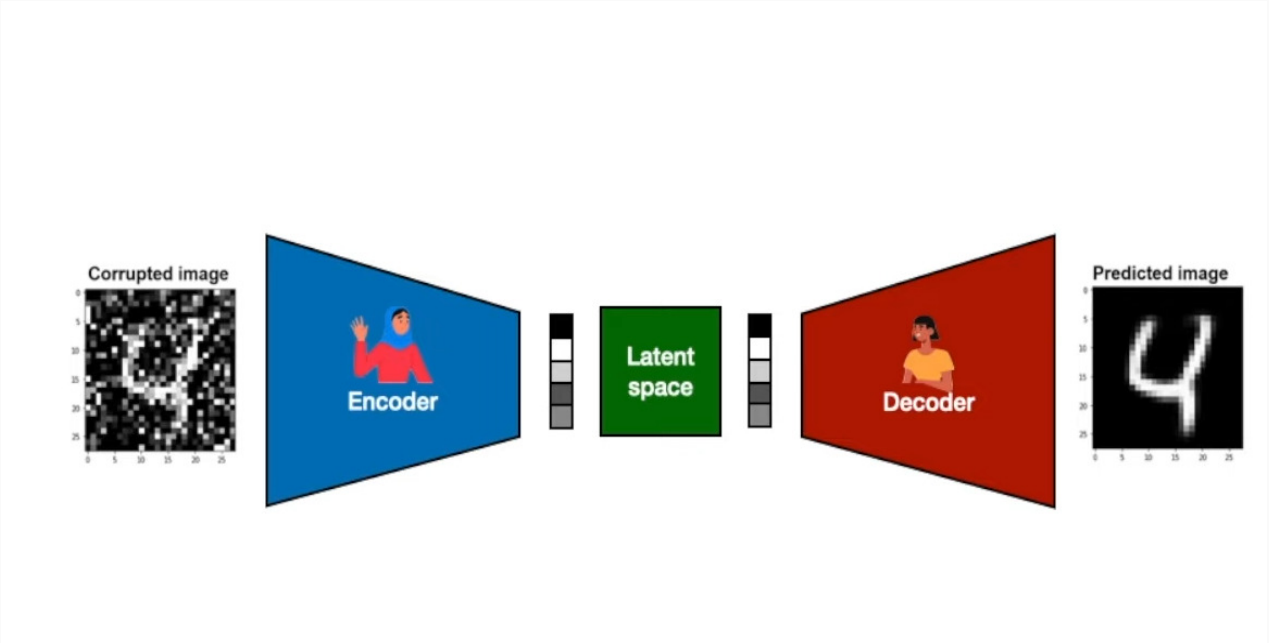
Évidemment, tous les coefficients de tous les noyaux sont des paramètres à apprendre.

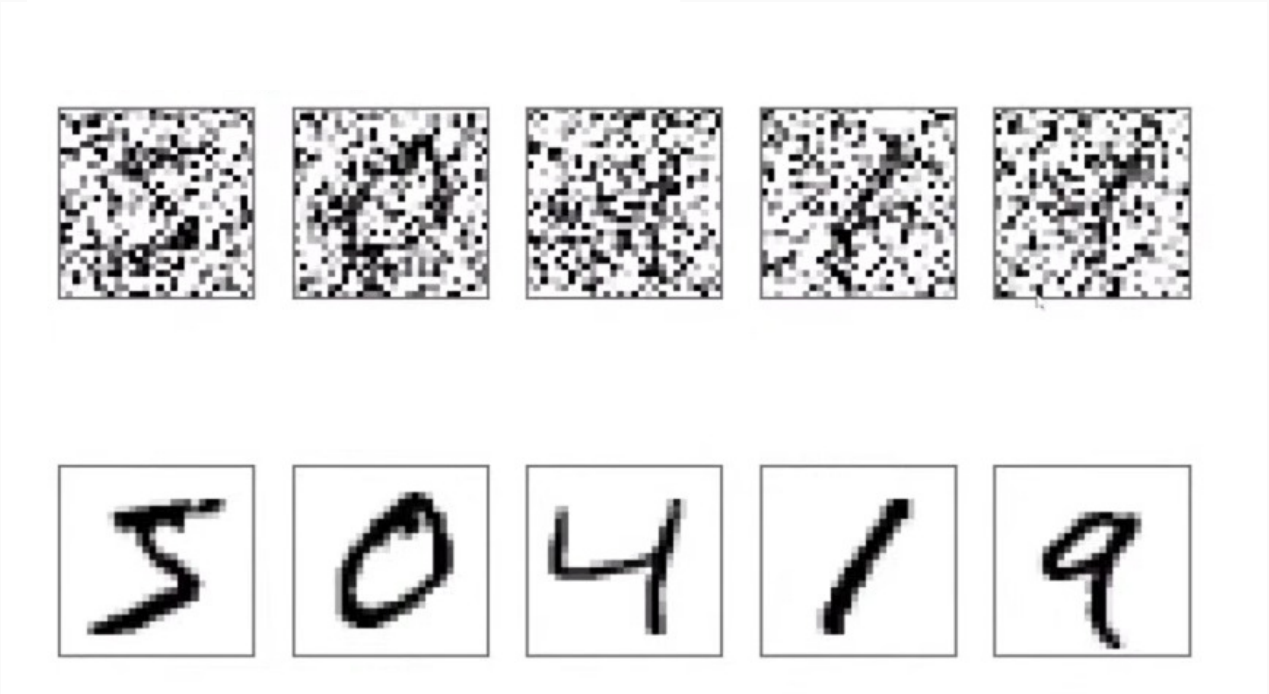
De manière étonnante, ça fonctionne !

De manière étonnante, ça fonctionne !

Espace latent : représentation *parcimonieuse* des images.

Plus fort ! Auto-encodeurs débruiteurs





Apprentissage non supervisé.

Autoencodeurs variationnels

Une image : on tire au hasard la valeur de chacun des pixels.

Quelle est la, probabilité de tomber sur une *vraie* image ?

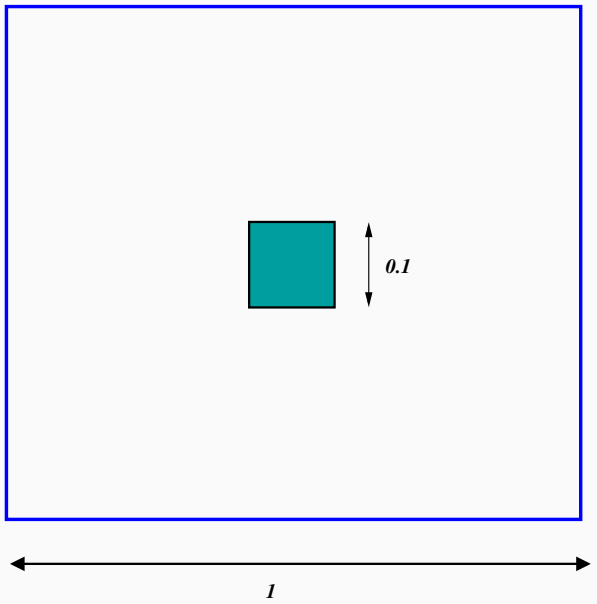
Autoencodeurs variationnels

Une image : on tire au hasard la valeur de chacun des pixels.

Quelle est la, probabilité de tomber sur une *vraie* image ?

Extraordinairent faible !

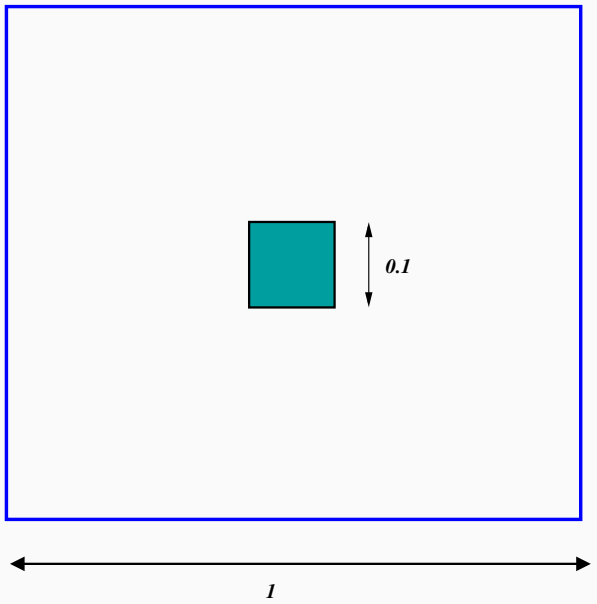
La malédiction de la grande dimension



Espace occupé :

- $0,1^2 = \frac{1}{100}$.

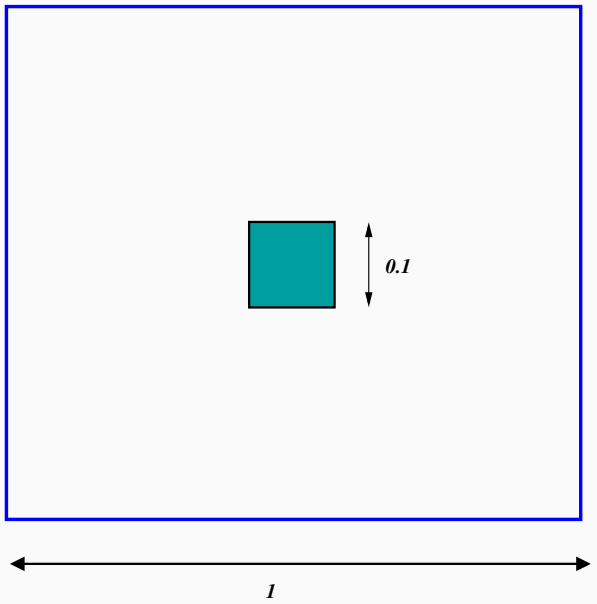
La malédiction de la grande dimension



Espace occupé :

- $0,1^2 = \frac{1}{100}$.
- dans un cube : $0,1^3 = \frac{1}{1000}$.

La malédiction de la grande dimension

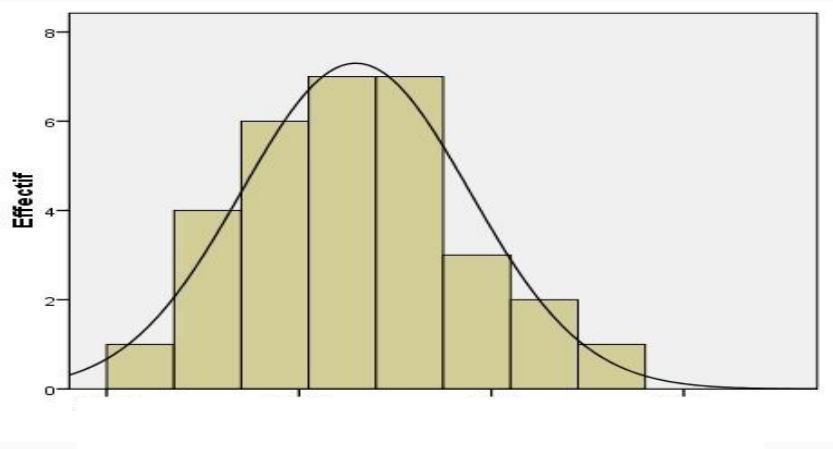


Espace occupé :

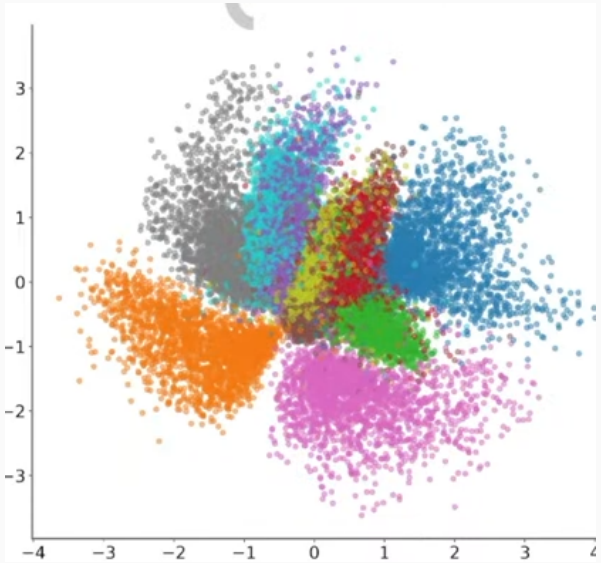
- $0,1^2 = \frac{1}{100}$.
- dans un cube : $0,1^3 = \frac{1}{1000}$.
- dans une image de 10^6 pixels : $0.1^{10^6} = \frac{1}{1 \text{ million de zéros}}$.

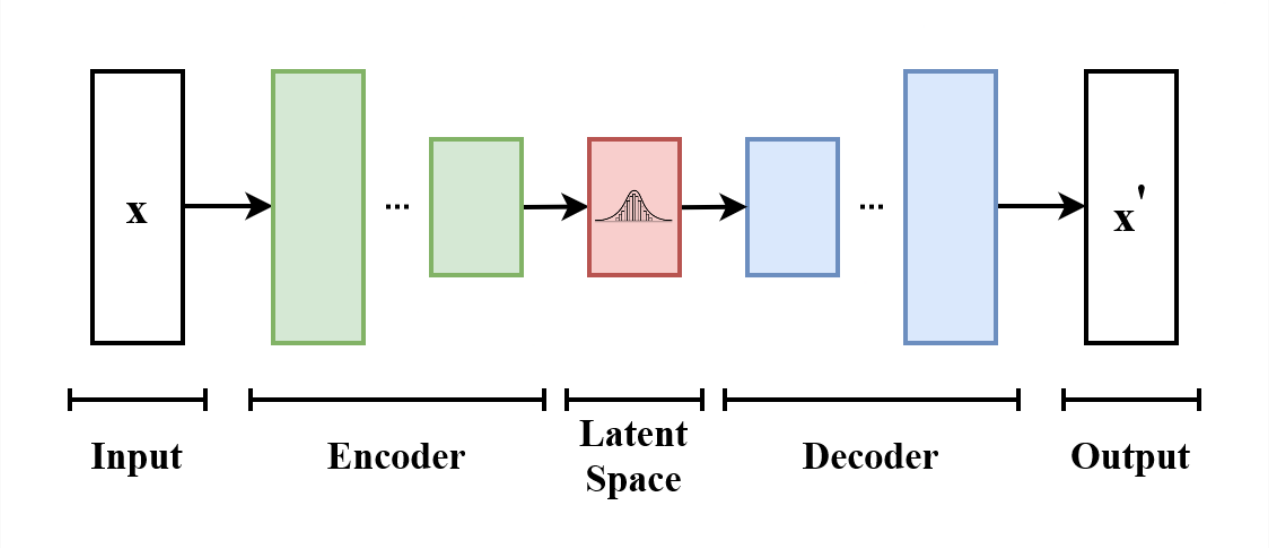
Mais on dispose d'images réelles, un échantillonage des images possibles.

Mais on dispose d'images réelles, un échantillonage des images possibles.



- Entrainer un réseau de neurones pour transformer la loi de probabilité des images en une loi simple, dans un espace de petites dimension (encodeur), les variables latentes.
- Recontruire une image approchée à partir de l'espace latent.




x
...
...
x'
Input
Encoder
Latent Space
Decoder
Output

En parcourant l'espace latent :

IA *générative*.

En parcourant l'espace latent :

IA *générative*.

On peut sûrement mettre des moustaches à la Joconde...

Le langage naturel

Le bas de gamme, genre SMS :

« Salut Anatole, je vais passer ??? »

Le bas de gamme, genre SMS :

« Salut Anatole, je vais passer ??? »

1. Découper la phrase en *jetons* (tokens).

Le bas de gamme, genre SMS :

« Salut Anatole, je vais passer ??? »

1. Découper la phrase en *jetons* (tokens).
2. Associer à chaque *jeton* un ou plusieurs nombres.

Le bas de gamme, genre SMS :

« Salut Anatole, je vais passer ??? »

1. Découper la phrase en *jetons* (tokens).
2. Associer à chaque *jeton* un ou plusieurs nombres.
3. Retour aux méthodes précédentes.

Le bas de gamme, genre SMS :

« Salut Anatole, je vais passer ??? »

1. Découper la phrase en *jetons* (tokens).
2. Associer à chaque *jeton* un ou plusieurs nombres.
3. Retour aux méthodes précédentes.

Ce mécanisme ne regarde qu'en arrière.

Le bas de gamme, genre SMS :

« Salut Anatole, je vais passer ??? »

1. Découper la phrase en *jetons* (tokens).
2. Associer à chaque *jeton* un ou plusieurs nombres.
3. Retour aux méthodes précédentes.

Ce mécanisme ne regarde qu'en arrière.

Voir le prochain exposé !

Calculer

Calcul : ce qui rend l'entraînement possible :
« heureuse coïncidence »

Entraînement :

- énorme quantité de données,
- énorme quantité de calculs.
- Pas besoin d'une grande précision.

Entraînement :

- énorme quantité de données,
- énorme quantité de calculs.
- Pas besoin d'une grande précision.
- Il **faut** paralléliser les calculs.
- Calculs aisément *parallélisables.*

Entraînement :

- énorme quantité de données,
- énorme quantité de calculs.
- Pas besoin d'une grande précision.
- Il **faut** paralléliser les calculs.
- Calculs aisément *parallélisables.*

Heureuse coïncidence : le développement des processeurs graphiques (GPU).

Les deux sortes de parallélisme :

1. MIMD (Multiple Instructions, Multiple Data) :
   - Processeurs à n cœurs,
   - Super calculateurs (réseaux de machines classiques, avec processeurs à n cœurs). Exemple : TaiHu, 16 millions de cœurs.

Les deux sortes de parallélisme :

1. MIMD (Multiple Instructions, Multiple Data) :
   - Processeurs à n cœurs,
   - Super calculateurs (réseaux de machines classiques, avec processeurs à n cœurs). Exemple : TaiHu, 16 millions de cœurs.
2. SIMD (Single Instruction, Multiple Data) :
   - Exemple : additionner ou multiplier 2 tableaux en une seule instruction.

Les deux sortes de parallélisme :

1. MIMD (Multiple Instructions, Multiple Data) :
   - Processeurs à n cœurs,
   - Super calculateurs (réseaux de machines classiques, avec processeurs à n cœurs). Exemple : TaiHu, 16 millions de cœurs.
2. SIMD (Single Instruction, Multiple Data) :
   - Exemple : additionner ou multiplier 2 tableaux en une seule instruction.

GPU.

Graphic Processing Unit.

Les deux sortes de parallélisme :

1. MIMD (Multiple Instructions, Multiple Data) :
   - Processeurs à n cœurs,
   - Super calculateurs (réseaux de machines classiques, avec processeurs à n cœurs). Exemple : TaiHu, 16 millions de cœurs.
2. SIMD (Single Instruction, Multiple Data) :
   - Exemple : additionner ou multiplier 2 tableaux en une seule instruction.

GPU.

Graphic Processing Unit.

Chance ! les calculs sont bien adaptés à ce type de parallélisme.

Performances :

- CPU : 64 Gigaflops par cœur ($= 64\ 10^9$).
- NVIDIA H100 : 134 Teraflops ($= 134\ 10^{12}$) (et même 3000 teraflops dans certains cas) !

Performances :

- CPU : 64 Gigaflops par cœur ($= 64\ 10^9$).
- NVIDIA H100 : 134 Teraflops ($= 134\ 10^{12}$) (et même 3000 teraflops dans certains cas) !

  Coût : 40 000 euros.
  Nvidia peut les vendre par « cabinets » de 4608 GPU.

Logiciels les plus fréquents

- TensorFlow (Google).
- PyTorch (Meta).
- Keras : (F. Cholet) sur-ensemble des précédents.

Tous sont libres (licences Apache 2.0, BSD, MIT)

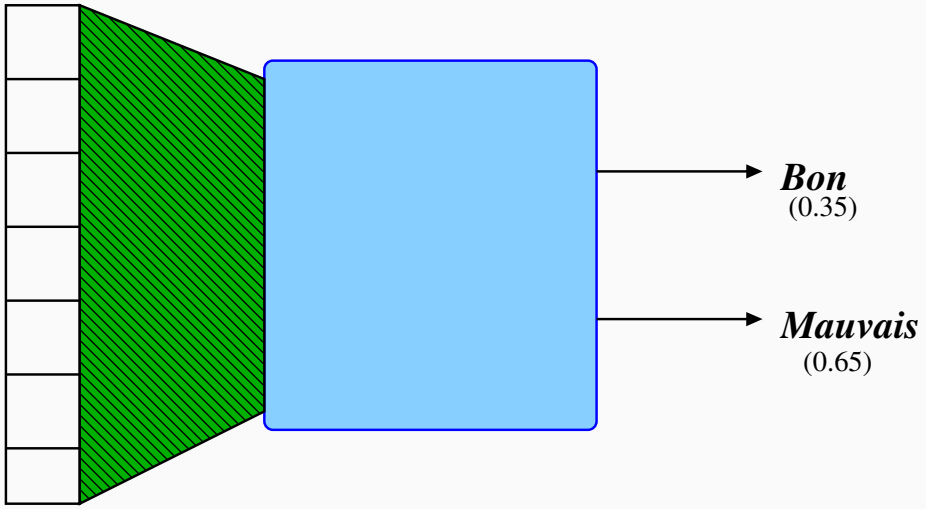
Logiciels les plus fréquents

- TensorFlow (Google).
- PyTorch (Meta).
- Keras : (F. Cholet) sur-ensemble des précédents.

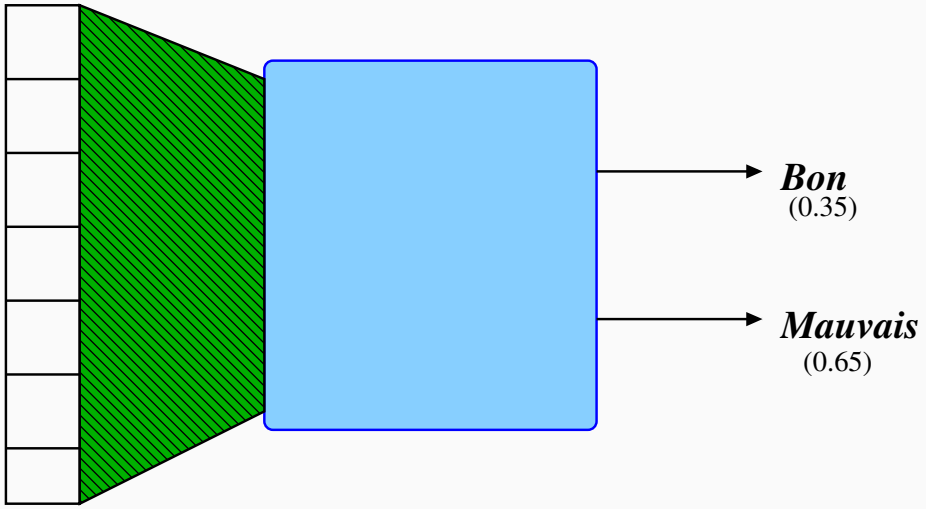
Tous sont libres (licences Apache 2.0, BSD, MIT) cependant CUDA (bibliothèque de Nvidia) est open-source, mais pas libre.

Dépendance pratiquement absolue à Nvidia !

Dangers



Dites bonjour au crédit social !



Dites bonjour au crédit social !

Rappel : à priori, aucun résultat d'un processus de classification n'est sûr à 100 % !

Un exemple donné par Stéphane Mallat (Collège de France) :

système d'aide à la décision pour la libération des prisonniers aux États Unis.

Un exemple donné par Stéphane Mallat (Collège de France) :

système d'aide à la décision pour la libération des prisonniers aux États Unis.

Pose la question : « Pourquoi et comment ça fonctionne ? ».

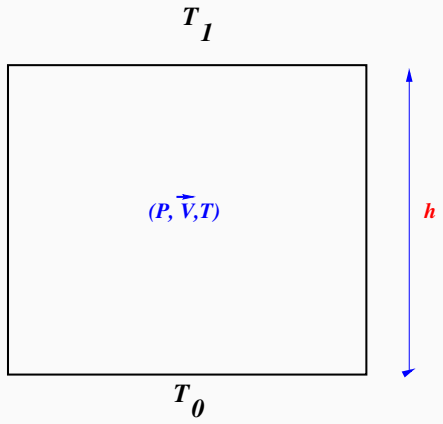
- Confondre corrélation et causalité.

- Confondre corrélation et causalité.
- On ne peut prévoir au mieux que ce qui est contenu dans les données.

Prévoir seulement ce qui est contenu dans les données

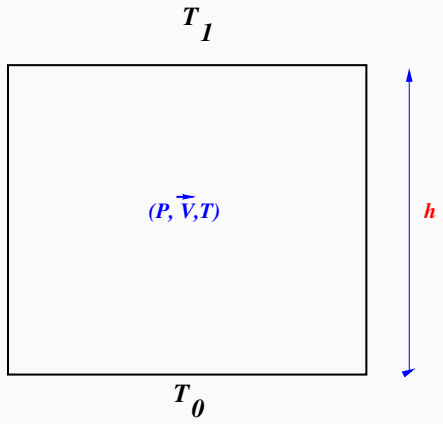
Exemple :

l'instabilité de Rayleigh-Bénard



Prévoir seulement ce qui est contenu dans les données

Exemple :
l'instabilité de Rayleigh-Bénard

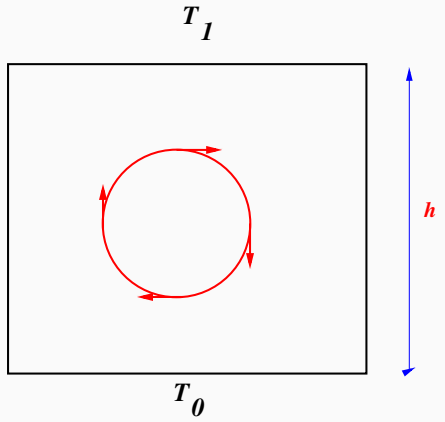


On observe que :

1. la pression P augmente proportionnellement à $T_0 - T_1$,
2. la vitesse $\vec{V}$ reste nulle jusqu'à...

jusqu'à ce que $T_0 - T_1$ dépasse une valeur critique $\mathrm{Ra} \geq 1700$ (environ)

$$\mathrm{Ra} = \frac{\beta g h^3 (T_0 - T_1)}{\nu \kappa}$$



- (bifurcation) :

- (bifurcation) : c'est quelque chose qu'on peut prévoir sans l'avoir observée en connaissant seulement les équations des fluides incompressibles.

- (bifurcation) : c'est quelque chose qu'on peut prévoir sans l'avoir observée en connaissant seulement les équations des fluides incompressibles.
- Comment faire par apprentissage ? Comment deviner la forme du coefficient de Rayleigh ? Conséquences sur une installation industrielle ?

Mais pourquoi est-ce que ça fonctionne ?

Pourquoi est-ce que ça fonctionne (assez correctement) ?

On ne sait pas trop !

Nécessité de comprendre (voir ci-dessus le problème des prisonniers).

Voir par exemple l'exposé de Stéphane Mallat (Collège de France) à la MMI.

Pistes :

- Régularité.

Voir par exemple l'exposé de Stéphane Mallat (Collège de France) à la MMI.

Pistes :

- Régularité.
- Parcimonie.

Voir par exemple l'exposé de Stéphane Mallat (Collège de France) à la MMI.

Pistes :

- Régularité.
- Parcimonie.
- Liens avec différentes branches des maths et de la physique (invariance).
- ...

Tout ou presque reste à faire.

Intelligence, vraiment ?

N’oublions pas qu’on utilise des ordinateurs...

N'oublions pas qu'on utilise des ordinateurs... qui exécutent des programmes.

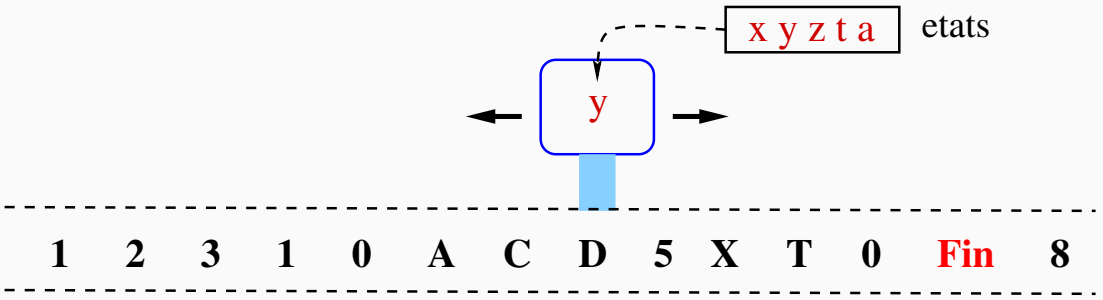
N'oublions pas qu'on utilise des ordinateurs... qui exécutent des programmes. Les programmes implantent des algorithmes.

N'oublions pas qu'on utilise des ordinateurs... qui exécutent des programmes. Les programmes implantent des algorithmes.

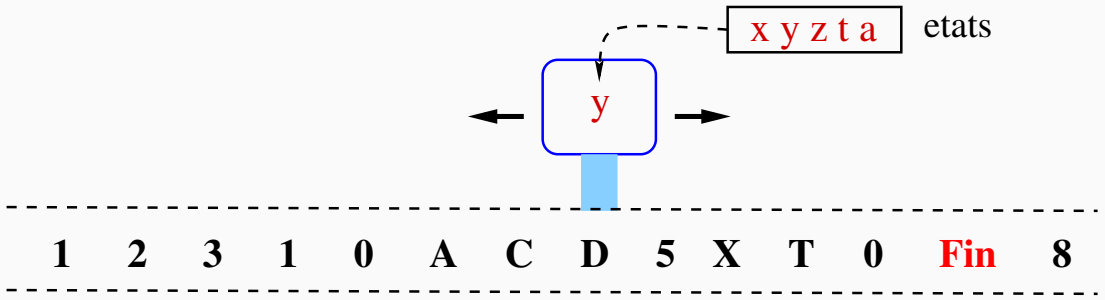
Que peut-on (ou ne peut-on pas) faire dans ce cadre ?

Machines de Turing (Alan Turing, 1936)

Machines de Turing (Alan Turing, 1936)



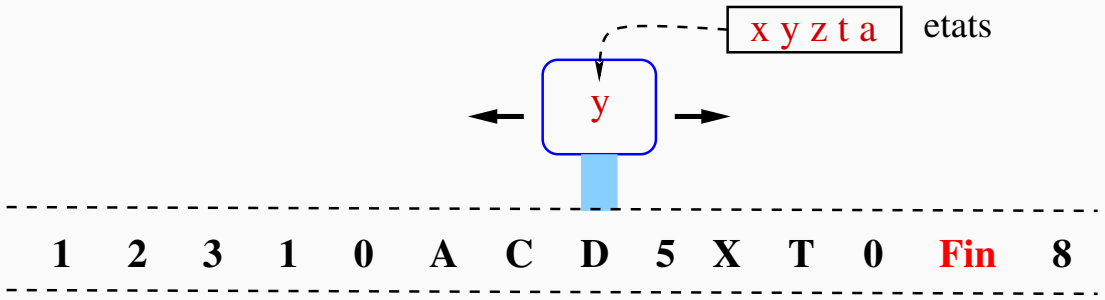
Machines de Turing (Alan Turing, 1936)



Un pas de la machine :

- lecture d'un symbole sur la bande.

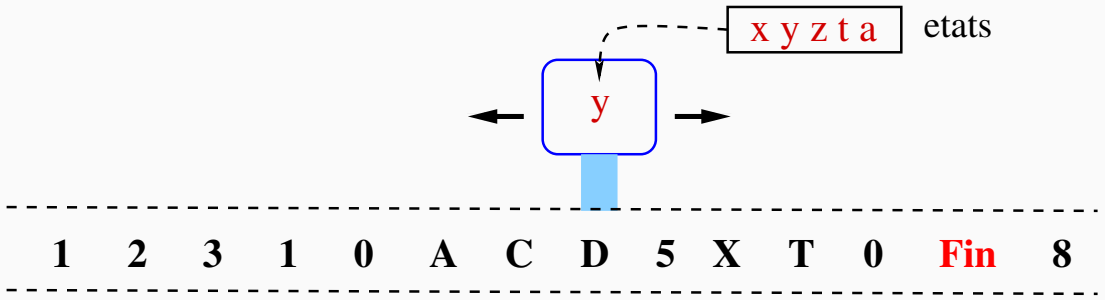
Machines de Turing (Alan Turing, 1936)



Un pas de la machine :

- lecture d'un symbole sur la bande.
- table de transition : en fonction de l'état de la tête et du symbole lu :

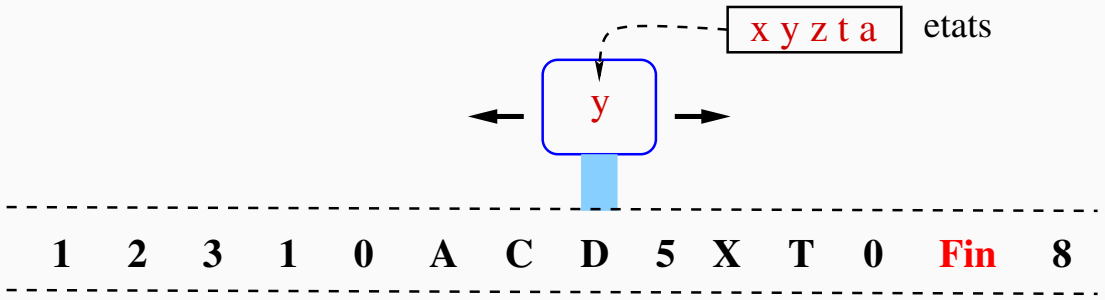
Machines de Turing (Alan Turing, 1936)



Un pas de la machine :

- lecture d'un symbole sur la bande.
- table de transition : en fonction de l'état de la tête et du symbole lu :
  - écriture sur la bande,
  - déplacement éventuel à gauche ou à droite,
  - modification de l'état de la tête.
  - arrêt si le symbole lu est Fin.

Machines de Turing (Alan Turing, 1936)



Un pas de la machine :

- lecture d'un symbole sur la bande.
- table de transition : en fonction de l'état de la tête et du symbole lu :
  - écriture sur la bande,
  - déplacement éventuel à gauche ou à droite,
  - modification de l'état de la tête.
  - arrêt si le symbole lu est Fin.

  Exemple de transition : { y, **D**} => { t, **X**, $\leftarrow$}

- Les machines de Turing sont « universelles » (base de la *calculabilité*).

- Les machines de Turing sont « universelles » (base de la *calculabilité*).
- Les ordinateurs, munis de langages de programmation sont « Turing équivalents ».

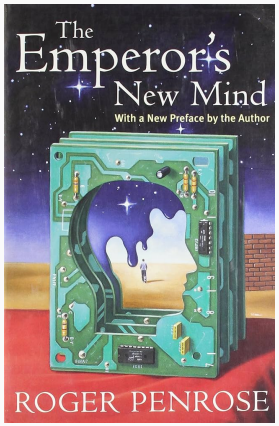
- Les machines de Turing sont « universelles » (base de la *calculabilité*).
- Les ordinateurs, munis de langages de programmation sont « Turing équivalents ».
- Donc, ce que peux faire l'IA (l'apprentissage profond), c'est au plus ce que peut faire une machine de Turing.

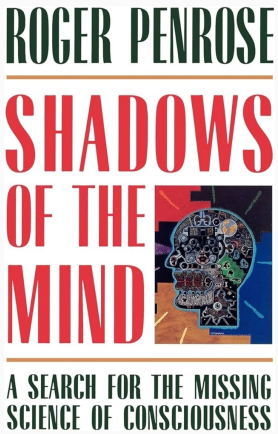
- Les machines de Turing sont « universelles » (base de la *calculabilité*).
- Les ordinateurs, munis de langages de programmation sont « Turing équivalents ».
- Donc, ce que peux faire l'IA (l'apprentissage profond), c'est au plus ce que peut faire une machine de Turing.

- Le cerveau est-il « Turing équivalent » ?
- La pensée, la conscience, l'intelligence ne sont-elles que des algorithmes ?

Les idées de Roger Penrose

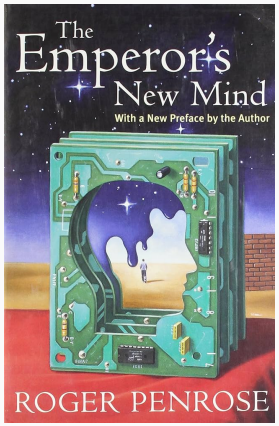
Les idées de Roger Penrose

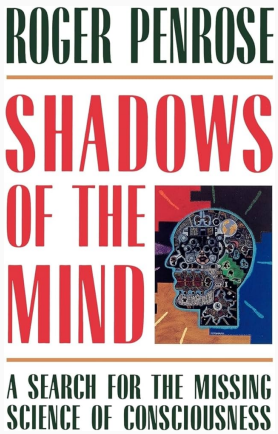




- Arguments basés sur le *théorème de l'arrêt* (Turing) et les résultats de Gödel.

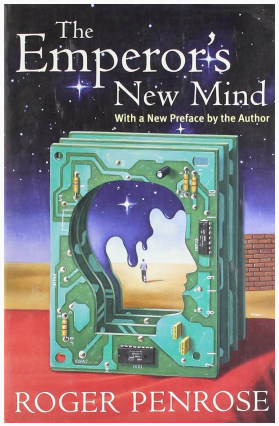
Les idées de Roger Penrose

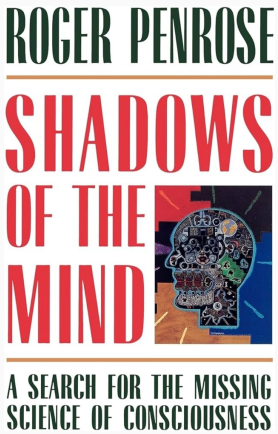




- Arguments basés sur le *théorème de l'arrêt* (Turing) et les résultats de Gödel.
- Si conscience = algorithmes, quel peut avoir été le rôle de l'évolution ?

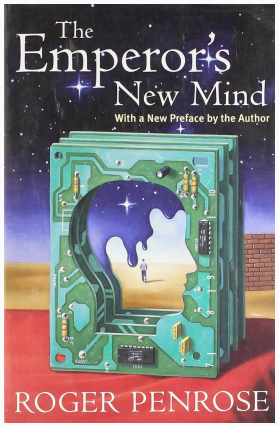
Les idées de Roger Penrose

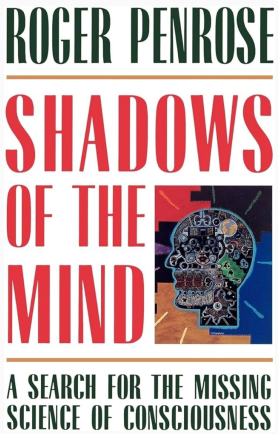




- Arguments basés sur le *théorème de l'arrêt* (Turing) et les résultats de Gödel.
- Si conscience = algorithmes, quel peut avoir été le rôle de l'évolution ?
- Roger Penrose s'intéresse aux *éclairs de compréhension.*

Les idées de Roger Penrose





- Arguments basés sur le *théorème de l'arrêt* (Turing) et les résultats de Gödel.
- Si conscience = algorithmes, quel peut avoir été le rôle de l'évolution ?
- Roger Penrose s'intéresse aux *éclairs de compréhension.*
- Existence de phénomènes quantiques dans le cerveau ? *(=> Quantum mind)*.

« Quantum mind »

Penrose, Hameroff et pas mal d'autres (ainsi qu'E. Wigner auparavant).

« Quantum mind »

Penrose, Hameroff et pas mal d'autres (ainsi qu'E. Wigner auparavant).

- Existence de phénomènes quantiques *non calculables*.
- Candidats : micro tubules.

Beaucoup de réfutations, mais une expérience plutôt positive en avril 2022.

Noam Chomsky

The New York Times mars 2023.

- « Il est à la fois comique et tragique que tant d'argent et d'attention se concentrent sur si peu – quelque chose de si trivial comparé à l'esprit humain qui, par le biais du langage, pour reprendre les mots de Wilhelm von Humboldt, peut faire « un usage infini de moyens finis », élaborant des idées et des théories ayant une portée universelle. »

- « J'ai fini par demander à ChatGPT s'il connaissait l'idée de banalité du mal et s'il se sentait concerné. Voilà sa réponse :

- « J'ai fini par demander à ChatGPT s'il connaissait l'idée de banalité du mal et s'il se sentait concerné. Voilà sa réponse : *«Il est vrai que je suis un outil créé par des humains, et par conséquent, je peux refléter les limites et les biais de mes créateurs et des données sur lesquelles je suis entraîné. »*

- « J'ai fini par demander à ChatGPT s'il connaissait l'idée de banalité du mal et s'il se sentait concerné. Voilà sa réponse : *«Il est vrai que je suis un outil créé par des humains, et par conséquent, je peux refléter les limites et les biais de mes créateurs et des données sur lesquelles je suis entraîné. »* Une intelligence servile et sans pensée, c'est en effet une bonne définition de ce que Hannah Arendt cherchait à saisir à travers l'idée de la **banalité du mal.** »

Pour finir...

Références, lectures plus ou moins recommandables

- Cours de Stéphane Mallat au Collège de France (en ligne) *Science des données.*
- Conférence de Stéphane Mallat à la Maison des Maths et de l'Informatique (Gerland) `https://www.youtube.com/watch?v=UJwBx7eqgBw`
- Deep Learning for dummies.

- Cours de Stéphane Mallat au Collège de France (en ligne) *Science des données.*
- Conférence de Stéphane Mallat à la Maison des Maths et de l'Informatique (Gerland) `https://www.youtube.com/watch?v=UJwBx7eqgBw`
- Deep Learning for dummies.
- FIDLE : Cours en ligne, libre, gratuit (UJF Grenoble) `https://gricad-gitlab.univ-grenoble-alpes.fr/talks/fidle/-/wikis/home`
- F. Chollet : *L'apprentissage profond avec Python*.

- Cours de Stéphane Mallat au Collège de France (en ligne) *Science des données.*
- Conférence de Stéphane Mallat à la Maison des Maths et de l'Informatique (Gerland)
  `https://www.youtube.com/watch?v=UJwBx7eqgBw`
- Deep Learning for dummies.
- FIDLE : Cours en ligne, libre, gratuit (UJF Grenoble)
  `https://gricad-gitlab.univ-grenoble-alpes.fr/talks/fidle/-/wikis/home`
- F. Chollet : *L'apprentissage profond avec Python*.
- G. Strang : *Linear Algebra and Learning from Data* (Welessley-Cambridge Press).
  Cours du MIT par le *génial* pédagogue Gilbert Strang.

- Cours de Stéphane Mallat au Collège de France (en ligne)
  Science des données.
- Conférence de Stéphane Mallat à la Maison des Maths et de l'Informatique (Gerland)
  `https://www.youtube.com/watch?v=UJwBx7eqgBw`
- Deep Learning for dummies.
- FIDLE : Cours en ligne, libre, gratuit (UJF Grenoble)
  `https://gricad-gitlab.univ-grenoble-alpes.fr/talks/fidle/-/wikis/home`
- F. Chollet : *L'apprentissage profond avec Python*.
- G. Strang : *Linear Algebra and Learning from Data* (Welessley-Cambridge Press).
  Cours du MIT par le *génial* pédagogue Gilbert Strang.
- Les livres de R. Penrose.

Thierry Dumont :

mail : thierry@thierry-dumont.fr

Diapos disponibles sur : https://thierry-dumont.fr/Exposes