Deep Learning : Je t'aime moi non plus…

Jean-Luc Parouty
Laboratoire SIMaP
1130, rue de la piscine
38 402 Saint-Martin d'Hères

Résumé

Reproduire ou simuler « l'intelligence » est un objectif qui nous anime depuis longtemps et dont les premiers travaux scientifiques sont antérieurs aux ordinateurs.

Longuement occultées, les techniques d'apprentissage automatique (machine learning) et plus particulièrement l'apprentissage profond (deep learning), basées sur les réseaux de neurones, ont fait des progrès exceptionnels ces dernières années.

En faisant le lien entre l'expérimental et la modélisation et en ouvrant de nouvelles voies dans la méthode scientifique, ces technologies ont un potentiel considérable dans la plupart de nos domaines scientifiques mais également dans nos sociétés.

L'objectif de cet article est d'apporter des éléments de compréhension simples et illustrés de ce qu'est le « deep learning », ses principes, usages et enjeux.

Pour cela, je vous propose un parcours en quatre étapes :

- *Nous explorerons **l'épopée tumultueuse** et incertaine de l'intelligence artificielle, avec ses royaumes et ses hivers,*
- *les neurones ayant vaincu, nous parcourrons les **différentes architectures** et usages, en nous appuyant sur des exemples simples et librement accessibles,*
- *nous aborderons les différentes **solutions et offres de services** académiques,*
- *nous terminerons avec quelques **interrogations** concernant l'usage de ces outils dans nos sociétés.*

Mots-clefs

Intelligence Artificielle, Apprentissage machine, Apprentissage profond, réseau de neurones. Artificial intelligence, Machine learning, Deep learning, Neural network

Notebooks

Des notebooks (Jupyter Lab) illustrant cet article sont librement accessibles sur :
Gitlab : https://gricad-gitlab.univ-grenoble-alpes.fr/talks/deeplearning
Binder : https://bit.ly/2m1qfuO

1. [l'intelligence], c'est quoi ?

Parler d'intelligence « artificielle » sans définir ce qu'est l'intelligence « naturelle » est une excellente opportunité d'ouvrir la boite à polémiques.

Des générations de philosophes, scientifiques et autres penseurs, nous ont longuement octroyé, à nous autres humains, le monopole de la pensée intelligente. Las, au fil du temps et de manière inexorable, notre présupposée exception s'est révélée être tout à fait banale et commune dans le monde animal. Même les abeilles savent compter et faire des soustractions [1]…

Pour autant, qualifier d'intelligent une abeille ou un poulpe est une chose, mais une machine ?

Si l'on considère la définition retenue dans Wikipedia[1], la réponse est clairement positive.

> « (...) elle [l'intelligence] peut être décrite comme la capacité de percevoir ou d'inférer de l'information, et de la conserver comme une connaissance à appliquer à des comportements adaptatifs dans un environnement ou un contexte. »

Chacun pourra bien évidemment trouver une autre définition susceptible de conforter sa conviction ou sa vision. Cette définition a cependant le mérite de reposer sur une large observation du vivant et de s'appuyer sur des concepts fondamentaux, comme la capacité à percevoir son environnement, à apprendre et à s'adapter.

Et cela fonctionne ! En imitant le vivant, les stratégies basées sur l'apprentissage se sont progressivement imposées et le *Machine Learning* est devenue une composante essentielle de l'intelligence artificielle.

Une autre conception de l'intelligence, reposant sur des concepts de haut niveau, est également possible, dans laquelle l'intelligence peut être définie comme :

> « [L']Ensemble des fonctions mentales ayant pour objet la connaissance conceptuelle et rationnelle »[2]

Cette vision nous emmène loin des mécanismes basiques vus précédemment et ces deux définitions reflètent deux perceptions et approches fondamentalement opposées de l'intelligence (artificielle).

1. Wikipedia : https://en.wikipedia.org/wiki/Intelligence, traduction via http://deepl.com

2. Larousse : https://www.larousse.fr/dictionnaires/francais/intelligence/43555

2. Deux conceptions de l'intelligence

Concevoir des machines capables de (re)produire des pensées intelligentes est un rêve très ancien. Deux approches antagonistes se sont affrontées durant près de 60 ans.

2.1 L'approche connexionniste

L'idée que l'intelligence pouvait être le résultat d'un ensemble de mécanismes fondamentaux massivement parallèles est apparue dans les années 40.

Le mouvement **connexionniste**, porteur de cette vision, s'est appuyé sur l'étude du vivant, avec l'observation des neurones.

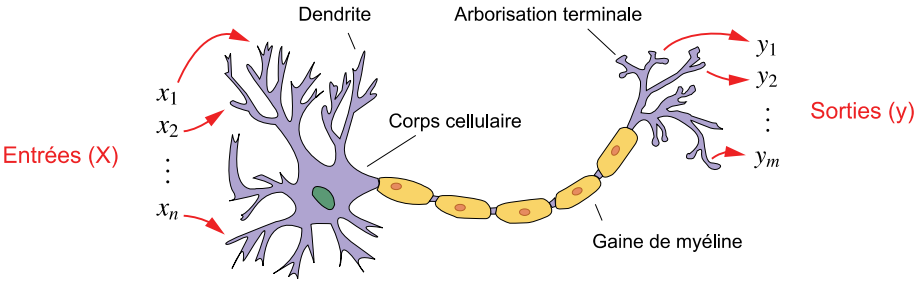


Figure 1 - Neurone naturel

Chaque neurone possède un grand nombre d'entrées et de sorties, lui permettant de constituer un réseau complexe en couches successives.

Le fonctionnement d'un **neurone artificiel**, proposé par Warren McCulloch et Walter Pitts en 1943 [4], est directement inspiré de son pendant naturel :

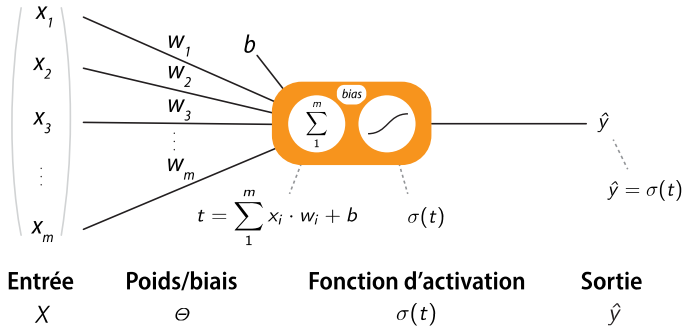


Figure 2 - Neurone artificiel

Le fonctionnement est le suivant :

- Chaque neurone recevant une valeur X en entrée va calculer une somme pondérée t à partir d'un ensemble de poids w_i et d'un biais b qui lui sont propres : $t = \sum_1^m x_i \cdot w_i + b$
- Cette somme pondérée sera ensuite soumise à une « fonction d'activation », dont le résultat constituera la sortie du neurone : $\hat{y} = \sigma(t)$

Une notation compacte peut s'écrire : $y = \sigma(\Theta^T \cdot X + b)$

► L'apprentissage, base de l'intelligence

L'objectif de l'apprentissage va être de modifier progressivement des neurones de manière à obtenir un comportement donné.

Dans le cas des neurones artificiels, on va modifier graduellement les différents poids $\theta = (w_1, \ldots, w_m)$ et le biais b, de manière à ce que les sorties effectives d'un neurone correspondent à des valeurs attendues.

Le principe est le suivant :

- À partir d'un ensemble de n données d'apprentissage (X_i, y_i) $i \in [0, n]$,
- on va calculer une prédiction $\hat{y}_i$ pour chaque valeur X_i

  soit $\hat{y}_i = \sigma(t)$

  avec $t = \sum_{k=1}^{m} x_k \cdot w_k + b$

  et σ une fonction d'activation, par exemple $\sigma(t) = \frac{1}{1+e^{-t}}$
- En comparant ces prédictions $\hat{y}_i$ aux valeurs attendues y_i on va calculer une fonction d'erreur, telle que l'erreur quadratique moyenne :

  $$E_\Theta(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} [\hat{y}_i - y_i]^2$$
- On modifiera nos poids w_i de manière à minimiser cette fonction d'erreur, par une méthode de descente de gradient :

  $$\Theta \leftarrow \Theta - \eta \cdot \frac{\delta E}{\delta \Theta}$$

Cela correspond à une régression logistique et on peut parfaitement considérer la régression logistique comme étant un réseau de neurones à… 1 neurone.

Des exemples de régression linéaire [NB2] et de régression logistiques [NB4] sont proposés dans les notebooks qui accompagnent cet article.

► Une conception inductive de l'intelligence

Dans cette approche, l'information est considérée comme un **simple signal** qui ne comporte pas de sens. L'approche est **inductive** : à partir de faits observés, on va déterminer des règles.

Figure 3 - Principe d'une approche inductive

Le « programme », que l'on appellera « modèle », est construit à partir des données.

Toute l'intelligence reposant sur l'apprentissage, on peut dire que « l'intelligence est dans les données », ouvrant la voie à un profond changement de paradigme, théorisé notamment par Jim Gray [2].

2.2 L'approche cognitiviste ou symbolique

Cette seconde conception de l'intelligence est fondamentalement différente et s'est considérablement développée avec l'essor des ordinateurs. L'information est considérée ici comme une donnée de haut niveau, dotée d'un sens symbolique et sémantique, que l'on va pouvoir traiter avec un ensemble de règles :

```
Tout [homme] est [mortel]
[Socrate] est un [homme]
donc [Socrate] est [mortel]
```

L'approche symbolique a été portée par des chercheurs comme John McCarthy, inventeur du langage LISP en 1958, ou encore James Slagles, inventeur du premier système expert en 1961.

La démarche est ici typiquement déductive :

Figure 4 - Principe d'une approche déductive

À partir d'un programme, issu d'une expertise, on va pouvoir calculer une sortie correspondant à une entrée.

Le programme est ici un ensemble de règles et d'instructions qui s'appliquent à des informations dotées d'un sens, qu'il soit intrinsèque ou attribué.

Cette approche est celle de notre informatique « traditionnelle ».

3. 60 ans de controverse

Ces deux conceptions de la pensée intelligente se sont affrontées pendant plus d'un demi-siècle, jalonnant l'histoire de l'intelligence artificielle de nombreuses périodes d'euphorie …et de glaciation.

La « victoire des neurones », avec l'avènement des réseaux de neurones profonds, n'est que l'ultime épisode de cette longue controverse [3].

3.1 Le premier âge des neurones

Le courant connexionniste est apparu dans les années 1940, avec notamment les travaux de McCulloch et Pitts en 1943, qui posèrent les premières bases des réseaux de neurones artificiels [4], et de Hebb en 1949 qui propose une méthode permettant à ces neurones de se doter d'une capacité d'apprentissage [5].

Cette période coïncide également avec les conférences de Macy (1941-1960) dont l'un des objectifs était de jeter les bases d'une « science générale du fonctionnement de l'esprit humain » autour de la « cybernétique », qui conduira elle-même à l'émergence des sciences cognitives et de l'intelligence artificielle.

L'aboutissement de cette volonté de « modéliser le cerveau » amènera à l'invention du **Perceptron** par Frank Rosenblatt [6], en 1957, dont la conception et le mode de fonctionnement est très proche de nos neurones artificiels modernes :

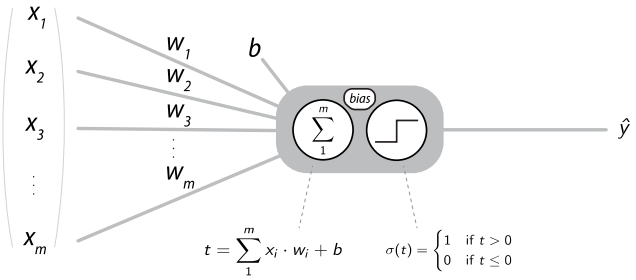


Figure 5 - Perceptron, 1957

Le Perceptron était un « classifieur » à un seul neurone, capable d'apprendre.

Un exemple de Perceptron est disponible dans les notebooks [NB6]

Toutefois, n'étant constitué que d'un seul neurone, il possédait de nombreuses limitations dont la principale était de ne pouvoir effectuer que des classifications linéaires.

Il constituait néanmoins une avancée majeure et engendra un optimisme un rien excessif :

> « L'armée américaine a révélé un embryon de ce qui pourrait être une machine qui marche, parle, voit, écrit, se reproduit et serait consciente d'elle-même » New-York Times, 1958

3.2 Premier hiver

Tandis que les « connexionnistes » développaient le Perceptron avec un certain succès et beaucoup de promesses, les apôtres de l'approche « symbolique » s'organisaient.

Le développement des premiers ordinateurs et leur montée en puissance permit l'écriture de programmes de plus en plus complexes. L'invention du langage LISP, par John McCarthy en 1958 et des premiers systèmes experts permirent d'importants progrès.

Petit à petit, les « symbolistes » monopolisèrent les ressources de la DARPA[3] au niveau des grosses universités : Minsky et Papert au MIT, McCarthy à Stanford, Simon et Newel à Carnegie Mellown…

Le terme même d'« intelligence artificielle » sera proposé par Minsky en 1956, lors d'ateliers à l'université de Darmouth.

En 1969, Minsky et Papert publient « Perceptron », un ouvrage dans lequel ils dénoncent les limitations conceptuelles du Perceptron, entraînant la fin de celui-ci.

Déjà fortement marginalisé, le courant connexionniste tombe en désuétude et mettra plus de 20 ans à s'en remettre.

Mais l'histoire ne s'arrête pas là et les symbolistes promirent également le ciel et la terre : que l'on pourrait prochainement traduire automatiquement le russe en anglais (guerre froide oblige), battre les meilleurs joueurs aux échecs ou faire des chars d'assaut autonomes…

On était en 1970 et rien de tout cela n'arriva. Les promesses non tenues se transformèrent en déceptions et les déceptions en rapports et audits. Petit à petit les financements se tarirent et en ce début des années 70, ce fut bientôt toute l'intelligence artificielle naissante qui disparut dans la brume de ses rêves.

3.3 Le temps et la fin des dinosaures

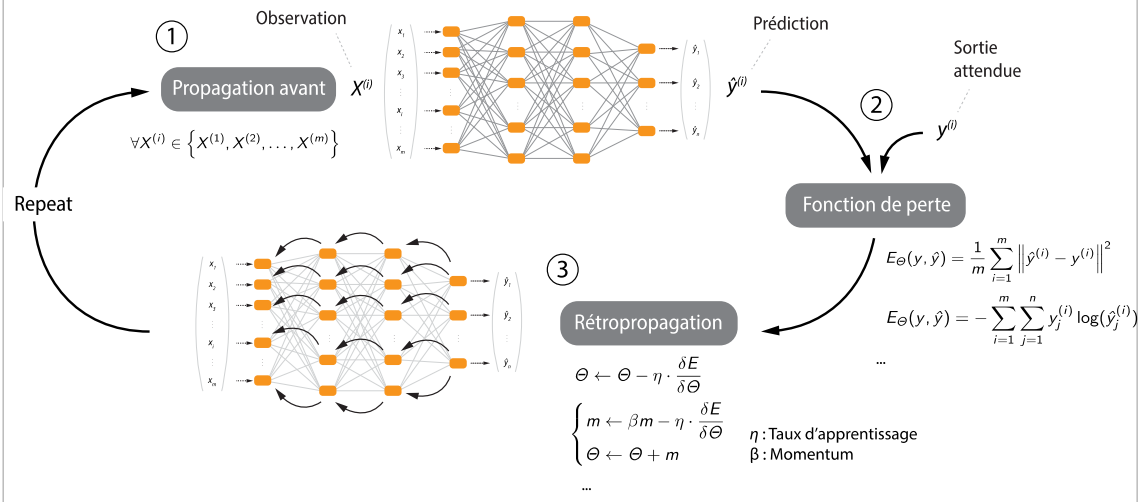
Mais l'histoire ne s'arrête pas là. Durant les années 80, la puissance des ordinateurs ne cesse de progresser, permettant aux systèmes experts de gagner en performance et de recoloniser massivement le paysage de l'intelligence artificielle en plein dégel.

Cependant, même si les symbolistes règnent en maîtres et sans partage sur le monde scientifique [3], deux percées majeures vont changer le cours des choses…

En 1986, Rumelhart, Hinton et William proposent un algorithme de **rétropropagation du gradient** [8], permettant d'étendre l'apprentissage à des réseaux de neurones complexes, de plusieurs couches, capables de traiter des problèmes non linéaires. L'obstacle soulevé en 1969 par Minsky et Papert, presque 20 ans auparavant, n'est plus.

3. « Defense Advanced Research Projects Agency », Agence de la Défense des États-Unis chargée de la recherche et du développement des nouvelles technologies.

Le principe de la rétropropagation est une généralisation de la descente de gradient :



Principe de la rétropropagation (backpropagation)

L'apprentissage est un processus itératif qui se fait à partir d'un ensemble de m données d'apprentissage $(X^{(i)}, y^{(i)}), i \in [0, m]$.

Lors de chaque itération, appelée époque :

① Les observations $X^{(i)}$ sont soumises au réseau qui fournit un ensemble de prédictions $\hat{y}^{(i)}$

② L'écart entre les prédictions $\hat{y}^{(i)}$ et les valeurs attendues $y^{(i)}$ est mesuré avec une fonction de perte $E_\Theta(y, \hat{y})$

③ Les poids de chaque neurone sont modifiés à partir du gradient de la fonction de perte. La correction est rétro-propagée depuis la sortie du réseau, vers les neurones d'entrée.

Au fil des épisodes et des corrections, les poids vont progressivement converger vers des valeurs permettant de minimiser la fonction de perte.

À intervalle régulier, par exemple à chaque époque, on effectuera une évaluation de l'apprentissage en cours avec un second ensemble de données qui ne serviront que pour l'évaluation.

La seconde avancée majeure, proposée en 1989 par LeCun *et al. [9]*, concerne les réseaux de neurones **convolutifs**, qui finiront par propulser les réseaux de neurones profonds sur le devant de la scène, en révolutionnant la classification d'image.

Tandis que les connexionnistes posaient les bases de l'apprentissage profond (*deep learning)*, les machines LISP qui avaient dominé le monde de l'intelligence artificielle depuis la fin des années 70, s'éteignaient. Victimes de la complexité du monde réel qu'ils ne pouvaient appréhender efficacement et incapables de s'adapter, les systèmes experts, vénérables dinosaures de la vision symbolique, perdaient leur suprématie avec la fin des années 80…

3.4 Second hiver

L'approche symbolique ayant atteint ses limites, les connexionnistes purent revenir sur le devant de la scène. Mais si les bases théoriques de l'apprentissage profond étaient posées, la puissance des machines et l'absence de gros corpus de données rendaient son usage difficile et peu intéressant.

Sur des jeux de données de taille moyenne, une autre branche connexionniste allait s'imposer à la fin des années 90, avec les travaux de Vladimir Vapnik sur les « machines à vecteurs de support » (*Vector Support Machine* ou SVM).

Efficaces dans le contexte des années 1990/2000, moins gourmandes en ressources et garantissant mathématiquement une solution lors du processus de minimisation de la fonction de perte, les SVMs s'imposèrent et marginalisèrent à nouveau les réseaux de neurones…

Ce second hiver allait durer jusqu'au début des années 2010…

3.5 Le réveil et la victoire des neurones

Ce qui avait conduit les systèmes experts à ses limites en fit de même avec SVM.

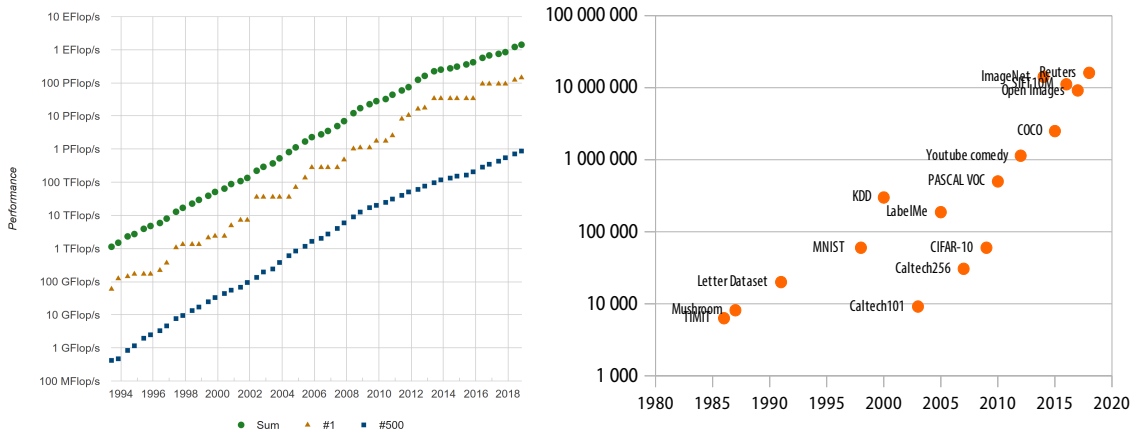


Figure 6 - Evolution des performances[10] et des tailles de datasets[11]

Entre 1995 et 2015, la puissance des machines va être multipliée par 10^6, permettant de travailler sur des jeux de données de plus en plus conséquents.

Avec Internet, de nouveaux acteurs apparurent, avec des besoins aussi vastes que leurs moyens (Google, Facebook, Amazon, etc.) et de vastes corpus de données sont constitués, issus du « vrai » monde.

Face à ce changement d'échelle et à la complexité du monde réel, SVM atteindra à son tour ses limites.

Symboliquement, on peut retenir l'année 2012 comme un point de bascule, lorsque lors de la compétition de classification d'image ILSVRC [12], un réseau de neurones convolutif AlexNet [13], conçu par Alex Krizhevsky alors étudiant en thèse, bat haut la main les meilleures équipes du moment.

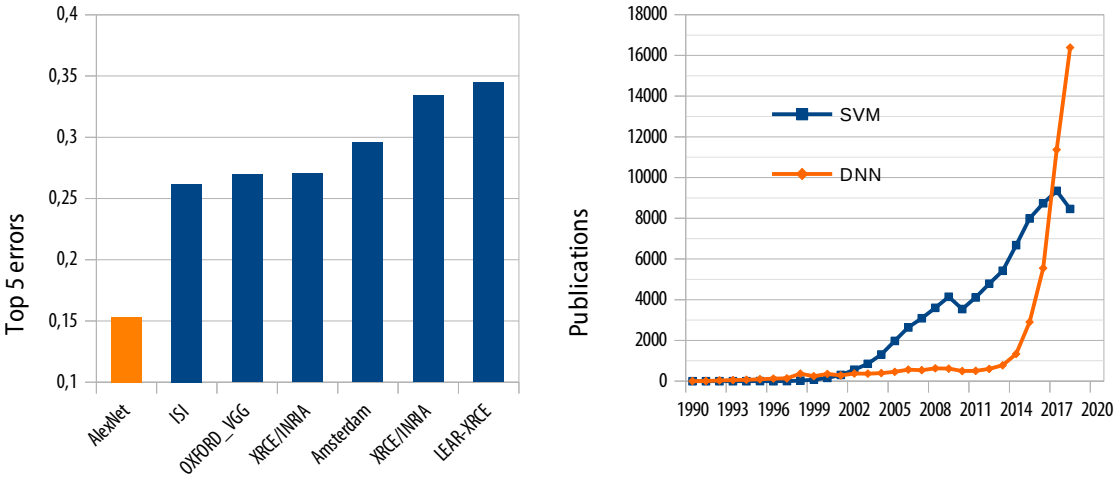


Figure 7 - ILSVRC 2012 et Evolution du nombre de publications[14]

Ce qui arriva pour la classification d'images arriva dans la plupart des autres domaines : traitement automatique du langage, traduction automatique, robotique, jeux, etc. En l'espace de quelques années, l'approche connexionniste basée sur les réseaux de neurones va s'imposer dans une multitude de domaines, quittant les laboratoires et gagnant notre quotidien.

Winter has gone.

4. Des neurones et des données

Les cas d'usage sont aujourd'hui innombrables et les exemples qui suivent ne sont que des illustrations de quelques-unes des grandes familles de réseaux de neurones.

4.1 Réseaux de neurones complètement connectés

Le plus simple des réseaux de neurones profonds est celui où chaque entrée est connectée à un neurone.

Un exemple classique est la classification de chiffres manuscrits issus de la base MNIST [15], composée de 70 000 images de chiffres. Dans notre exemple, 60 000 images servent à l'apprentissage, 10 000 à la validation[4]. Chaque pixel correspond à un neurone de la couche d'entrée :

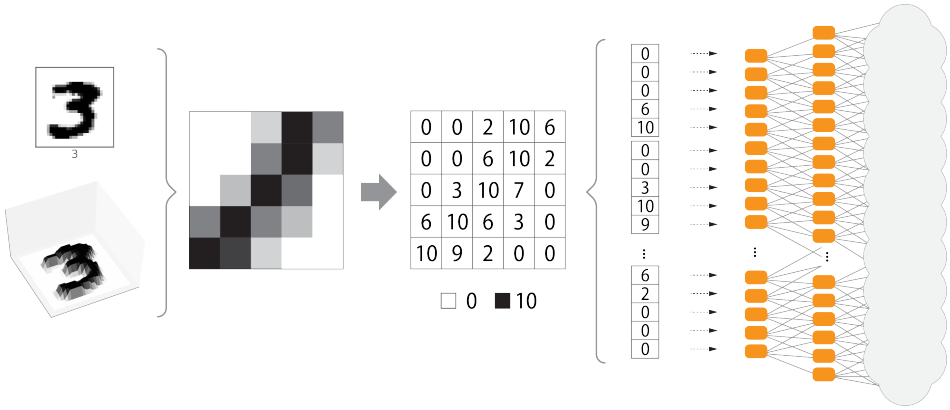


Figure 8 - Entrées d'un réseau complètement connecté

Les images sont en noir et blanc et comportent 28x28 pixels. Elles sont vues comme de simples matrices. Ces dernières sont transformées en vecteurs (colonne par colonne ou ligne par ligne) et le vecteur est pris comme entrée du réseau. La couche d'entrée comporte donc 28x28=784 neurones.

Celle-ci est entièrement connectée avec des couches cachées et une couche de sortie :

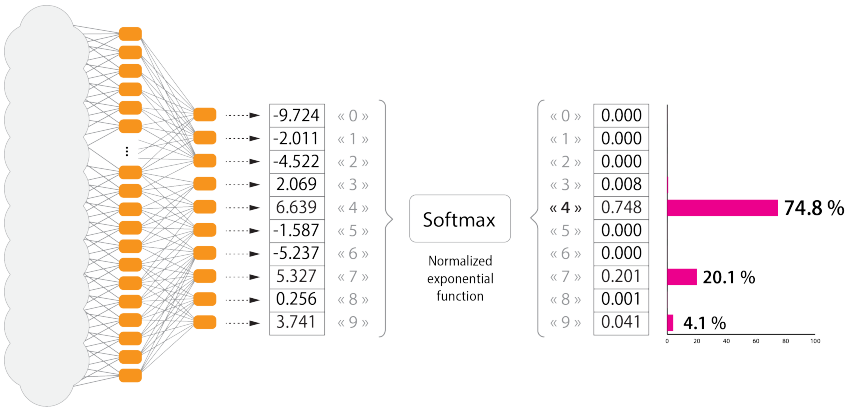


Figure 9 - Sorties d'un réseau classifieur

4. La validation sert à évaluer le résultat et ne peut se faire qu'avec des données distinctes de l'apprentissage.

Dans le cas d'un réseau classifieur, chaque classe est représentée par un neurone au niveau de la couche de sortie. La valeur produite par ces neurones est normalisée via une fonction softmax[5]. Le résultat final est interprété comme la probabilité d'appartenance à chaque classe.

Une implémentation de cet exemple est disponible dans les notebooks [NB7].

Les données sont séparées en deux sous-ensembles : *train* et *test*. L'apprentissage se fait par itération avec le corpus *train*. Chaque itération est appelée une époque (*epoch*) ou un épisode. L'évaluation (validation) est effectuée avec le second corpus, *test.*

Les courbes de la précision[6] (*accuracy*) et de la fonction de perte (*loss*) permettent de suivre la progression de l'apprentissage :

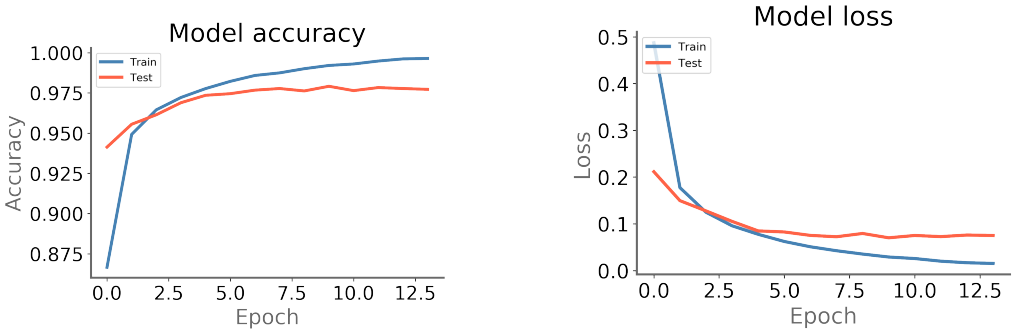


Figure 10 - Courbes d'apprentissage

Au-delà d'une certaine durée d'apprentissage, on peut observer que la précision continue de progresser avec le corpus d'apprentissage tandis qu'elle stagne ou se met à régresser avec le corpus de test. On parle alors de sur-apprentissage (*overfitting*).

Dans notre exemple, nous obtenons une précision globale de 97,7 %. Celle-ci peut être analysée plus finement en visualisant la matrice de confusion :

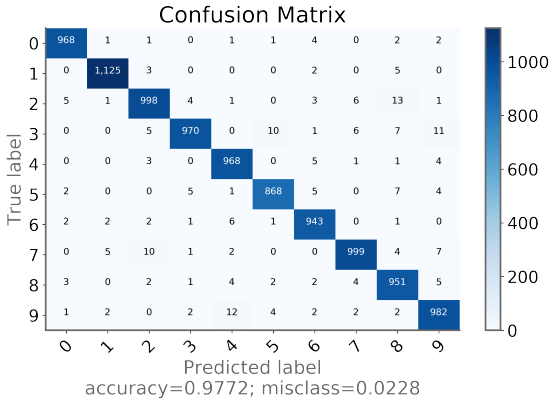


Figure 11 - Matrice de confusion

5. La fonction softmax $\sigma(y_i)$, ou *fonction exponentielle normalisée*, est une généralisation de la fonction logistique.
Pour une sortie y d'un réseau classifieur de K classes : $\sigma(y_j) = \frac{e^{y_j}}{\sum_{k=1}^{K} e^{y_k}}$ pour $j \in \{1, \ldots, K\}$

6. La précision est le rapport : $\frac{\text{Nombre de prédictions justes}}{\text{Nombre de prédictions}}$

4.2 Réseaux convolutifs (CNN)

Disposer d'un neurone d'entrée pour chaque pixel est faisable tant que le nombre de pixels reste limité. Avec des images de grande dimension ce n'est plus possible. L'utilisation de réseaux convolutifs permet de répondre à cette problématique, tout en apportant une bien meilleure précision.

Le principe de la convolution est de construire une nouvelle image, appelée couche de convolution, où chaque pixel est construit à partir des pixels environnants :

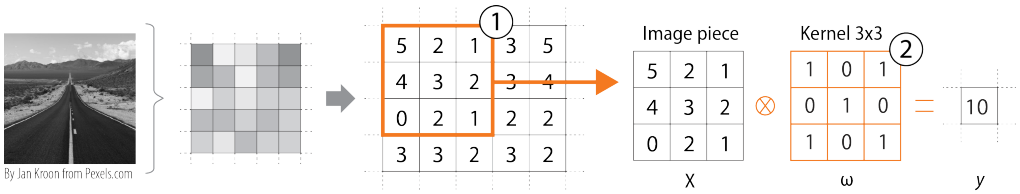


Figure 12 - Convolution 2D

On balaye notre image en calculant la somme d'une portion de l'image ① pondérée par les éléments d'un noyau (kernel) ②.

Soit : $y = \sum_{i=1}^{n} \sum_{j=1}^{m} x_{i,j} \cdot \omega_{i,j}$ où (n, m) est la dimension du kernel ω

Ce principe peut être étendu avec des images possédant plusieurs canaux, par exemple RVB. Dans ce cas les kernels sont de dimension >2.

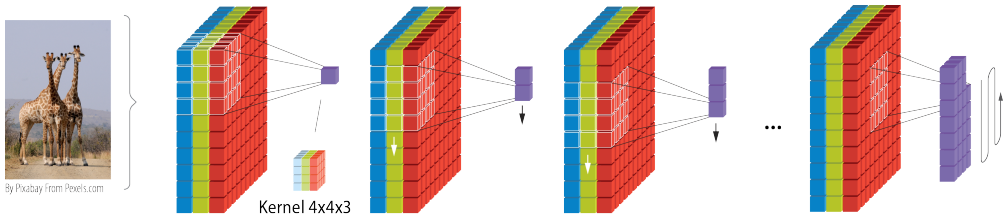


Figure 13 - Convolution 3D

On effectue une succession de convolutions et de *pooling*, le *pooling* consistant à réduire la dimension des couches de convolution. Les couches terminales sont aplanies puis connectées à un réseau dense « classique ».

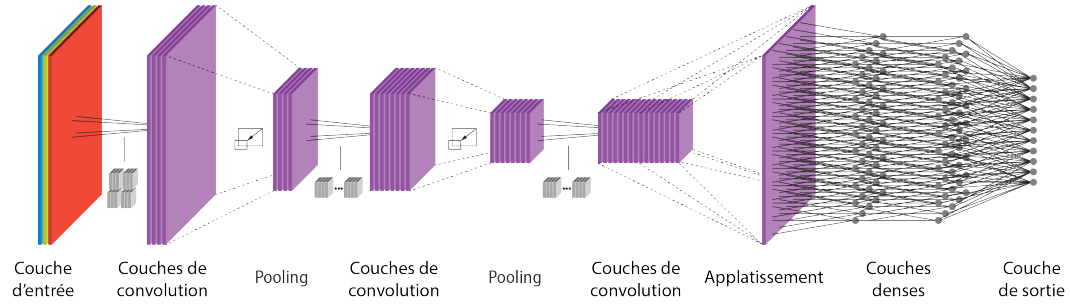


Figure 14 - Principe d'un réseau convolutif (CNN)

Trois exemples sont disponibles : une implémentation de CNN avec Keras [NB9] et l'utilisation de modèles en Javascript, avec Tensorflow.js [NB10].

4.3 Données creuses, *text embedding*

Les données de type textuelles posent elles aussi un problème de dimension.

La manière usuelle de numériser un texte consiste à construire une matrice, où chaque colonne référence un mot par rapport à un dictionnaire :

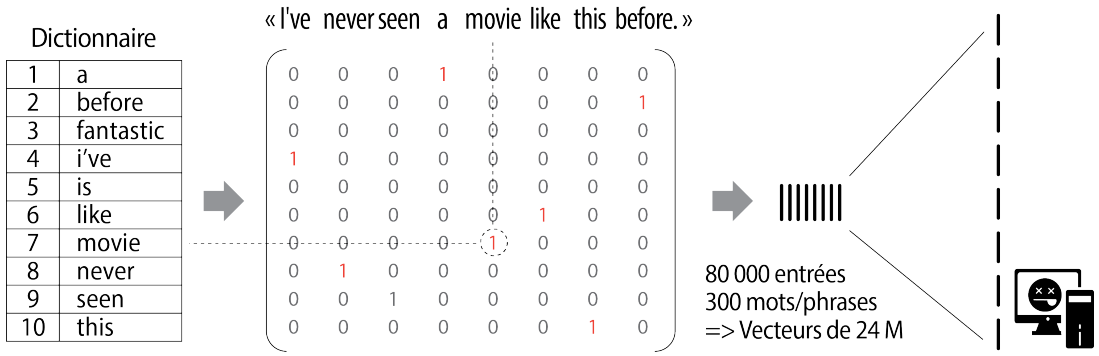


Figure 15 - Traitement du langage naturel et matrices creuses

En traitement automatique du langage naturel, tout est comptabilisé comme un mot : abréviations, approximations, expressions, etc. et les dictionnaires deviennent rapidement énormes, conduisant à des matrices inutilisables.

Pour pallier cela, une technique, nommée *word embedding* ou « plongement de mots » a été proposée en 2013 par Tomas Mikolov *et al.*[16] L'idée est de représenter chaque mot du dictionnaire par un vecteur dense de dimension réduite.

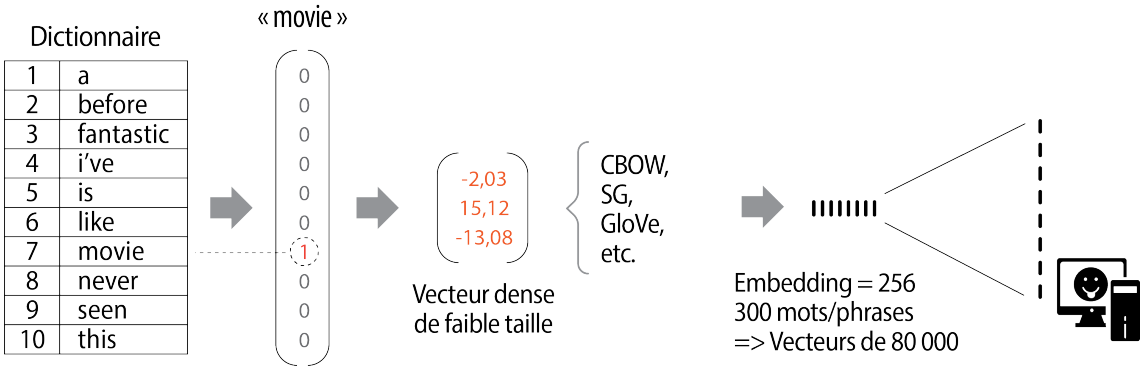


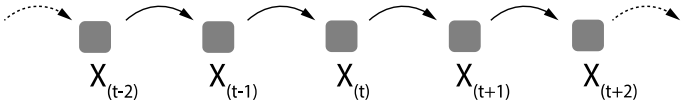
Figure 16 - Word Embedding, ou plongement de mots

Les vecteurs de chaque mot sont construits en tenant compte du contexte de chaque mot par apprentissage non supervisé via un réseau de neurones. Plusieurs méthodes sont disponibles, telles que CBOW et Skip-Gram [16] ou GloVe [17].

Un *notebook* d'illustration est disponible [NB12]. Dans cet exemple, l'objectif est de déterminer si des critiques de films, issues de la base IMDB, sont positives ou négatives. Le taux de réussite, après apprentissage, est d'environ 87 %.

4.4 Réseaux de neurones récurrents (RNN)

Un grand nombre de données sont évolutives ; trajectoires, spectres audios, fonctions temporelles, données de marché boursier, etc.



Ces données peuvent être vues comme un état qui évolue dans le temps.

Pour traiter ces données en intégrant cette dimension récursive, nous utilisons des neurones particuliers, dont la sortie est reprise en entrée :

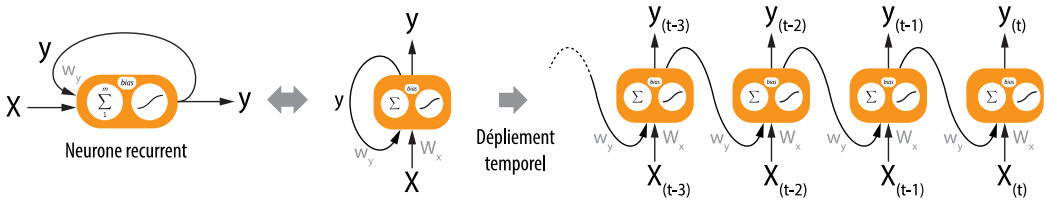


Figure 17 - Neurone récurrent

La sortie d'un tel neurone est calculée de la manière suivante :
$y_{(t)} = \sigma(W_x^T \cdot X_{(t)} + w_y \cdot y_{(t-1)} + b)$ où W_x et w_y sont les poids associés aux entrées.

Ce modèle est généralisé à des couches de neurones et l'on parle alors de « cellules » :

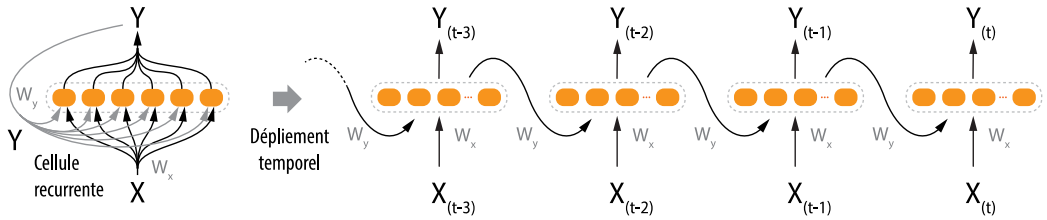


Figure 18 - Cellules récurrentes

Avec $Y_{(t)} = \Phi\left(W_x^T \cdot X_{(t)} + W_y^T \cdot Y_{(t-1)} + b\right)$ où W_y est le vecteur poids de Y

Ce modèle simple de cellule récurrente permet de bien comprendre le principe, mais n'est pas très efficace. Des cellules plus évoluées ont été proposées, notamment LSTM par Hochreiter et Schmidhuber en 1997 [18] et GRU par Kyunghyun, en 2014 [19].

Elles peuvent être utilisées de différentes manières :

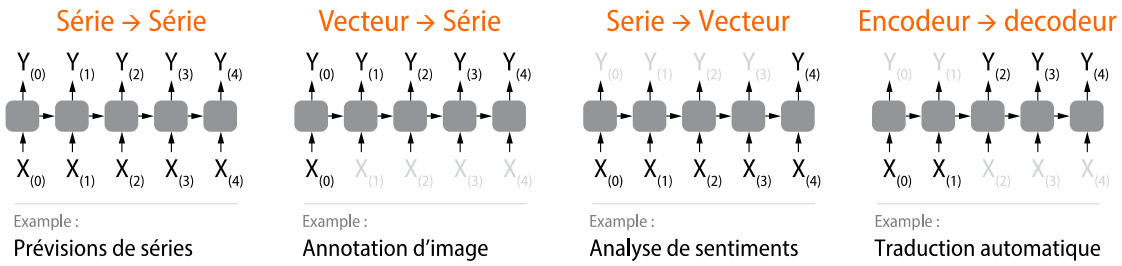


Figure 19 - Utilisation des réseaux récurrents

Un notebook d'illustration, de type Série vers série, est proposé [NB13]

Dans l'exemple suivant, l'objectif est de déterminer la suite d'une série de données dont la nature est totalement inconnue, à partir d'un apprentissage.

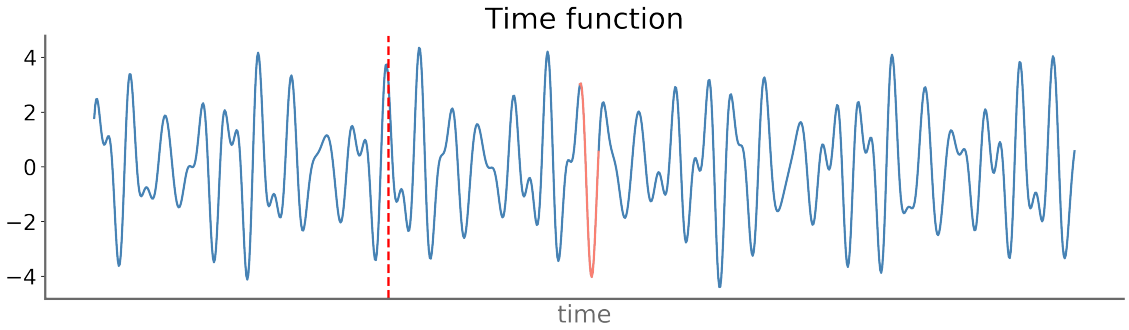
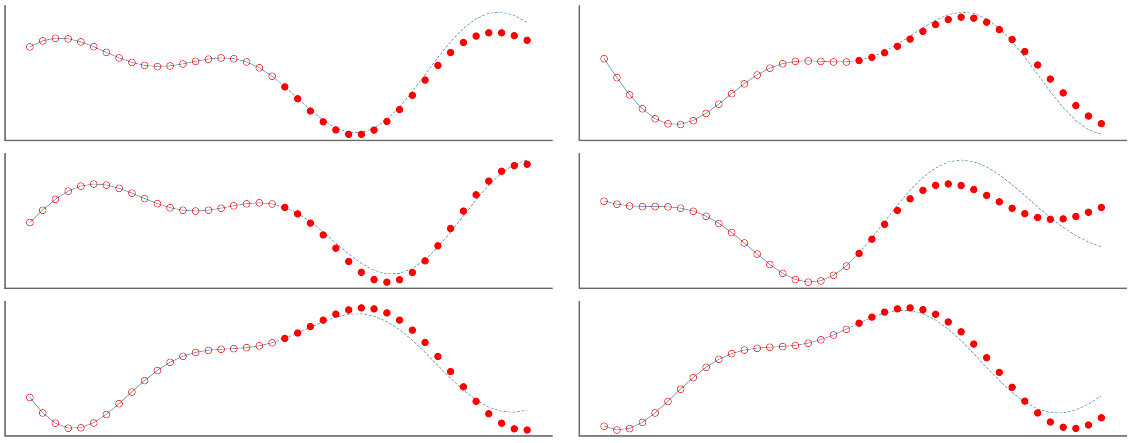


Figure 20 - Exemple de séquence temporelle

L'apprentissage est effectué à partir des données de la partie gauche de la courbe, par rapport au pointillé rouge.

L'implémentation est effectuée avec Tensorflow et utilise des cellules LSTM.

Après apprentissage (<1 minute), il est possible d'effectuer des prévisions de type « grandes tendances »



Les ○ réprésentent un début de séquence connu, les ● la suite prédite.

Figure 21 - Prévisions de séquence

4.5 Apprentissage par renforcement profond (RL)

L'objectif de l'apprentissage par renforcement est de permettre à un agent autonome, typiquement un robot, de développer une stratégie lui permettant d'interagir « positivement » vis-à-vis de son environnement.

L'agent va effectuer une action sur son environnement et, en retour de cette action, il reçoit une observation et une récompense associée. L'apprentissage consistera pour l'agent à adapter sa stratégie de manière à maximiser les récompenses.

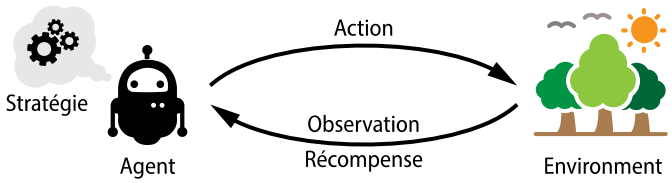


Figure 22 - Principe de l'apprentissage par renforcement

L'apprentissage par renforcement ne nécessite pas nécessairement de neurones. On parle d'apprentissage par renforcement profond lorsqu'un réseau de neurone est utilisé pour conserver l'expérience acquise.

Plusieurs techniques existent, *Policy gradient*, *Deep Q-Network* (DQN), *Markov Decision Processes* (MDP), etc.

L'exemple détaillé ci-dessous est disponible dans les notebooks [NB14]. Il s'agit d'une solution à un problème proposé en 1983 par Barto, Sutton et Anderson [20].
L'apprentissage est effectué par *Policy gradient*.

Un robot doit parvenir à maintenir un pendule inversé en équilibre :

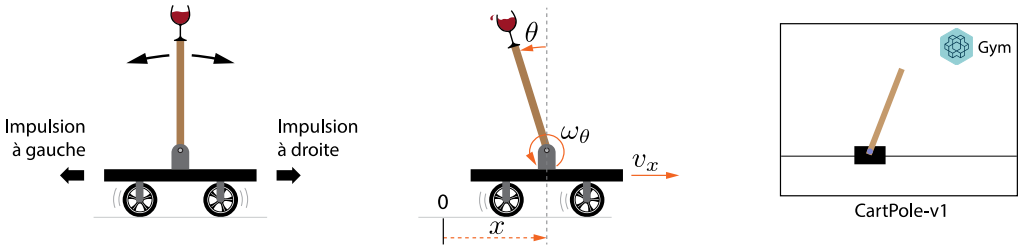


Figure 23 - Problème du pendule inversé. À droite le simulateur Gym.

Le robot n'a que 2 actions possibles : pousser le chariot à gauche ou à droite. Les observations qu'il reçoit sont au nombre de 4 : la position x et la vitesse v_x du chariot, la position θ et la vitesse angulaire ω_θ du pendule. Son objectif est d'éviter que le pendule penche au-delà d'un angle donné ou que le chariot sorte de la scène… et cela le plus longtemps possible !

Pour des raisons pratiques, l'apprentissage ne se fera pas dans le monde réel, mais avec le simulateur d'environnements *Gym* proposé par OpenAI [21].

Le fonctionnement de notre robot est le suivant :

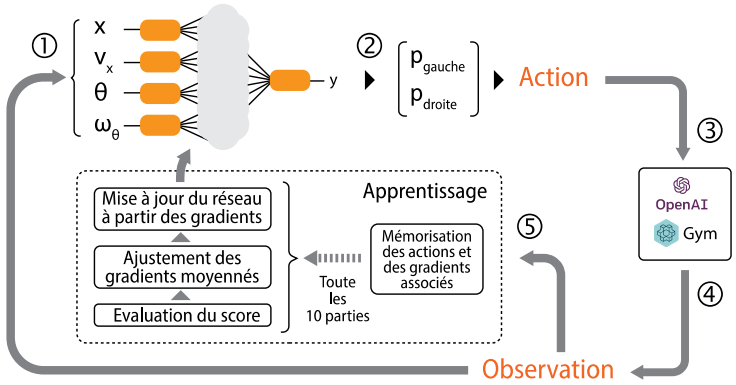


Figure 24 - Apprentissage par renforcement avec policy gradient

① Le réseau de neurones de notre agent reçoit en entrée les observations de son environnement.

② La sortie du réseau est interprétée comme une probabilité de devoir pousser vers la gauche ou la droite, mais l'action retenue ne sera pas celle correspondant à la plus forte probabilité. Elle sera tirée au sort en fonction de cette probabilité. Ce « droit à l'erreur » est indispensable, car il permet d'explorer le champ des possibles.

③ L'action retenue est communiquée au simulateur (ie. Appliquée à l'environnement)

④ De cette action découle une observation, qui sera à son tour communiquée à la couche d'entrée du réseau de neurones. Le cycle se poursuit pour constituer une partie qui terminera avec la chute du pendule ou une sortie de scène.

⑤ Les observations issues de l'environnement servent également au processus d'apprentissage :

- l'ensemble des actions, des gradients et des récompenses associées sont conservées,
- toutes les 10 parties, on va calculer des gradients pondérés par les récompenses, les moyenner et les appliquer au réseau.

Dans notre exemple, l'apprentissage est effectué sur 200 épisodes et prend environ 15 minutes sur un ordinateur portable correct.

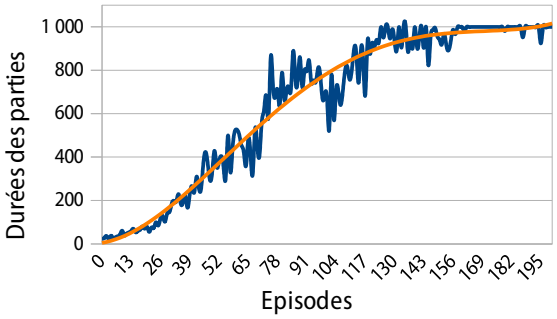


Figure 25 - Courbe d'apprentissage

À l'issue de l'apprentissage, notre robot parvient parfaitement à maintenir le pendule en équilibre, sans sortir de la scène !

Le plafonnement horizontal de la courbe est lié au fait que les parties sont arbitrairement interrompues après 1000 itérations.

4.6 Réseaux antagonistes génératifs (GAN)

L'idée de pouvoir faire progresser des joueurs en les mettant en compétition est ancienne, mais la transposition de ce concept avec des réseaux de neurones est relativement récente et date de 2014, avec les travaux de Ian Goodfellow [22].

Le principe est le suivant :

Figure 26 - Réseau antagoniste génératif (GAN)

Un premier réseau, appelé générateur *(générator)* génère une donnée, par exemple une image, et un second réseau, le discriminateur (*discriminator*) essaye de détecter si cette donnée est réelle ou issue du générateur. Les deux réseaux apprennent à partir de leurs erreurs et le discriminateur s'entraîne en plus avec un corpus de données réelles.

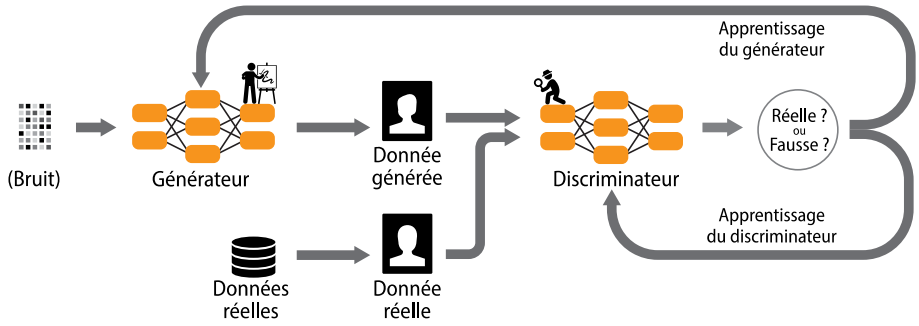


Figure 27 - Architecture d'un réseau antagoniste génératif

Cette approche, relativement récente, constitue l'une des plus importantes avancées en intelligence artificielle et un très grand nombre de variantes ont été proposées : DCGAN, cGAN, WGAN, CycleGAN, ProGAN, StyleGAN, BigGAN, GANSynth, MaskGAN, etc.

Les architectures GAN nécessitant beaucoup de ressources lors de l'apprentissage, elles sont peu adaptées à des exemples sur un poste de travail basique. Il est cependant possible de trouver un grand nombre d'illustrations de leurs étonnantes possibilités sur Internet :

- « This X Does Not Exist » : https://thisxdoesnotexist.com/ ;
- « Which Face Is Real ? » : http://www.whichfaceisreal.com/ ;
- « Colorful Image Colorization » : https://richzhang.github.io/colorization/ ;
- « Image-to-Image Translation » : https://github.com/junyanz/CycleGAN ;
- « Pixel Level Domain Transfer » : https://github.com/fxia22/PixelDTGAN ;
- « DeOldify project » : https://github.com/jantic/DeOldify.

5. Un monde ouvert et une nouvelle donne

L'explosion du *deep learning* s'exprime également dans une compétition très forte sur le plan scientifique et en ingénierie logicielle.

5.1 Un développement rapide et une forte compétition

Le nombre de publications scientifiques dans le domaine de l'intelligence artificielle a pratiquement été multipliée par 10 en 20 ans et reste massivement dominé par l'Amérique du Nord, l'Europe et la Chine.

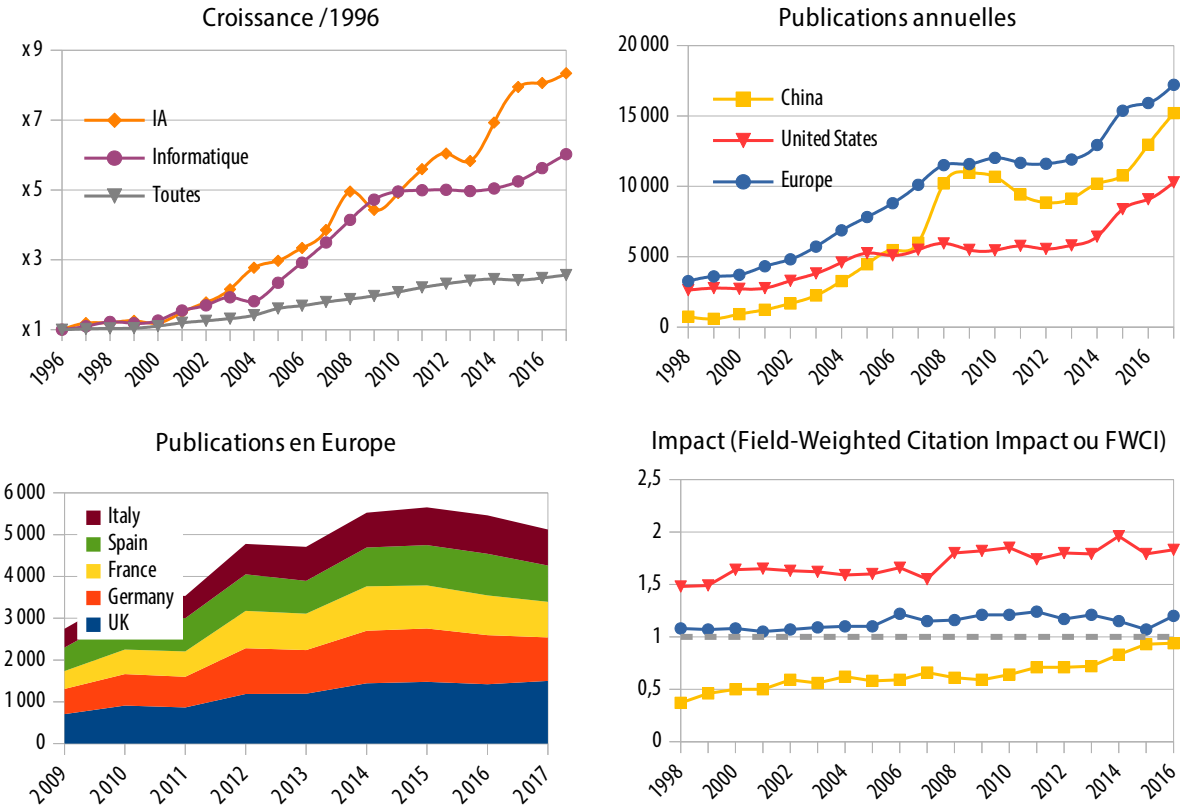


Figure 28 - Publications en Intelligence Artificielle [23]

Si l'Europe (élargie) est le plus gros contributeur en nombre de publications, l'impact FWCI[7] des publications nord américaines est largement supérieur. Au sein de l'Europe, le Royaume-Uni est le premier contributeur européen, suivi par l'Allemagne. France, Espagne et Italie suivent à un niveau équivalent. La qualité et le nombre des publications chinoises est en forte augmentation, portés par une stratégie de développement très forte et des moyens considérables.

7. L'indicateur FWCI (Field-Weighted Citation Impact) ou CNCI (Category Normalized Citation Impact) mesure l'impact relatif d'une publication au sein d'une discipline.

5.2 Un monde centré autour de Python

La conception et le développement d'un modèle est rapidement très complexe et nécessite d'importantes ressources : calcul distribué, GPU, etc. L'utilisation d'environnements de développement *(frameworks)* est de ce fait incontournable.

La plupart, des environnements disponibles ont moins de 5 ans, sont ouverts, centrés autour de Python (ou R). Ils sont issus ou supportés par les grands acteurs de l'Internet pour leurs besoins propres.

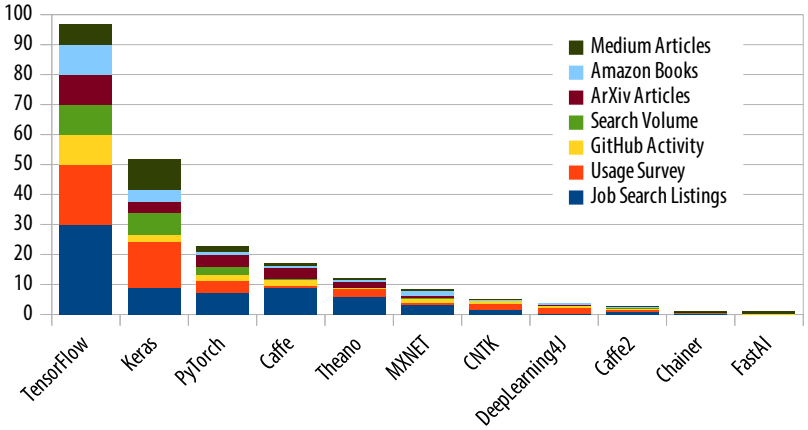


Figure 29 - Popularité 2018 des principaux frameworks d'apprentissage profond [24]

TensorFlow est incontestablement le poids lourd des *frameworks* de *deep learning*. Il est le plus actif sur GitHub, le plus cité sur ArkXiv, Medium ou encore dans les requêtes Google. Il possède également la plus grande communauté de développeurs et est mentionné dans la plupart des annonces d'emplois. TensorFlow est sous licence Apache, a été développé et est soutenu par Google.

Keras n'est pas un *framework* en tant que tel mais une API de haut niveau qui se place au-dessus de TensorFlow, CNTK et Theano. C'est de loin la solution la plus facile à utiliser. Depuis 2017, Keras est directement intégré dans TensorFlow. La définition des modèles est (presque !) intuitive, et l'utilisation de l'API fonctionnelle permet de définir des couches comme des fonctions. Keras a été développé par François Chollet, Chercheur chez Google et est disponible sous licence MIT.

PyTorch est un *Framework* soutenu par Facebook et dérivé de la bibliothèque Torch, sous licence BSD. En mars 2018, le projet Caffee2, également soutenu par Facebook, a fusionné avec PyTorch.

La plupart de ces frameworks sont disponible via des environnements de développement intégrés, tel que Anaconda[8].

8. Anaconda est une distribution libre et open source, des langages Python et R dédiée au calcul scientifique (apprentissage automatique, analyse prédictive, etc.). Anaconda permet de gérer environ 1500 packages scientifiques et est utilisé par plusieurs millions d'utilisateurs. https://www.anaconda.com/distribution/

5.3 Une offre de services qui se met progressivement en place

Nouvelles méthodes, nouveaux environnements et… nouveaux outils.

► Nouvelle complexité

Une contrainte issue de ces nouveaux environnements est incontestablement leur complexité sous-jacente, complexité qu'il est possible de masquer ou de conditionner via l'usage de distributions intégrées et de *notebooks*.

À titre d'exemple, l'installation de l'environnement nécessaire aux *notebooks* illustrant cet article ne demande que quelques lignes de commande[9] et environ 5 minutes… mais derrière cette apparente simplicité, il y a presque 300 000 fichiers !

Parallèlement à cette complexité, les corpus de données d'apprentissage peuvent également être considérables et devoir être couplés avec des moyens de calcul conventionnels (traitement, préparation des données, simulation, etc.). Les calculs liés aux taches d'apprentissage ont aussi leurs spécificités avec une utilisation massive de GPU et des mécanismes de distribution propres.

Toutes ces contraintes sont autant de spécificités nouvelles auxquelles nos infrastructures de calcul doivent s'adapter.

► Une nouvelle offre académique

Les offres de type « cloud » (Google, Amazon, Microsoft, etc.) n'étant pas recevables pour des questions connues de sécurité, nous ne nous intéresserons qu'aux solutions académiques.

Compte-tenu de la complexité croissante de ces nouveaux environnements de calcul, la mutualisation est plus que jamais indispensable.

Au-delà des offres locales qui apparaissent au sein des mésocentres, le GENCI[10] vient tout juste d'ouvrir (1er octobre 2019) un nouveau service national dédié à l'intelligence artificielle, avec la machine Jean Zay, hébergée et mise en œuvre par l'IDRIS[11].

Classée au 42e rang dans le Top500 [10], Jean Zay offre une puissance de 14 pétaflops et plus de 1000 GPUs Nvidia v100, dont 500 sont dédiées à l'intelligence artificielle via une nouvelle procédure « Accès Dynamique », rapide et souple[12].

TensorFlow, Keras, PyTorch et Caffe sont disponibles, avec toutes les compétences, le support et l'accompagnement des équipes de l'IDRIS. La qualité de nos offres de service académique est essentielle, compte-tenu de la concurrence des offres « clouds ».

9. La procédure est disponible sur le Gitlab des exemples : https://gricad-gitlab.univ-grenoble-alpes.fr/talks/deeplearning

10. Le GENCI est une société civile détenue à 49 % par l'Etat, 20 % par le CEA, 20 % par le CNRS, 10 % par les Universités et 1 % par l'INRIA. Le GENCI a pour objectif de favoriser l'usage du calcul intensif au bénéfice des communautés scientifiques françaises, en mettant notamment à disposition des moyens de calcul via les 3 centres de calculs nationaux (IDRIS, CINES, TGCC).

11. L'IDRIS (Institut du développement et des ressources en informatique scientifique) est une Unité Propre de Service du CNRS chargée de mettre en œuvre les moyens de calcul numérique intensif de très haute performance du CNRS et du GENCI.

12. Voir « GENCI lance l'Accès Dynamique sur le supercalculateur Jean Zay » : http://www.genci.fr/fr/node/995

6. Les limites de l'IA : je t'aime moi non plus

Ne pas aborder la question des limites de l'intelligence artificielle serait comme glisser un alligator sous le tapis en espérant que tout se passe bien…

► Opacité, efficacité et biais

La grande question de l'IA ne devrait pas être de savoir si l'on pourra faire des voitures autonomes ou des tamagotchis intelligents mais comment, sous quelle forme, avec quels contrôles ou quelles garanties nous les utiliserons.

Si l'usage massif et non raisonné des algorithmes pose déjà un grand nombre de problèmes, comme le montre Cathy O'Neil dans son livre *« Algorithmes, la bombe à retardement » [25]*, l'intelligence artificielle risque d'aggraver considérablement la situation en leur apportant opacité, efficacité et biais de manière décuplée. Kate Crawford, chercheuse à Microsoft Research et professeur à l'université de New York ne fait que renforcer cette crainte en inscrivant l'avènement de ces outils dans le contexte de la montée des monopoles, des populismes et des totalitarismes [26].

Contrairement à un algorithme traditionnel, un « modèle » issu d'un processus d'apprentissage est par nature opaque. Si ces données sont fausses, manipulées ou incomplètes, le résultat sera biaisé. Il est impossible de produire un algorithme ou d'expliquer un résultat autrement que par les données d'apprentissage.

► Le diable se cache dans les data

Le risque de biais est omniprésent et cela à tous les niveaux. Le fait que le milieu de l'IA soit dominé par des hommes, majoritairement blancs et appartenant aux catégories sociales supérieures est déjà un biais. Un exemple célèbre est celui d'Amazon qui souhaita intégrer de l'IA dans son processus de recrutement [27]. L'apprentissage ayant été effectué à partie des CV des personnels d'Amazon, majoritairement des hommes, le système pénalisa les femmes. Même en anonymisant le sexe, le biais persistait et s'exprimait dans les détails : tournures de phrases, expressions, etc. Discrédité et faute de pouvoir être corrigé, le système fut abandonné.

Les biais des données est un problème complexe, car ils ne sont pas toujours visibles. Des corpus d'images issues de Facebook ou des critiques de films issues de IMDB comporteront tous les biais de notre société. Sur Facebook, les enfants apparaîtront davantage avec des femmes et sur IMDB, plus de 80 % des votes sont effectués par des hommes…

Beaucoup de travaux sont en cours pour essayer de corriger ces biais ou d'atténuer leurs conséquences. Faut-il rééquilibrer les données en travestissant celles-ci ? Si oui, sur quelles bases, avec quels contrôles ? …et qui serait légitime pour effectuer cela ?

► Éthique et contrôle

Du reste, le problème n'est pas que dans l'existence de biais, tout recruteur humain comporte des biais, mais dans l'absence de recours et le risque qu'un tel outil puisse devenir une norme. Facebook, LinkedIn, Meetic, des états ou des villes, sont en position d'imposer leurs règles, en toute bonne foi (ou non).

Tout système complexe est faillible et comporte des limites et une zone grise dans laquelle résultats justes et faux positifs ou négatifs se mélangent. La question de cette zone grise est centrale. En « délégant » à une machine la « responsabilité » de cette zone grise, nous cherchons à nous affranchir de toute responsabilité ou réflexion.

La mise en place de systèmes permettant de « profiler » des personnes sur une base interprétative est une bonne illustration de cette dérive. Par exemple toujours dans le domaine du recrutement, des sociétés comme HireVue.com ou EasyRecrue.com [28] proposent via l'analyse d'entretiens filmés d'effectuer une sélection « moins arbitraire » et « dénuée de biais », grâce à un « modèle algorithmique de *machine learning* basé sur la psychologie organisationnelle » (!).

Un second exemple est celui du projet européen iBorderCtrl [29], consistant à vouloir automatiser le processus d'entrée d'une personne sur le territoire de l'UE avec une analyse comportementale automatique des entretiens d'entrée. L'objectif annoncé étant de pouvoir identifier automatiquement des terroristes potentiels… Est-il besoin de préciser que le résultat reste pour le moins perfectible sur le plan technique et plus que discutable sur le plan éthique [30]. Un projet similaire, nommé AVATAR[13] a été abandonné aux États-Unis à cause de son manque de précision, qui était pourtant équivalente à des humains.

Le défi n'est pas de disposer d'un système parfait, dénué de tout biais et d'une précision absolue. Aucun système ne le sera jamais. La difficulté réside avant tout dans notre capacité à mettre en place des mécanismes de contrôle, de recours et d'amélioration continue et de garantir une pluralité et une diversité dans les traitements, comme cela est le cas avec « l'intelligence naturelle ». À l'image de la justice, il est possible de disposer d'un référentiel et de règles communes, avec une marge d'interprétation variable d'un juge à l'autre dès lors que la contradiction et le recours sont possibles.

Une large réflexion, un débat et de la transparence sont plus que jamais nécessaires sur le sujet. La mise en œuvre par la Chine (ou la ville de Nice [31]) de ces technologies dans un cadre sécuritaire ne va pas vraiment dans ce sens et devrait nous interroger et nous alerter… On ne peut pas cacher un alligator sous un tapis.

13. Pour *Automated Virtual Agent for Truth Assessments in Real-Time*

Bibliographie

[1] Adrian G. Dyer, Andrew D. Greentree, Jair E. Garcia et Aurore Avarguès-Weber, « Numerical cognition in honeybees enables addition and subtraction », Science Advances, vol. 5, no 2, 1er février 2019, eaav0961 (ISSN 2375-2548, DOI 10.1126/sciadv.aav0961

[2] Gray, J. (2001), from « The Fourth Paradigm: Data-Intensive Scientific Discovery » Tony Hey, Stewart Tansley, Kristin Tolle (2009). Published by Microsoft Research. ISBN: 978-0-9825442-0-4

[3] Dominique Cardon, Jean-Philippe Cointet, Antoine Mazieres. (2018). « La revanche des neurones », Réseaux, La Découverte, 5 (211), <10.3917/res.211.0173>. <hal-01925644>

[4] McCulloch, Warren; Walter Pitts (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". Bulletin of Mathematical Biophysics. 5 (4): 115–133. doi:10.1007/BF02478259

[5] Hebb, D. O. (1949). « The Organization of Behavior: A Neuropsychological Theory. » New York: Wiley and Sons. ISBN 9780471367277.

[6] Rosenblatt, Frank. (1958). « The perceptron: A probabilistic model for information storage and organization in the brain. » Psychological Review, 65(6), 386-408.

[7] James Lighthill (1973): "Artificial Intelligence: A General Survey" in Artificial Intelligence: a paper symposium, Science Research Council

[8] Rumelhart, David E.; Hinton, Geoffrey E.; Williams, Ronald J. (1986). « Learning representations by back-propagating errors ». Nature. 323 (6088): 533–536. doi:10.1038/323533a0.

[9] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, « Backpropagation Applied to Handwritten Zip Code Recognition », AT&T Bell Laboratories

[10] Statistics on top 500 high-performance computers. (2018) « Exponential growth of supercomputing power as recorded by the TOP500 list ». https://www.top500.org

[11] Wikipedia/en. (2018) « List of datasets for machine-learning research ». https://en.wikipedia.org

[12] ImageNet Large Scale Visual Recognition Challenges (ILSVRC) http://image-net.org/challenges/LSVRC, https://en.wikipedia.org/wiki/ImageNet

[13] AlexNet fut conçu par Alex Krizhevsky, thésard, et publié avec Ilya Sutskever et Geoffrey Hinton. A. Krizhevsky, I. Sutskever, G. E. Hinton (2012). « ImageNet Classification with Deep Convolutional Neural Networks » in Advances in Neural Information Processing Systems 25 (NIPS 2012) http://www.cs.toronto.edu/~hinton/absps/imagenet.pdf

[14] Web Of Science : Core database : TS=("support vector machine*" OR ("SVM" AND "classification") OR ("SVM" AND "regression") OR ("SVM" AND "classifier") OR "support vector network*" OR ("SVM" AND "kernel trick*")) et TS=("deep learning" OR "deep neural network*" OR ("DNN" AND "neural network*") OR "convolutional neural network*" OR ("CNN" AND "neural network*") OR "recurrent neural network*" OR ("LSTM" AND "neural network*") OR ("RNN*" AND "neural network*"))

[15] Base de donnée MNIST : http://yann.lecun.com/exdb/mnist/

[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean (2013), « Distributed Representations of Words and Phrases and their Compositionality », https://arxiv.org/abs/1310.4546

[17] Jeffrey Pennington, Richard Socher, Christopher D. Manning (2014) « GloVe: Global Vectors for Word Representation », http://nlp.stanford.edu/projects/glove/

[18] Sepp Hochreiter, Jürgen Schmidhuber, (1997), « Long Short-Term Memory »,Neural Computation https://doi.org/10.1162/neco.1997.9.8.1735

[19] Cho, Kyunghyun; van Merrienboer, Bart; Gulcehre, Caglar; Bahdanau, Dzmitry; Bougares, Fethi; Schwenk, Holger; Bengio, Yoshua (2014), « Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation ». https://arxiv.org/abs/1406.1078

[20] AG Barto, RS Sutton and CW Anderson, "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problem", IEEE Transactions on Systems, Man, and Cybernetics, 1983.

[21] OpenAI est une émanation de l'organisation à but non lucratif OpenAI Inc, dont l'objet est de contribuer à un développement « humaniste » de l'intelligence artificielle. OpenAI propose notamment un certain nombre de simulateurs d'environnement, tel que *Gym*. https://gym.openai.com/

[22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio (2014), « Generative Adversarial Networks » https://arxiv.org/abs/1406.2661

[23] AI Index. « A starting point for informed conversations about progress in artificial intelligence. The report aggregates a diverse set of metrics, and makes the underlying data easily accessible to the general public ». https://aiindex.org

[24] Jeff Hale, « Deep Learning Framework Power Scores 2018 » https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a et https://www.kaggle.com/discdiver/deep-learning-framework-power-scores-2018

[25] Cathy O'Neil, « Algorithmes, la bombe à retardement », Traduit de l'anglais par Sébastien Marty. Édition : Les Arènes

[26] Kate Crawford, « les biais sont devenus le matériel brut de l'IA », https://www.lemonde.fr/blog/internetactu/2019/10/03/kate-crawford-les-biais-sont-devenus-le-materiel-brut-de-lia/

[27] « Amazon scraps secret AI recruiting tool that showed bias against women »,Reuters, 10 octobre 2018, https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G?utm_source=Twitter&utm_medium=Social

[28] https://www.hirevue.com/ et https://www.easyrecrue.com proposent d'*« Accélérer le recrutement des meilleurs candidats en digitalisant votre processus »* en s'appuyant notamment sur l'analyse automatique des vidéos d'entretiens.

[29] iBorderCtrl est un projet de 4,5 M€, financé par la Commission Européenne. *« (…) the project is aiming to deliver more efficient and secure land border crossings to facilitate the work of border guards in spotting illegal immigrants, and so contribute to the prevention of crime and terrorism. »* https://ec.europa.eu/research/infocentre/article_en.cfm?artid=49726 et https://www.iborderctrl.eu/

[30] « Des détecteurs de mensonge expérimentés aux frontières extérieures de l'Europe », Le Monde, 31 août 2019. https://www.lemonde.fr/international/article/2019/08/31/des-detecteurs-de-mensonge-experimentes-aux-frontieres-exterieures-de-l-europe_5504926_3210.html

[31] « Reconnaissance faciale : la CNIL tique sur le bilan de l'expérience niçoise », Le Monde, 29 août 2019, https://www.lemonde.fr/pixels/article/2019/08/28/reconnaissance-faciale-la-cnil-tique-sur-le-bilan-de-l-experience-nicoise_5503769_4408996.html

Notebooks

[NB1] *01-Linear-Regression.ipynb*
Régression linéaire en résolution directe

[NB2] *02-Gradient-descent.ipynb*
Régression linéaire avec descente de gradient

[NB3] *03-Polynomial-Regression.ipynb*
Régression polynomiale avec Numpy – Exemple d'overfitting

[NB4] *08.2-Logistic-Regression.ipynb*
Régression logistique avec descente de gradient

[NB5] *12.1-Activation-Functions.ipynb*
Fonctions d'activation usuelles

[NB6] *13-Simple-Perceptron.ipynb*
Perceptron avec scikit-learn

[NB7] *14.2.1-DNN-MNIST.ipynb*
Classification de chiffres manuscrits (MNIST) avec DNN / Keras

[NB8] *14.2.2-CNN-MNIST.ipynb*
Classification de chiffres manuscrits (MNIST) avec CNN / Keras

[NB9] *15.1-CNN-Krypton.ipynb*
Réseau convolutif (CNN) avec Keras

[NB10] *16.0-MobileNet-Coco.ipynb*
Classification d'image avec Tensorflow.js (MobileNet)
Détection d'objet avec Tensorflow.js (Coco-ssd)

[NB11] *22.2-Embedding-Gensim.ipynb*
Analyse de texte (TripAdvisor) - Embedding/CBOW avec Gensim

[NB12] *22.3-Embedding-Keras.ipynb*
Analyse de texte (IMDB) – Embedding avec Keras

[NB13] *21.3-RNN-Time-Series.ipynb*
Réseau récurent (RNN LSTM) avec Tensorflow

[NB14] *19.5-Reinf-NN-CartPole.ipynb*
Pendule inversé (Gym Cartpole) avec Tensorflow et descente de gradient

Illustrations

Figure 1 :	Wikimedia.org
Figures 2 à 30 :	Auteur
Photographies page 14 :	pixels.com
Iconographie :	thenounproject.com

Version : 1.4
Octobre 2019