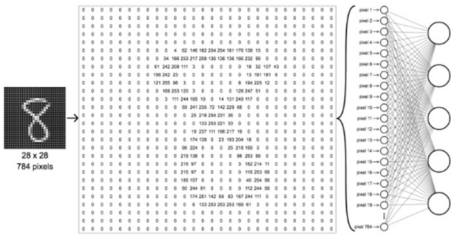
L'apprentissage par réseau de neurones

1. Description générale d'un réseau de neurones

Selon Wikipédia, un réseau de neurones artificielles est un système informatique inspiré du réseau de neurones biologiques qui compose le cerveau animal. Ce n'est pas un algorithme mais plutôt une structure dans laquelle différents algorithmes d'apprentissage machine se mettent en œuvre afin d'effectuer l'analyse données.



Par exemple, un réseau calibré peut prédire ce que représente une image. Dans l'illustration ci-contre, le réseau a été conçu pour reconnaître si une image représente le chiffre « 0 », « 1 », … , « 8 », « 9 ». Ce réseau comporte alors 10 sorties admissibles. Bien que cette tâche semble simple pour un humain, écrire un programme réalisant cette tâche est très difficile.

1.1 Survol des étapes de l'apprentissage supervisé par réseau de neurones

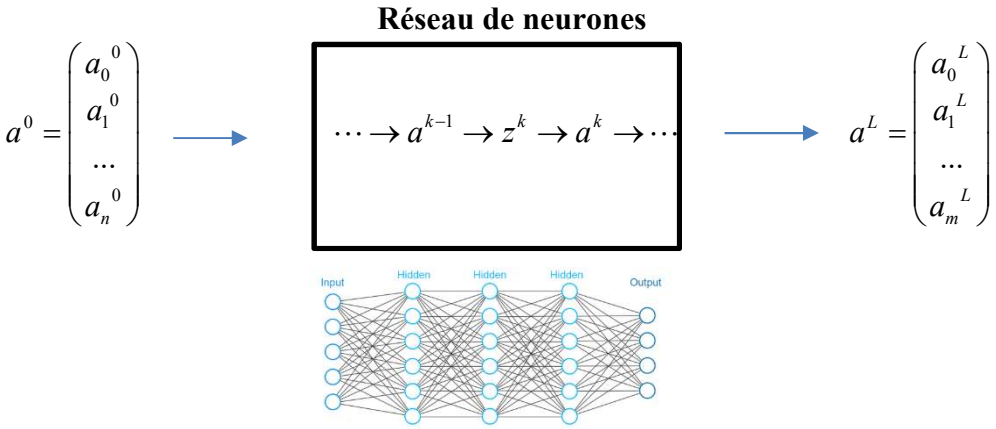
Étape 0

Les données à analyser : a^0 ainsi que le résultat attendu y sont données sous forme de vecteurs.

Données à analyser : $a^0 = \begin{pmatrix} {a_0}^0 \\ {a_1}^0 \\ \dots \\ {a_n}^0 \end{pmatrix}$ Résultats attendus : $y = \begin{pmatrix} y_0 \\ y_1 \\ \dots \\ y_m \end{pmatrix}$

Étape 1 *L'activation du réseau de neurones*

Les données passent à travers le réseau de neurone de L couches. Ce réseau est une fonction relativement simple dont les paramètres sont nombreux. L'image de la fonction est un vecteur de la même dimention que y et si le réseau de neurone est bien calibré, l'image devrait être semblable au résultat attendu.



À l'intérieur du **réseau de neurones**, les opérations suivantes doivent être éxécutées pour chacune des couches.

Étape 1a *L'agrégation d'une couche* $a^{k-1} \to z^k$

Il s'agit simplment de multiplication matricielle et d'addition de vecteurs .

$$z^k = W^k \cdot a^{k-1} + b^k$$

$$z^k = \begin{pmatrix} z_0^{\,k} \\ z_1^{\,k} \\ \dots \\ z_{N^k-1}^{\,k} \end{pmatrix} = \begin{pmatrix} w^k{}_{00} & w^k{}_{01} & & w^k{}_{0,N^{k-1}-1} \\ w^k{}_{10} & & & \\ & & & \\ w^k{}_{N^{k-1}-10} & & & w^1{}_{N^k-1,N^{k-1}-1} \end{pmatrix} \cdot \begin{pmatrix} a_0^{\,k-1} \\ a_1^{\,k-1} \\ \dots \\ a_{N^{k-1}-1}^{\,k-1} \end{pmatrix} + \begin{pmatrix} b_0^{\,k} \\ b_1^{\,k} \\ \dots \\ b_{N^k-1}^{\,k} \end{pmatrix}$$

Les paramètres de notre fonction sont toutes les composantes w^k_{uv} des matrices W^k et les composantes $b_u^{\,k}$ des vecteurs b^k . Au départ ces valeurs sont générées aléatoirement. Ce sont ces paramètres que nous devrons ajuster, calibrer afin que le résultat obtenu soit plus près du résultat attendu.

Remarque : W^k dénote la matrice utilisée pour le passage à la couche k , et non la matrice W élevée à la puissance k

Étape 1b *L'activation d'un couche* $z^k \to a^k$

Les composantes des vecteurs z^k obtenus par ces opérations matricielles doivent être ajustés afin d'être comprises dans l'intervalle $[0\ ,\ 1]$ ou $[-1\ ,\ 1]$.

Pour ce faire nous utilisons une fonction dite d'**activation** $f : \mathbb{R} \to [a\ ,\ b]$.

$$a^k = \begin{pmatrix} {a_0}^k \\ {a_1}^k \\ \dots \\ {z_{N^k-1}}^k \end{pmatrix} = f \begin{pmatrix} {z_0}^k \\ {z_1}^k \\ \dots \\ {z_{N^k-1}}^k \end{pmatrix} = \begin{pmatrix} f({z_0}^k) \\ f({z_1}^k) \\ \dots \\ f({z_{N^k-1}}^k) \end{pmatrix}$$

Les étapes **1a** et **1b** sont répétées pour toutes les L couches de notre réseau de neurones.

Après avoir activé le réseau de neurones, il faut ensuite le **corriger,** le calibrer.

Étape 2 *Le calcul de l'erreur*

Le résultat obtenu a^L doit être comparé avec le résultat attendu y via la fonction d'erreur C.

$$C = \frac{1}{2} \sum_{u=0}^{N^L} \left({a_u}^L - y_u \right)^2$$

Où $C = C({w^k}_{uv}, {b_u}^k)$.

Étape 3 *La calibration du réseau de neurones.* (à voir durant la partie 2)

Nous minimiserons C, à l'aide de la méthode nommée : *descente stochastique du gradient* qui permet de calibrer les paramêtres ${w^k}_{uv}$ et ${b_u}^k$.

Étape 4 *Et on recommence!!!*

On recommence les étapes **1, 2** et **3** jusqu'à l'atteinte d'un réseau suffisemment calibré.

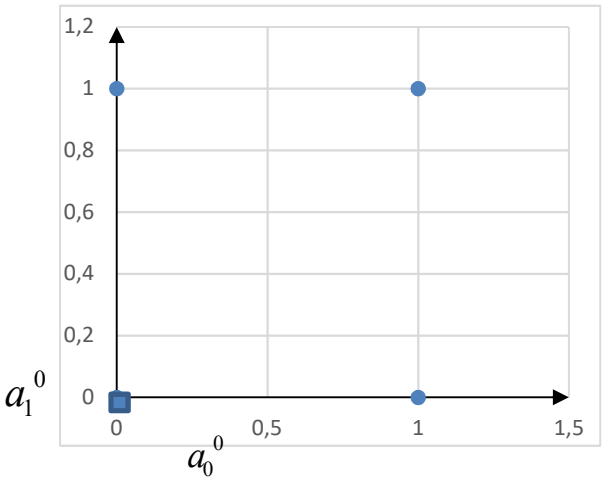
2 Des exemples classiques

2.1 Un exemple simple

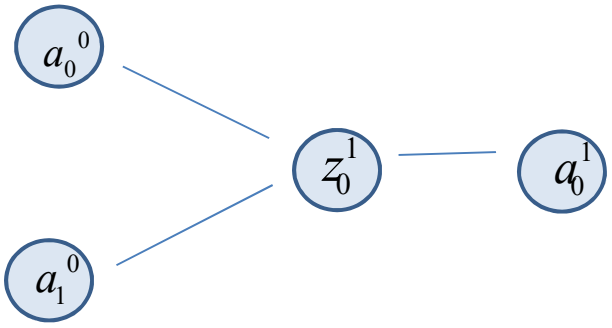
Considérons le connecteur logique **OU,** dont la sortie est **VRAI** lorsque qu'au moins une des deux entrées est **VRAI** . Rappel, nous utilisons 0 pour **FAUX** et 1 pour **VRAI.**

Table de vérité du connecteur logique **OU**

Données à analyser		Sortie attendue du réseau
a_0^0	a_1^0	y_0
0	0	0
0	1	1
1	0	1
1	1	1



Les données à analyser sont $a^0 = \begin{pmatrix} a_0^0 \\ a_1^0 \end{pmatrix}$ et le resultat attendu n'a qu'une composante $y = (y_0)$.



Étape 1. Activation du réseau.

Étape 1a. L'agrégation de la couche

$$z^1 = W^1 \cdot a^0 + b^1$$

$$z^1 = \begin{pmatrix} a & b \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + (c) = ax + by + c$$

Étape 1b l'activation de la couche

Nous utiliserons la fonction sigmoïde σ

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

$$a^1 = \left(a_0^{\,1}\right) = \left(\sigma(z_0^{\,1})\right) = \left(\frac{1}{1+e^{-z_0^{\,1}}}\right) = \left(\frac{1}{1+e^{-(ax+by+c)}}\right)$$

Comme règle de décision, cette valeur sera arrondie. Ainsi, si $a_0^{\,1} < 0,5$, le résultat obtenu sera traité comme un 0 **(FAUX)**, sinon le résultat sera traité comme un 1 **(VRAI)**.

La frontière entre les deux zones est donnée par $0,5 = a_1 = \left(\frac{1}{1+e^{-(ax+by+c)}}\right)$. Une fois simplifiée, cette équation est la droite $0 = ax + by + c$.

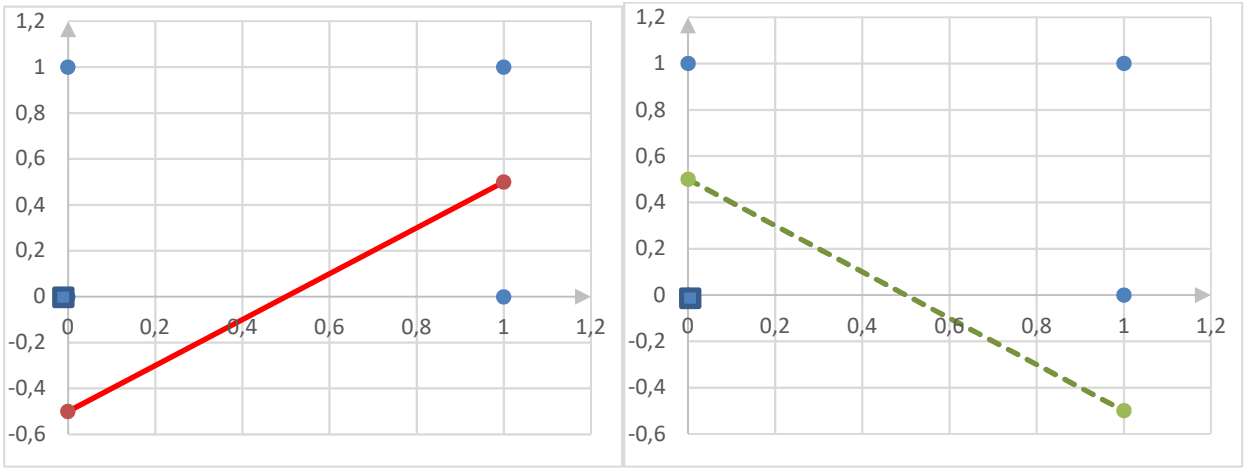
Initialement, les 3 paramètres a, b, et c sont générés aléatoirement. Si le réseau n'est pas entraîné, la droite sera quelconque la prédiction sera médiocre. Après entraînement, ce réseau simple est efficace à 100%. Par exemple :

Exemple	**a**	**b**	**c**	**Taux de succès**
#1 Trait plein Réseau non entraîné	1	-1	1/2	50%
#2 Trait pointillé Réseau entraîné	1	1	-1/2	100%

L'exemple **#1**, représente un réseau non calibré qui se trompe deux fois sur quatre. Les points (1,1) (0,1) et (0,0) sont dans la zone **VRAI** et le point (1,0) est dans la zone **FAUX.**

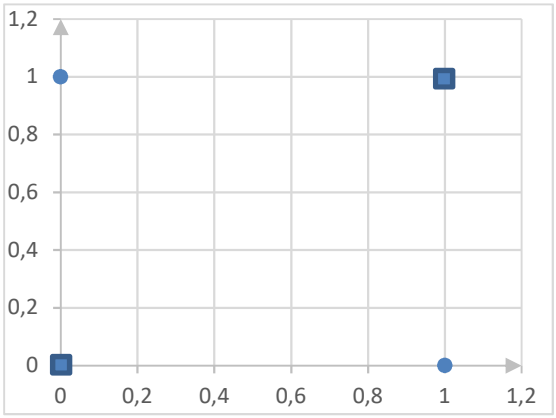
L'exemple **#2** représente un réseau qui serait calibré, la droite peut trancher, sans erreur, la zone où la sortie est **FAUX** de la zone où la sortie est **VRAI.**

Exemple #1 Exemple #2



Évidemment, une seule droite est rarement suffisante pour trancher entre deux séries de données. Le seul exemple du **OU exclusif** nous permet de le constater. Le connecteur logique **OU exclusif** est **VRAI** lorsque que seulement une des deux entrées est **VRAI** .

Représentation du connecteur logique **OU exlusif.**



2.2 Le MNIST Data Set

Les entrées : vecteurs à 784 composantes dont la valeur correspond à une nuance de gris entre 0 et 1

Les résultats attendus : vecteurs à 10 composantes.

Par exemple, $y = (0,0,1,0,0,0,0,0,0,0)$ sera la sortie attendue pour une image qui ressemble au chiffre 2.

Taille du réseau **784 x 16 x 16 x 10.**

- 3 couches,
- 1 couche de sortie ayant 10 neureones,
- 2 couches cachées ayant toutes deux 16 neurones.
- La fonction sigmoïde est utilisée pour l'activation des 3 couches du réseau

https://en.wikipedia.org/wiki/File:MnistExamples.png
Échantillons de la base de données MNIST.

3. Les fonctions utilisées dans un réseau de neurones

3.1 La fonction d'agrégation *z*

Cette fonction amorce le passage de la couche $k-1$ à la la couche k.

$$\cdots \rightarrow \boxed{a^{k-1} \rightarrow z^k} \rightarrow a^k \rightarrow \cdots$$

Les paramètres de cette fonction sont les poids ${w^k}_{uv}$ et les biais ${b^k}_u$. Au départ ces valeurs sont générées aléatoirement.

$$z^k = {W^k}_{(N^k-1)\times(N^{k-1}-1)} \cdot a^{k-1} + b^k$$

$$z^k = \begin{pmatrix} {z_0}^k \\ {z_1}^k \\ \cdots \\ {z_{N^k-1}}^k \end{pmatrix} = \begin{pmatrix} {w^k}_{00} & {w^k}_{01} & & {w^k}_{0,N^{k-1}-1} \\ {w^k}_{10} & & & \\ & & & \\ {w^k}_{N^{k-1}-10} & & & {w^1}_{N^k-1,N^{k-1}-1} \end{pmatrix} \cdot \begin{pmatrix} {a_0}^{k-1} \\ {a_1}^{k-1} \\ \cdots \\ {a_{N^{k-1}-1}}^{k-1} \end{pmatrix} + \begin{pmatrix} {b_0}^k \\ {b_1}^k \\ \cdots \\ {b_{N^k-1}}^k \end{pmatrix}$$

Notons que N^k est le nombre de neurones dans la couche k et que les indices débutent à 0

3.2 La fonction d'activation *A*

Les composantes des vecteurs z^k doivent être ajustés afin d'être comprises dans l'intervalle $[0\ ,\ 1]$ ou $[\text{-}1\ ,\ 1]$à l'aide d'une fonction d'**activation**.

$$\cdots \rightarrow a^{k-1} \rightarrow \boxed{z^k \rightarrow a^k} \rightarrow \cdots$$

La fonction sera appliquée à toutes les composantes du vecteur z^k. Les fonctions que nous utiliserons seront :

La fonction sigmoïde σ :

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

La fonction tangente hyperbolique :

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

3.3 L'activation d'un réseau

L'activation d'un réseau consiste à introduire un vecteur d'entrée a^0 (*input*) dans le réseau, à utiliser les fonctions d'agrégation et d'activation pour chacune des L couches et ainsi obtenir a^L, **l'activation du réseau.**

$$a^0 \to z^1 \to a^1 \to \cdots \to a^{k-1} \to z^k \to a^k \to \cdots \to z^L \to a^L$$

3.5 La fonction d'erreur *C*

La fonction d'erreur C permet d'évaluer à quel point il y a une différence entre l'activation d'un réseau a^L calculé à partir d'un vecteur d'entrée a^0 et le résultat attendu y. C'est à partir de la fonction d'erreur qu'il est possible de déterminer si la prédiction d'un réseau est bonne ou non. C'est également à partir de cette fonction que nous pourrons modifier le réseau afin que celui-ci s'améliore par un processus d'apprentissage et réduire l'erreur calculée.

La fonction que vous implémenterez dans ce laboratoire est la fonction d'erreur quadratique, qui porte également le nom de fonction d'énergie (*energy*).

$$C = \frac{1}{2}\sum_{u=0}^{N^L-1}\left(a_u{}^L - y_u\right)^2$$

Le réseau de neurone en quelques lignes

On active le réseau

$$a^0 \to z^1 \to a^1 \to \cdots \to a^{k-1} \to z^k \to a^k \to \cdots \to z^L \to a^L$$

$$z^k = W^k \cdot a^{k-1} + b^k \qquad \text{et} \qquad a^k = f\left(z^k\right)$$

On propage l'erreur…*la suite*

$$\Delta^0 \leftarrow \cdots \leftarrow \Delta^k \leftarrow \cdots \leftarrow \Delta^{L-1} \leftarrow \Delta^L$$

$$\Delta_u^L = (a_u^L - y_u)\sigma'(z_u^L) \quad \text{et} \quad \Delta_u^{k-1} = \sum_{v=0}^{N^k-1} \Delta_v^k \cdot w^k{}_{vu} \cdot \sigma'(z_u{}^{k-1})$$

On calibre le réseau

$$\tilde{w}_{uv}^k = w^k{}_{uv} - \alpha\ \Delta_u^k a_v{}^{k-1} \text{ et} \qquad \tilde{b}_u^k = b^k{}_u - \alpha\ \Delta_u^k,$$