

Apprentissage Profond - Partie 1

Redouane Lguensat

Master Data Engineering de l'EHTP

Avril 2020

Table des matières

1 Introduction

2 Réseaux de Neurones Artificiels

3 Apprentissage profond

Pourquoi un tel engouement?



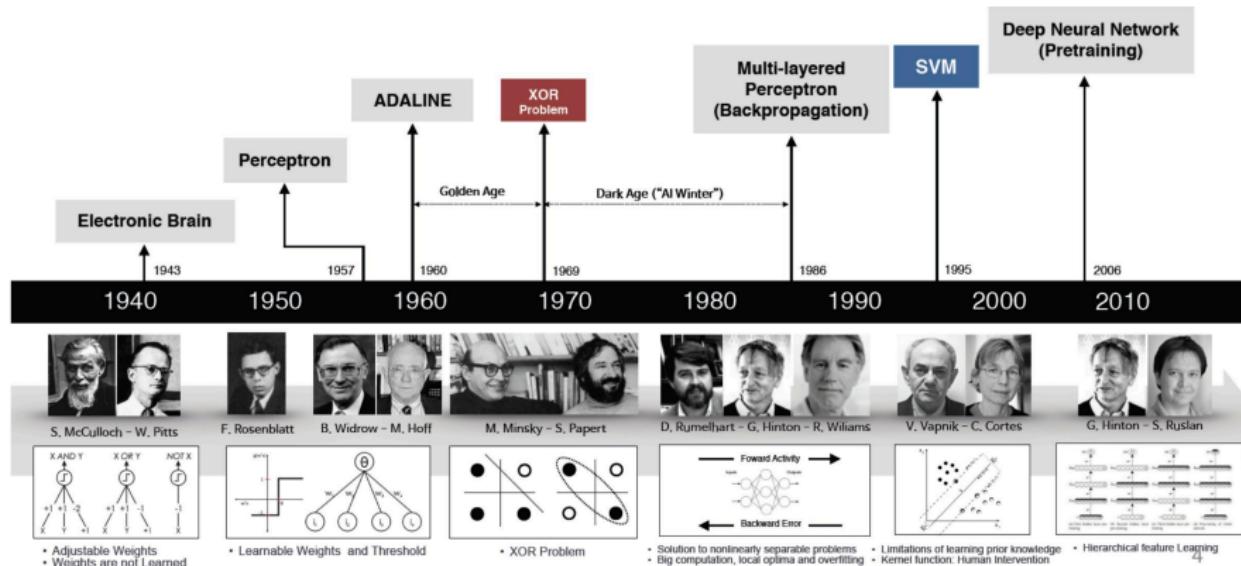
Source: <https://www.slideshare.net/Franrgo/gtam-vis-ban-the-new-cold-war-in-technological>

Impact socio-économique fort



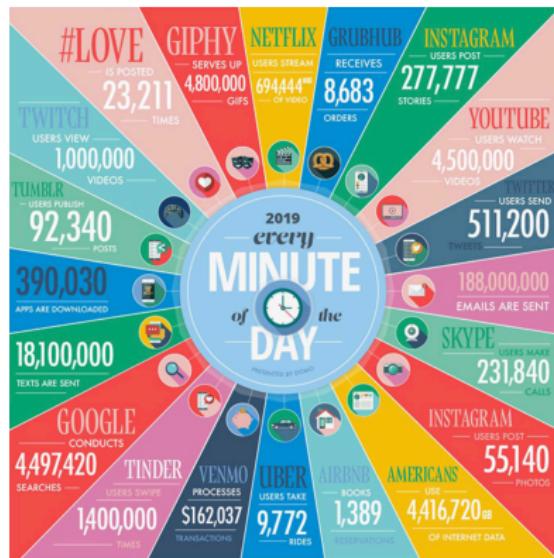
Découvertes majeures
en recherche scientifique

Un peu d'Histoire



Source: <https://www.slideshare.net/devview/251-implementing-deep-learning-using-cu-dnn>

Les catalyseurs de la révolution



<https://www.domo.com/learn/data-never-sleeps-7>

Volume de données grandissant



Avancées technologiques en capacité de calcul notamment sur cartes graphiques (GPU)



المدرسة الحسنية للأشغال العمومية
ECOLE HASSANIA DES TRAVAUX PUBLICS

Quelques applications de l'apprentissage profond

- Reconnaissance d'objets dans une image
- Traduction automatique
- Détection de contenu indésirable (ex: spam)
- Voiture autonome
- Assistance médicale
- Systèmes de recommandation
- Détection d'anomalies (ex: fraudes)
- Chatbots (agents conversationnels)
- Création d'oeuvre artistique
- Télédétection spatiale
- Jeux de table ou vidéo
- Transcription de la voix
- Génération de musique
- Recherche de nouveaux médicaments
- Détection de nouvelles galaxies
-

La liste est en pleine expansion!

Table des matières

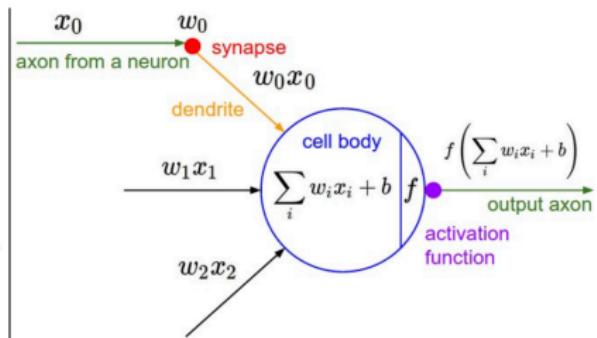
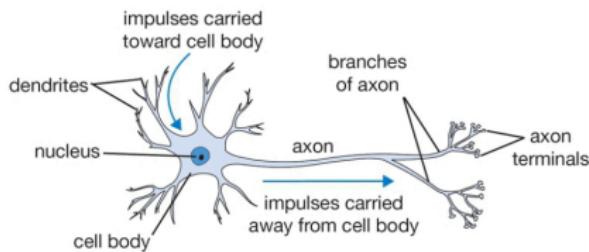
1 Introduction

2 Réseaux de Neurones Artificiels

3 Apprentissage profond

Inspiration de la biologie

Neuron biologique vs neuron "artificial"



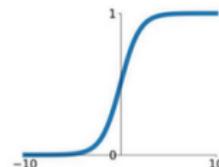
<https://cs231n.github.io/neural-networks-1/>

Exemples classiques de fonctions d'activation

Servent à introduire de la nonlinéarité dans le modèle

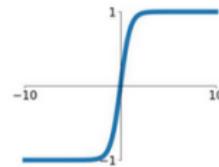
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



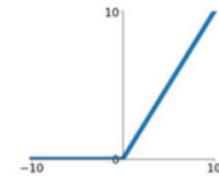
tanh

$$\tanh(x)$$



ReLU

$$\max(0, x)$$



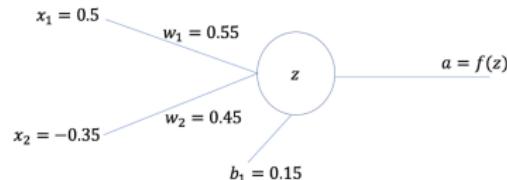
D'autres exemples de fonctions d'activation https://fr.wikipedia.org/wiki/Fonction_d%27activation

Un neurone artificiel en quelques lignes de code

```
from keras.models import Sequential
model = Sequential()
model.add(Dense(1, input_dim=5, activation='sigmoid'))
```

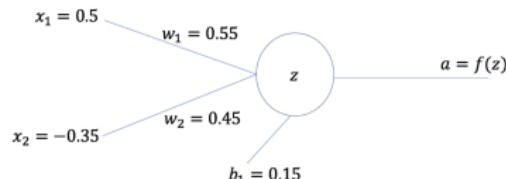
Exemple rapide

Exercice: supposons que f est une sigmoïde, calculez a la sortie du neurone suivant:



Exemple rapide

Exercice: supposons que f est une sigmoïde, calculez a la sortie du neuron suivant:



CO EHTP_MsDataEng.ipynb ☆

Fichier Modifier Affichage Insérer Exécution Outils Aide

+ Code + Texte

```
<> import numpy as np
     import math
```

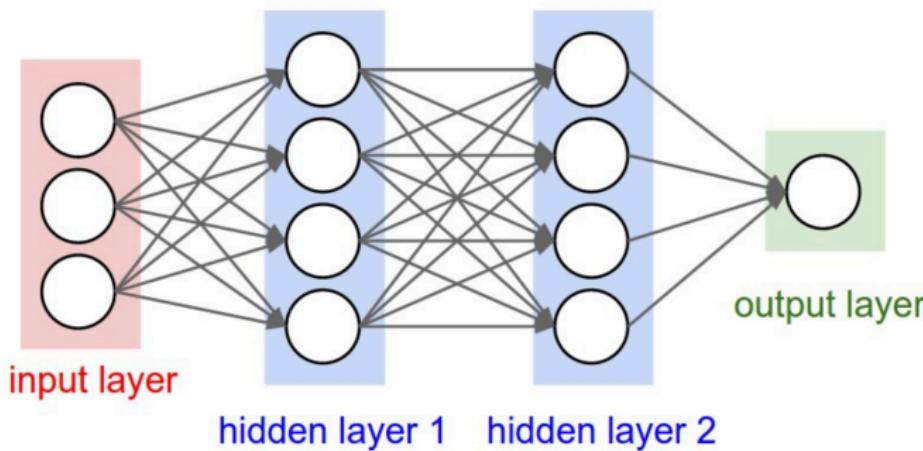
[2] def sigmoid(x):
 return 1 / (1 + math.exp(-x))

[3] sigmoid(0.55*0.5+(-0.35)*0.45+0.15)

0.5664790559676278

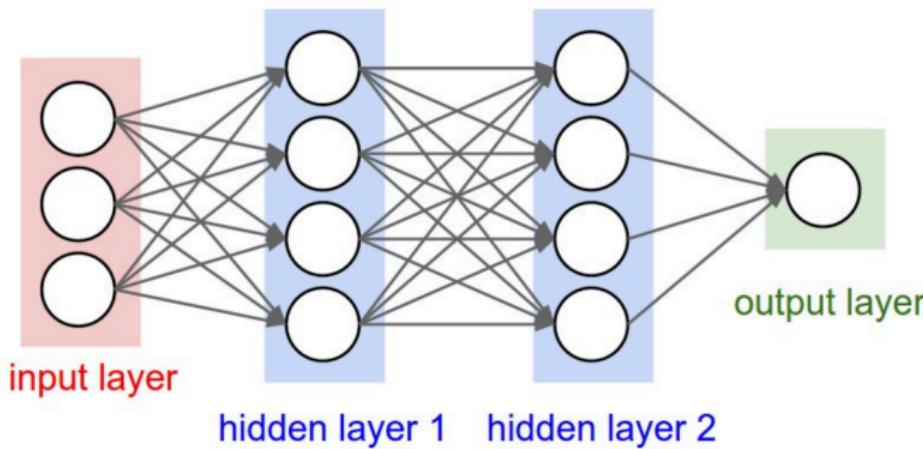
Réseaux de Neurones Artificiels

Les neurones artificiels peuvent être interconnectés et construire des **réseaux de neurones artificiels**. Ces derniers peuvent aussi être composés en couches (*layers*) de neurones. Un exemple très utilisé est le **perceptron multicouche** (*Multilayer perceptron (MLP)*).



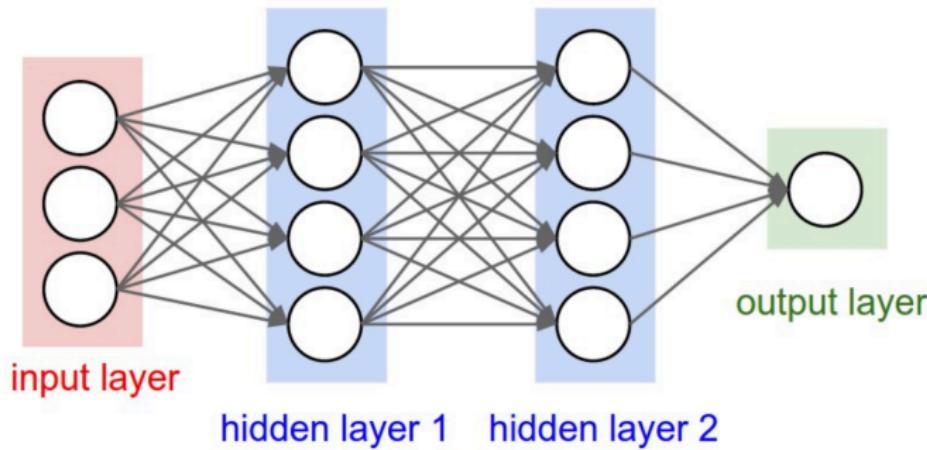
Réseaux de Neurones Artificiels

Le MLP est un réseau à propagation directe (*feed forward*), en général nous supposons qu'il est complètement connecté (*fully connected*).



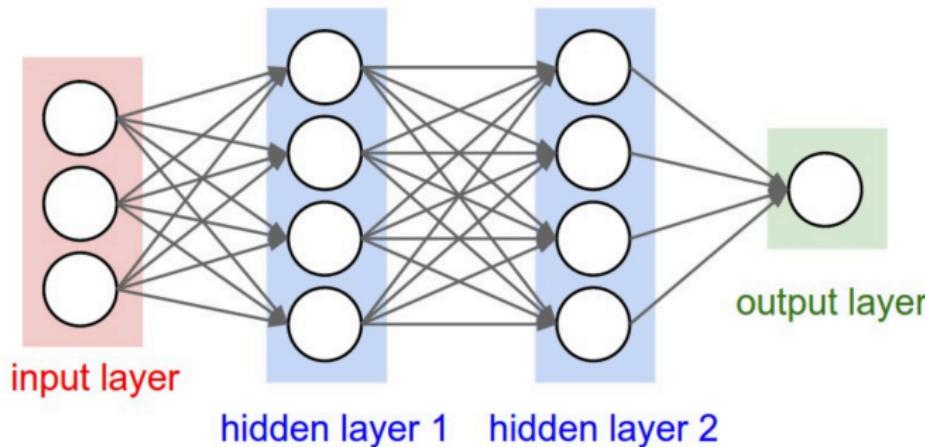
Réseaux de Neurones Artificiels

Traduire un MLP en code Keras



Réseaux de Neurones Artificiels

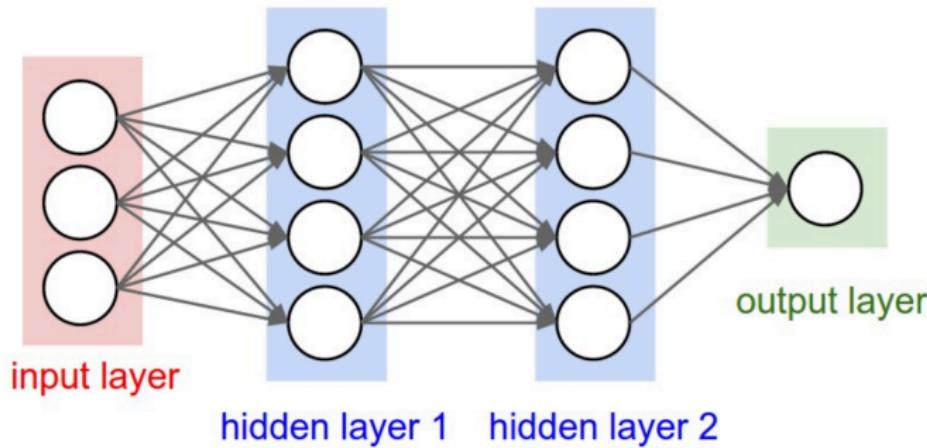
Traduire un MLP en code Keras



```
model = Sequential()  
model.add(Dense(4, activation='relu', input_shape=(3,)))  
model.add(Dense(4, activation='relu'))  
model.add(Dense(1))
```

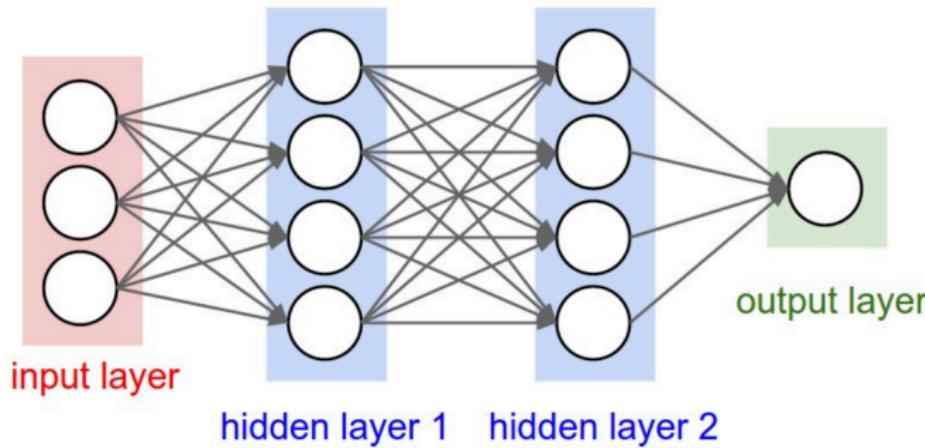
Entraîner des Réseaux de Neurones

Entraîner un réseau de neurones pour une tâche particulière revient à trouver les bons paramètres de ce réseaux: les poids w et les biais b de chaque neurone.



Entraîner des Réseaux de Neurones

Entraîner un réseau de neurones pour une tâche particulière revient à trouver les bons paramètres de ce réseaux: les poids w et les biais b de chaque neurone.



Exercice: le réseau illustré comporte combien de paramètres?

Rétropropagation du gradient (Backprop)

L'algorithme le plus populaire pour faciliter l'entraînement des réseaux de neurones est celui de la rétropropagation du gradient. L'idée est de définir une fonction de coût \mathcal{L} (*loss function*) puis mettre à jour les paramètres en utilisant un algorithme d'optimization tel que la descente de gradient (*Gradient Descent*).

Entraîner un réseaux de neurones revient à:

- Au début, initialiser les paramètres du réseau
- Passer un (ou un batch) de données d'apprentissage par le réseau pour obtenir les valeurs de sortie.
- Calculer l'erreur (la fonction coût) commise par le réseau
- Utiliser le backprop pour calculer les dérivées de la fonction de coût par rapport à tous les paramètres du réseau.
- Mettre à jour tous les paramètres du réseau
- Passer un autre batch de données et refaire les étapes jusqu'à convergence.

Backprop avec SGD

Pour pouvoir utiliser un grand nombre de données d'apprentissage, l'algorithme d'optimization le plus utilisé avec le backprop est la **descente de gradient stochastique** (SGD)

En prenant un batch de n données d'entrées/sorties $x^{(i:i+n)}; y^{(i:i+n)}$ et un taux d'apprentissage η (*learning rate*) , la descente de gradient se fait sur les poids et biais:

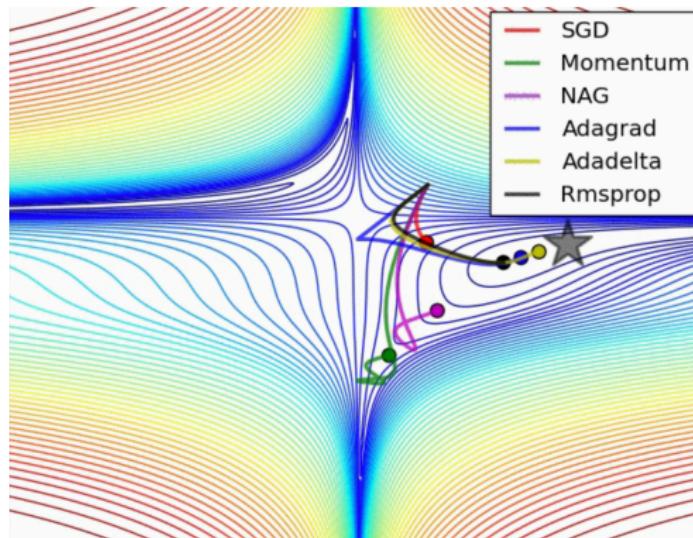
$$w = w - \eta \cdot \nabla_w \mathcal{L} \left(w; x^{(i:i+n)}; y^{(i:i+n)} \right)$$

$$b = b - \eta \cdot \nabla_b \mathcal{L} \left(b; x^{(i:i+n)}; y^{(i:i+n)} \right)$$

Plusieurs variantes du SGD sont utilisés en pratique: ADAM, Adadelta, RMSprop, etc.

Comparaison des variantes du SGD

Excellent travail de review par Sébastien Ruder



Cliquez ici

Backprop avec SGD en pratique avec Keras

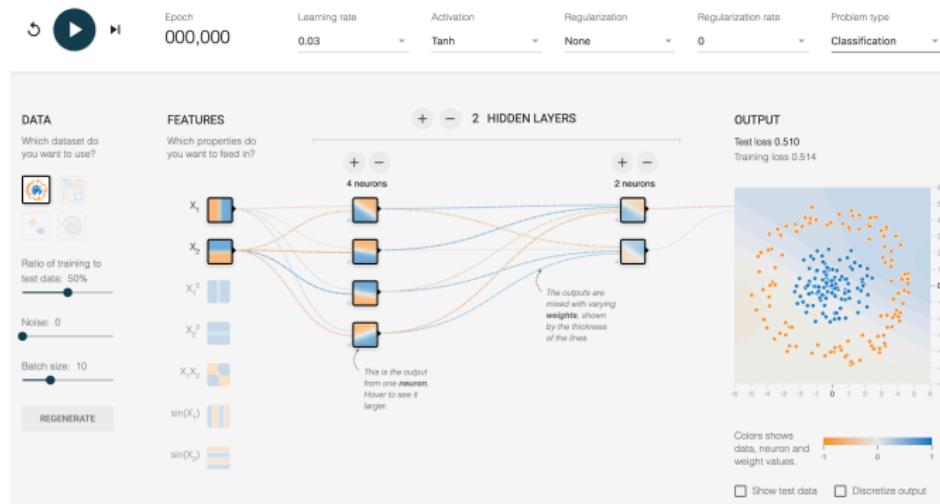
Heureusement vous n'avez pas à tout coder! quelques ligne suffisent

```
model = Sequential()
model.add(Dense(4, activation='relu', input_shape=(3,)))
model.add(Dense(4, activation='relu'))
model.add(Dense(1))

model.compile(loss='categorical_crossentropy',
              optimizer='adam')
model.fit(X_train, y_train, epochs=20, batch_size=128)
```

Outil pédagogique intéressant

Terrain de jeu sur navigateur par Tensorflow (Google)



Cliquez ici

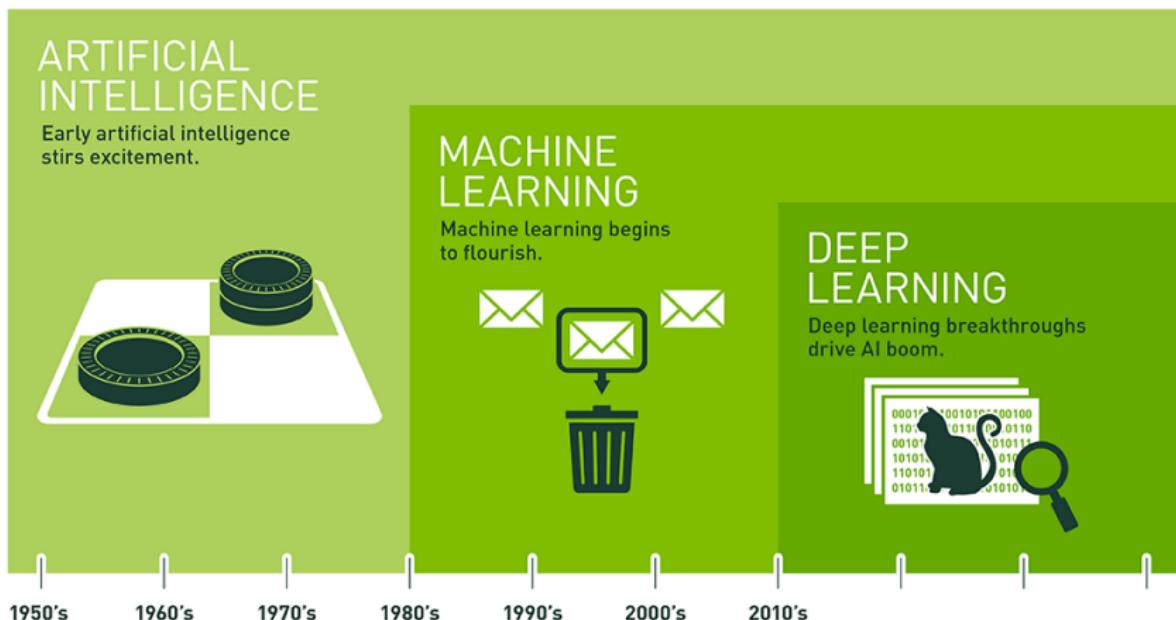
Table des matières

1 Introduction

2 Réseaux de Neurones Artificiels

3 Apprentissage profond

Un sous domaine du Machine Learning



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Apprentissage profond

Slide tiré d'une présentation de Prof. Geoffrey Hinton (un des héros principaux de la renaissance des réseaux de neurones)

What was actually wrong with backpropagation in 1986?

- We all drew the wrong conclusions about why it failed.
The real reasons were:
 1. Our labeled datasets were thousands of times too small.
 2. Our computers were millions of times too slow.
 3. We initialized the weights in a stupid way.
 4. We used the wrong type of non-linearity.

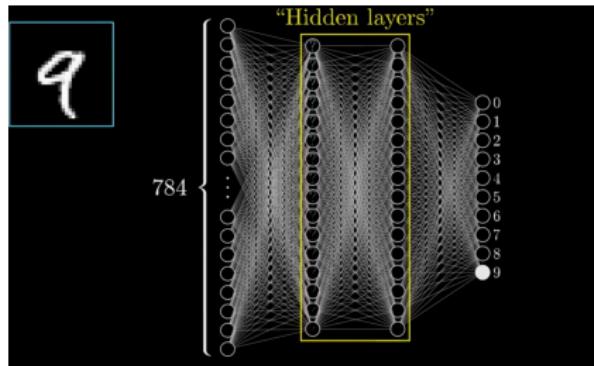
Apprentissage profond

- C'est l'idée de l'utilisation de capacités de calculs avancées et de données massives (Big Data) pour entraîner des réseaux de neurones profond avec une multitude de couches.
- "At which problem depth does Shallow Learning end, and Deep Learning begin? Discussions with DL experts have not yet yielded a conclusive response to this question." - Jürgen Schmidhuber

MNIST: Le "Hello world" de l'apprentissage profond

Dataset d'images en noir et blanc de chiffres manuscrits qui regroupe 60000 images d'apprentissage et 10000 images de test, 28 pixels de côté. Historiquement utilisé pour comparer les algorithmes de machine learning sur le problème de reconnaissance de chiffres manuscrits (utilisé par exemple pour les codes postaux).

Maintenant considéré comme problème "résolu" mais toujours utile pédagogiquement.



3Blue1Brown

MNIST: Le "Hello world" de l'apprentissage profond

Tutoriel sur Colaboratory

Prochain cours: Partie 2

L'apprentissage profond c'est surtout aussi le design et l'entraînement d'architectures adaptés à l'application en vue (classification, régression, etc.) et au type de données (signal, image, vidéo, etc..).
Nous verrons tout cela au prochain cours.

Des questions?