جامعة 8 ماي 1945 قالمة
UNIVERSITE 8 MAI 1945 - GUELMA

LabSTIC
Université 8 Mai 1945

Les approches cellulaires

Dr. Mohammed Nadjib KOUAHLA

Plan

- **Historique et développements**
- **Domaines d'application**
- **Le système neuronal artificiel:**

  Inspiration → modélisation
- **Types d'apprentissage**
- **Réseaux de neurones artificiels**
- **Les différents architectures**

Historique et développement

Approches informatiques pour résoudre un problème :

- Approche algorithmique (programmation complète)
- Création des « moteurs d'inférence » (programme qui raisonne ; règles SI..ALORS.. ; système expert)
- Approche connexionniste : le réseau s'organise par apprentissage (pas de programmation)

Historique et développement

Caractéristiques de l'approche connexionniste (réseaux neuronaux) :

Calcul non-algorithmique

Information et mémoire distribuée dans le réseau

Architecture massivement parallèle (processeurs élémentaires interconnectés)

Apprentissage par entraînement sur des exemples

Inspiré du fonctionnement du cerveau

Historique et développement

1943 J.Mc Culloch et W.Pitts établissent le "modèle logique" du neurone qui ouvre la voie à des modèles techniques.

1949 D.Hebb élabore une théorie formelle de l'apprentissage biologique par modifications des connexions neuronales.

1957 F.Rosenblatt réalise le Perceptron, le premier modèle technique basé sur la modification des poids.

1960 B.Widrow réalise Adaline (Adaptive Linear Element), un réseau adaptatif de type perceptron.

1969 M.Minsky et S.Papert émettent des critiques et démontrent les limites des modèles neuronaux de type perceptron.

La recherche s'arrête durant un peu plus d'une dizaine d'années.

1982 J.Hopfield (physicien) propose une nouvelle approche des réseaux neuronaux basée sur l'analogie avec les milieux à grand nombre de particules. Cela relance l'intérêt pour les réseaux neuronaux

depuis 1984 : développement croissant du domaine connexionniste aussi bien en IA qu'en informatique.

Domaines d'application

- Traitement des images
- Identification des signatures
- Reconnaissance des caractères (dactylos ou manuscrits)
- Reconnaissance de la parole
- Reconnaissance de signaux acoustiques (bruits sous-marins, …)
- Extraction d'un signal du bruit
- Contrôle de systèmes asservis non-linéaires (non modélisables)
- Robotique (apprentissage de tâches)
- Aide à la décision (domaine médical, bancaire, management, …)

Remarque: Dans toutes les applications des réseaux neuronaux les performances sont acquises suite à une étape d'entraînement sur des exemples (apprentissage)

Le système neuronal artificiel

Un système neuronal artificiel autrement dit «réseau neuronal artificiel » (ANN : Artificial neural network) est un modèle mathématique qui tente de simuler la structure et les fonctionnalités des réseaux de neurones biologiques. Il simule différents aspects liés au comportement du cerveau humain, tels que: traitement intelligent de l'information, traitement distribué, haut niveau de parallélisme, apprentissage, adaptation, tolérance élevée aux informations inexactes (ou erronées).

Un ANN se compose d'un groupe interconnecté de neurones artificiels et traite des informations en utilisant une approche connexionniste.

Inspiration biologique

Dans un cerveau, il y a 10^12 neurones avec 10^3 à 10^4 connexions par neurone.

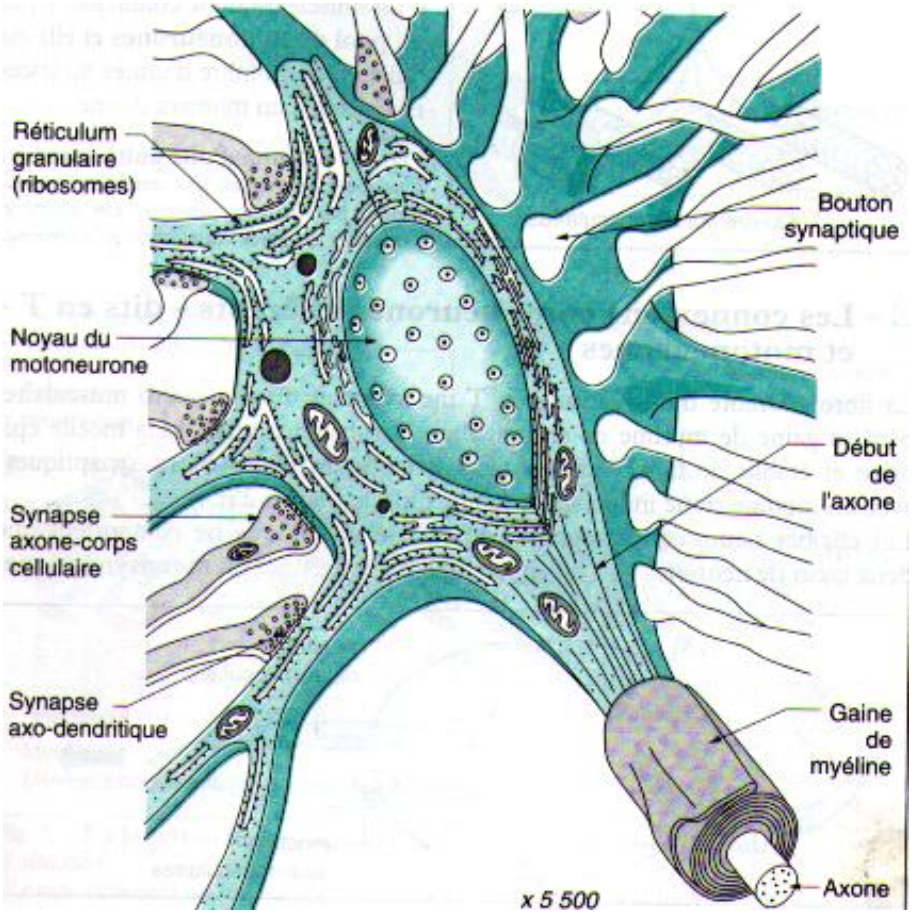
Dendrite : récepteur des messages

Corps : génère le potentiel d'action (la réponse)

Axone : transmet le signal aux cellules suivantes

Synapse : jonction axone - dendrite (plus ou moins passante)

Neurone : élément autonome dépourvu d'intelligence



Modélisation d'un neurone

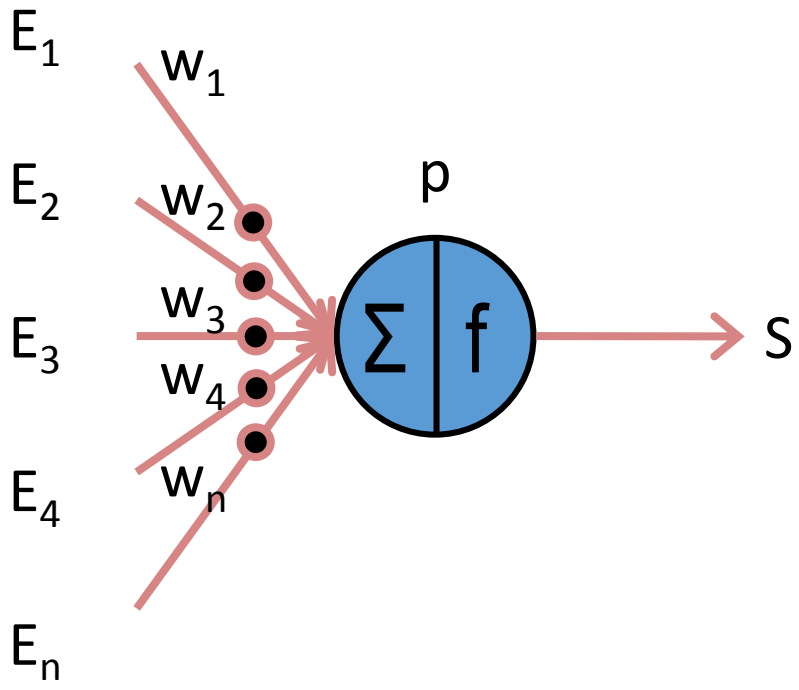
La modélisation du système nerveux biologique repose sur la correspondance suivante:

Système nerveux	Système de calcul neuronal
Neurone	Processeur
Dendrite	Fonction de combinaison
Corps du neurone	Fonction de transfert
Axone	Elément de sortie
Synapse	Poids

Cette correspondance est à la base du « modèle logique » du neurone élaboré en 1946 par McCulloch et Pitts.

Modélisation

La représentation graphique (conventionnelle) d'un neurone formel modélisé par Mc Culloch et Pitts.



Modélisation

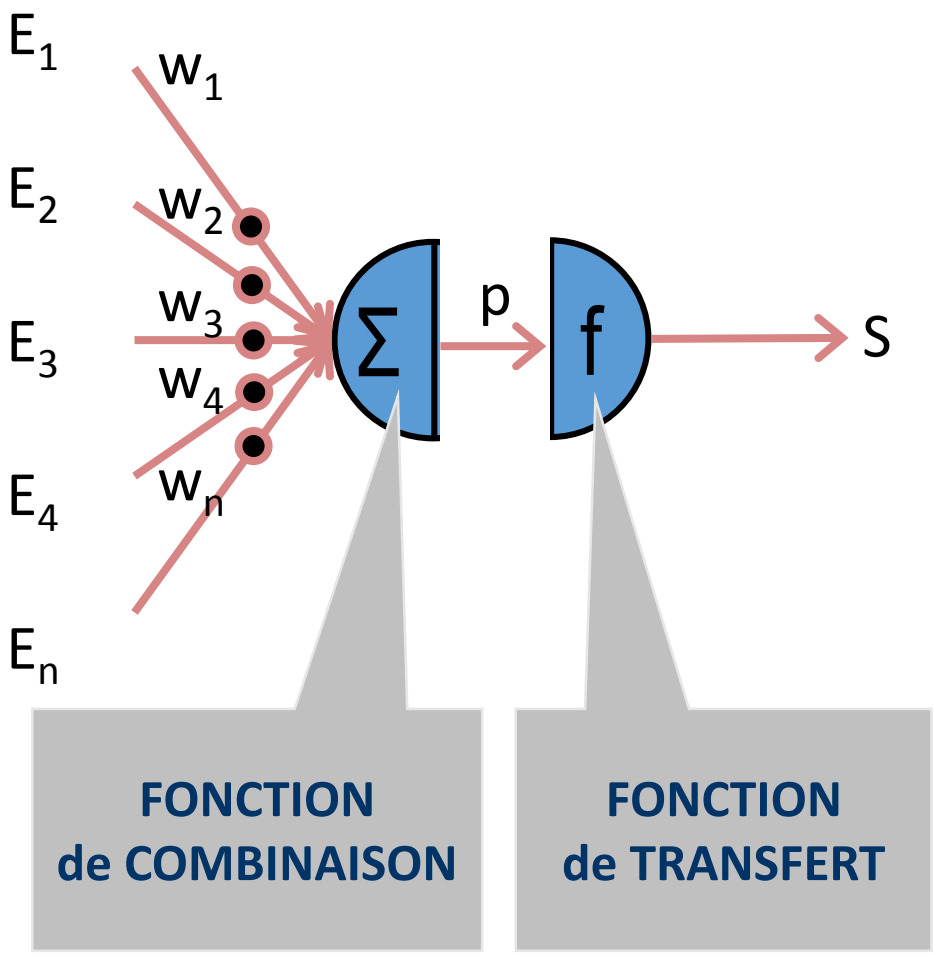
Les éléments constitutifs du neurone artificiel

Les entrées "**E**" du neurone proviennent soit d'autres éléments "processeurs", soit de l'environnement.

Les poids "**W**" déterminent l'influence de chaque entrée.

La fonction de combinaison "**p**" combine les entrées et les poids.

La fonction de transfert calcule la sortie "**S**" du neurone en fonction de la combinaison en entrée.



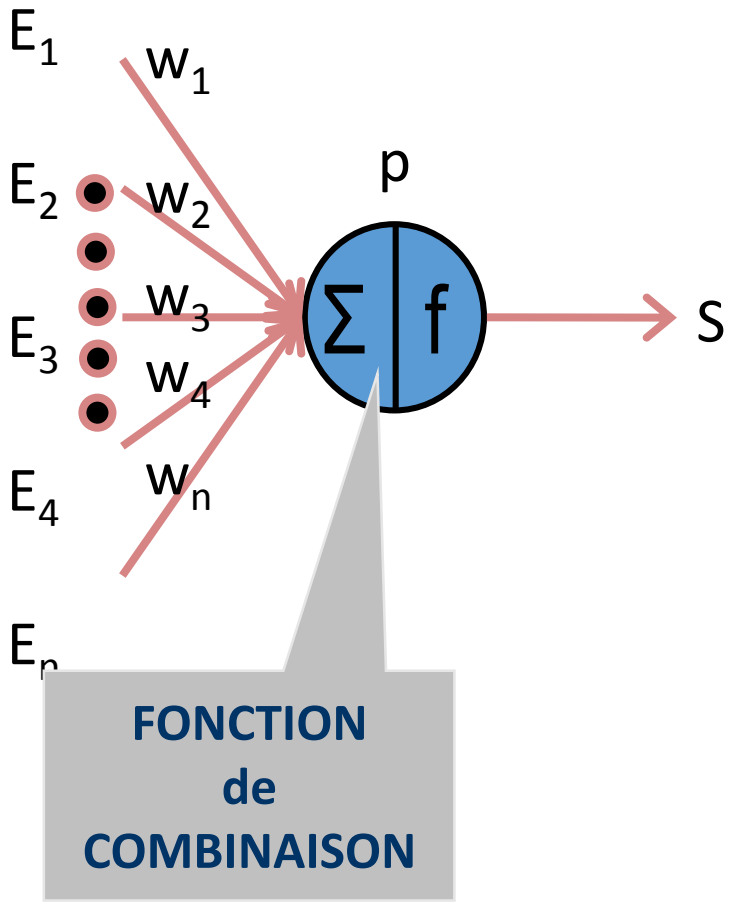
Modélisation

La Fonction de Combinaison calcule l'influence de chaque entrée en tenant compte de son poids. Elle fait la somme des entrées pondérées :

$$p = \sum W_i E_i$$

W_i : Poids de la connexion à l'entrée i.

E_i : Signal de l'entrée i.



Modélisation

La Fonction de Transfert détermine l'état du neurone (en sortie)

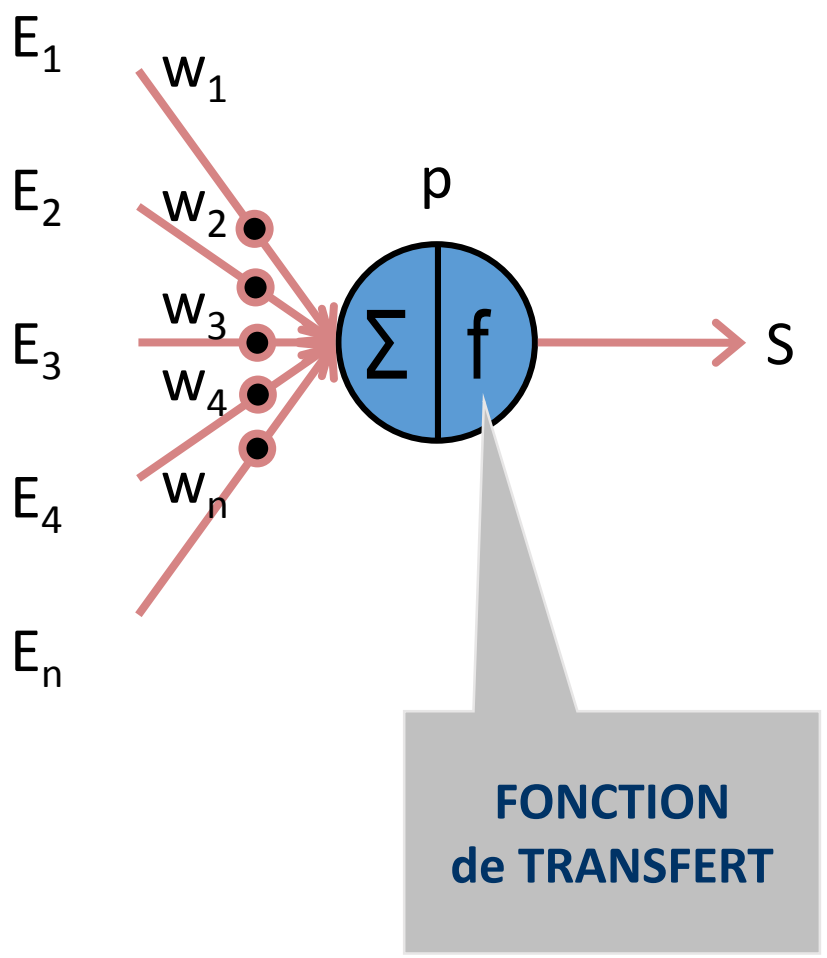
Calcul de la sortie :

$$S = f(p)$$

ou encore :

$$S = f(\sum W_i E_i)$$

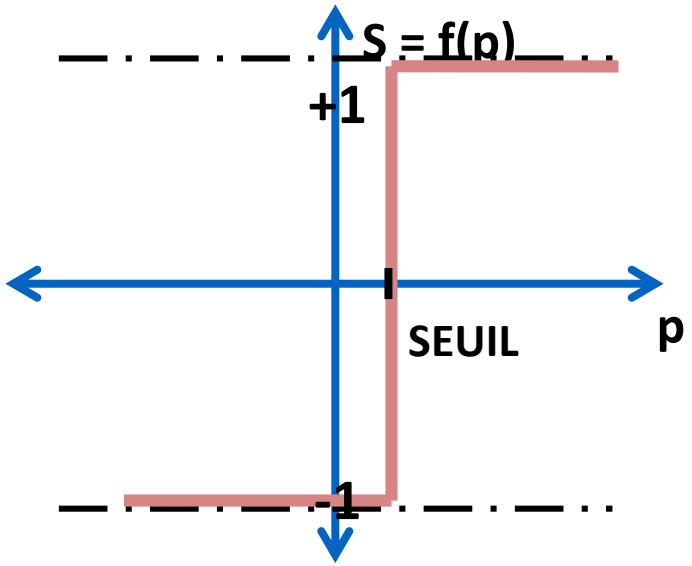
La fonction de transfert "**f**" peut avoir plusieurs formes.



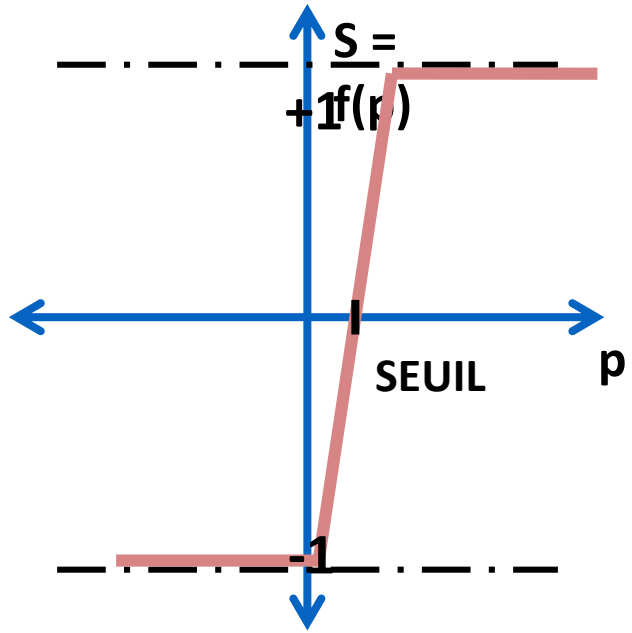
La fonction de transfert 'f' a été modélisée sous plusieurs formes pour plus de ressemblance avec le neurone réel

Modélisation

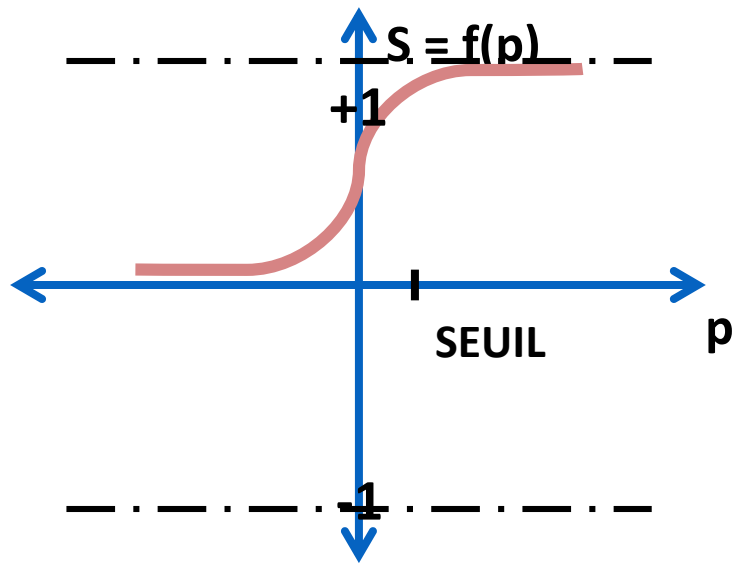
La fonction 'f' peut être de la forme :



Fonction de transfert en échelon : La sortie peut avoir deux états distincts (-1 et +1) selon le dépassement d'un Seuil

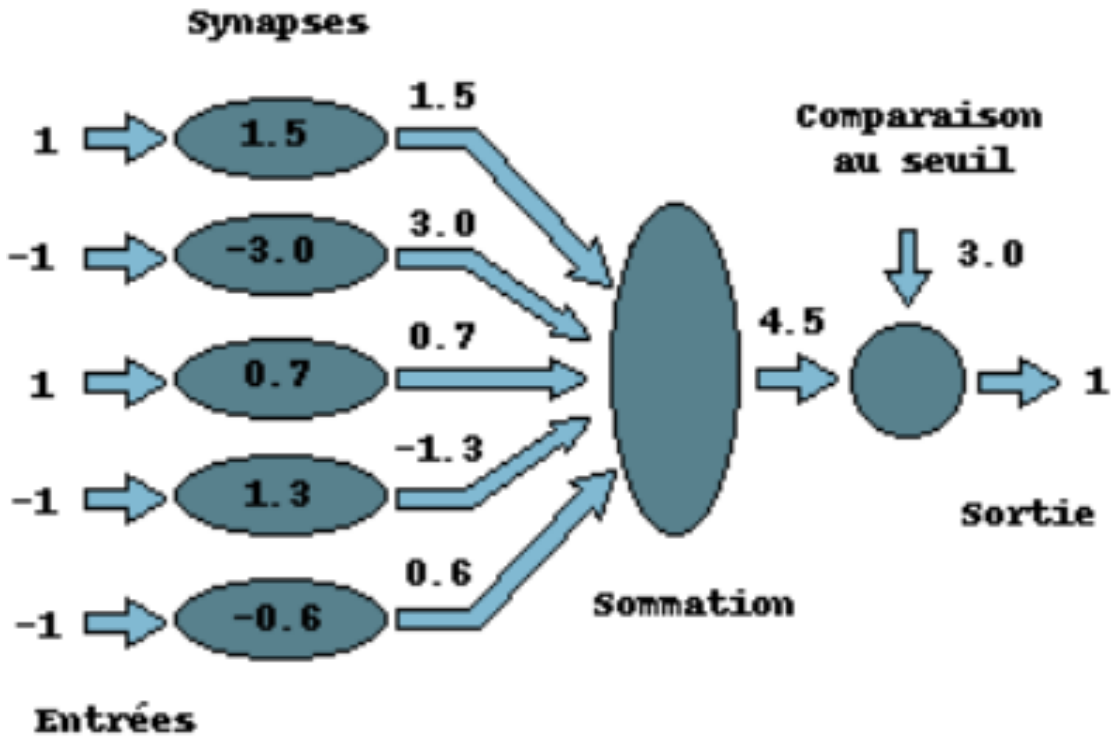


Fonction de transfert linéaire par morceaux : Le passage d'un état à l'autre se fait progressivement



Fonction de transfert dérivable :
Le passage entre les deux états est « continu » donc « dérivable »

Exemple



→ Pas de notions temporelles

Définitions

Le but des réseaux neuronaux est d'apprendre à répondre correctement à différentes entrées.

<u>Moyen :</u> modification des poids par apprentissage : les caractéristiques du réseau sont modifiées jusqu'à ce que le comportement désiré soit obtenu.

- ***Base d'apprentissage*** : exemples représentatifs du comportement ou de le fonction à modéliser. Ces exemples sont sous la forme de couples (entrée ; sortie) connus.

- ***Base d'essai*** : pour une entrée quelconque (bruitée ou incomplète), calculer la sortie. On peut alors évaluer la performance du réseau.

Définitions

• ***Apprentissage supervisé*** : les coefficients synaptiques sont évalués en minimisant l 'erreur (entre sortie souhaitée et sortie obtenue) sur une base d 'apprentissage.

• ***Apprentissage non-supervisé*** : on ne dispose pas de base d 'apprentissage. Les coefficients synaptiques sont déterminés par rapport à des critères de conformité : spécifications générales.

• ***Sur-apprentissage*** : on minimise l 'erreur sur la base d 'apprentissage à chaque itération mais on augmente l 'erreur sur la base d 'essai.
Le modèle perd sa capacité de généralisation : c 'est l ***'apprentissage par cœur***.
⇒ Explication : trop de variables explicatives dans le modèle ; on n 'explique plus le comportement global mais les résidus.

Apprentissage supervisé: caractéristiques

Association imposée entre un vecteur d'entrée (forme multidimensionnelle) et un vecteur de sortie (la réponse désirée).

L'erreur est calculée à chaque essai afin de corriger les poids.

Les poids sont modifiés jusqu'à l'erreur minimale, voire aucune erreur.

Apprentissage non-supervisé: caractéristiques

Pas d'indication sur les erreurs.

Le réseau détecte les caractéristiques communes des différentes entrées.

Il tente ainsi de former des « classes » de façon autonome.

Il apprend à donner des réponses aux classes.

→ **Ayant formé des classes, le système établit la réponse adéquate pour chaque type de classe.**

C'est un apprentissage beaucoup plus difficile à réaliser.

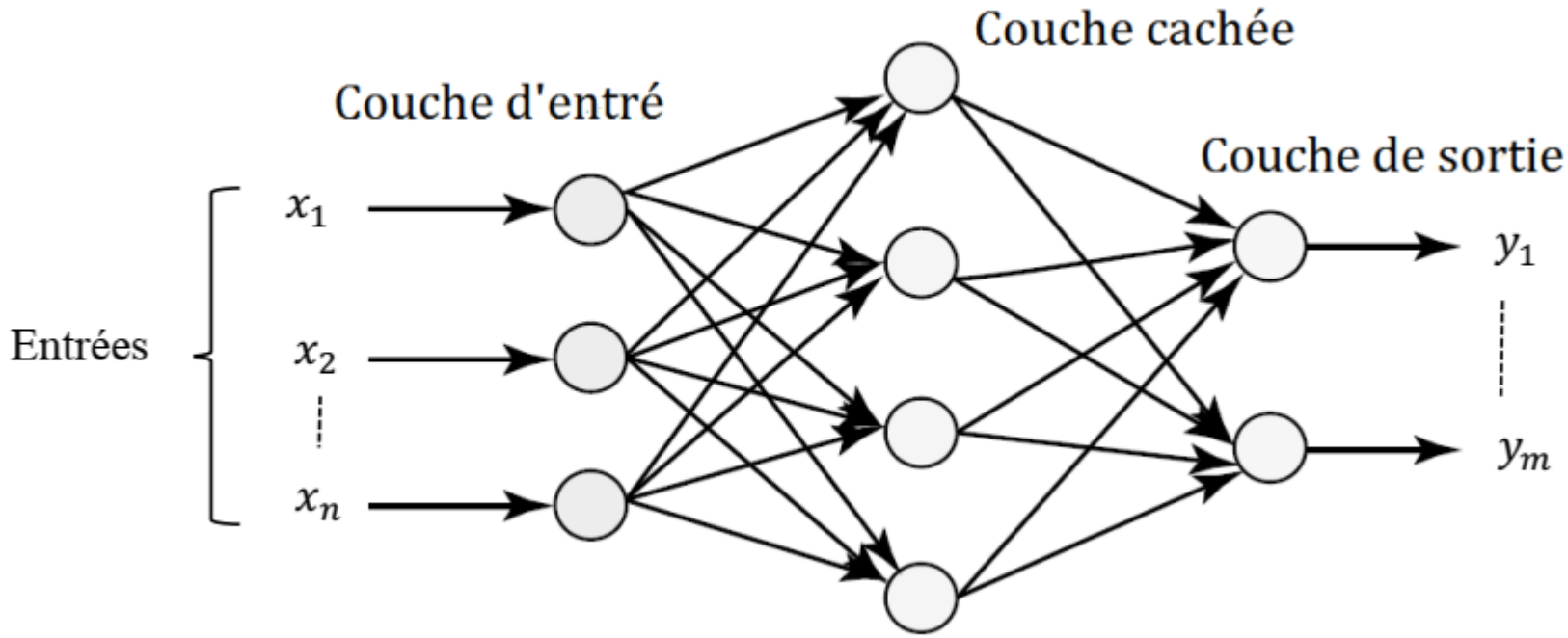
Apprentissage: règles

Règles d'apprentissage du réseau neuronal artificiel:
La règle de Hebb
La règle de Perceptron :

Module : Apprentissage automatique

Réseau de neurones: architecture

En général, l'architecture de base se compose de trois types de couches neuronales: les couches d'entrée, les couches cachées et les couches de sortie :



Réseau de neurones: architecture

- **La couche d'entrée** : Cette couche est chargée de recevoir des informations (données), des signaux, des caractéristiques ou des mesures de l'environnement externe. Ces entrées (échantillons ou modèles) sont normalisées dans les valeurs limites produites par les fonctions d'activation. Cette normalisation se traduit par une meilleure précision numérique pour les opérations mathématiques effectuées par le réseau.

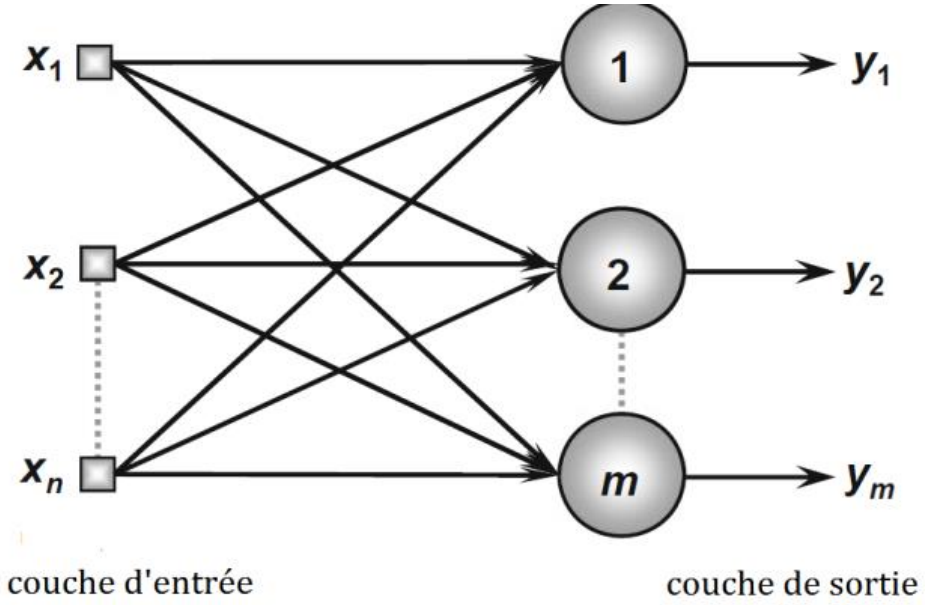
- **La couche cachée** : Cette couche est composée de neurones qui sont responsables de l'extraction des motifs associés au processus ou au système analysé. La couche cachée effectue la majeure partie du traitement interne.

- **La couche de sortie** : Cette couche est responsable de la production et de la présentation des sorties de réseau finales, qui résultent du traitement effectué par les neurones dans les couches précédentes.

Les architectures principales des réseaux neuronaux artificiels, en considérant la disposition des neurones, ainsi que leur interconnexion et la composition de ses couches, peuvent être réparties comme suit: réseau monocouche, réseau multicouche, réseau récurrent.

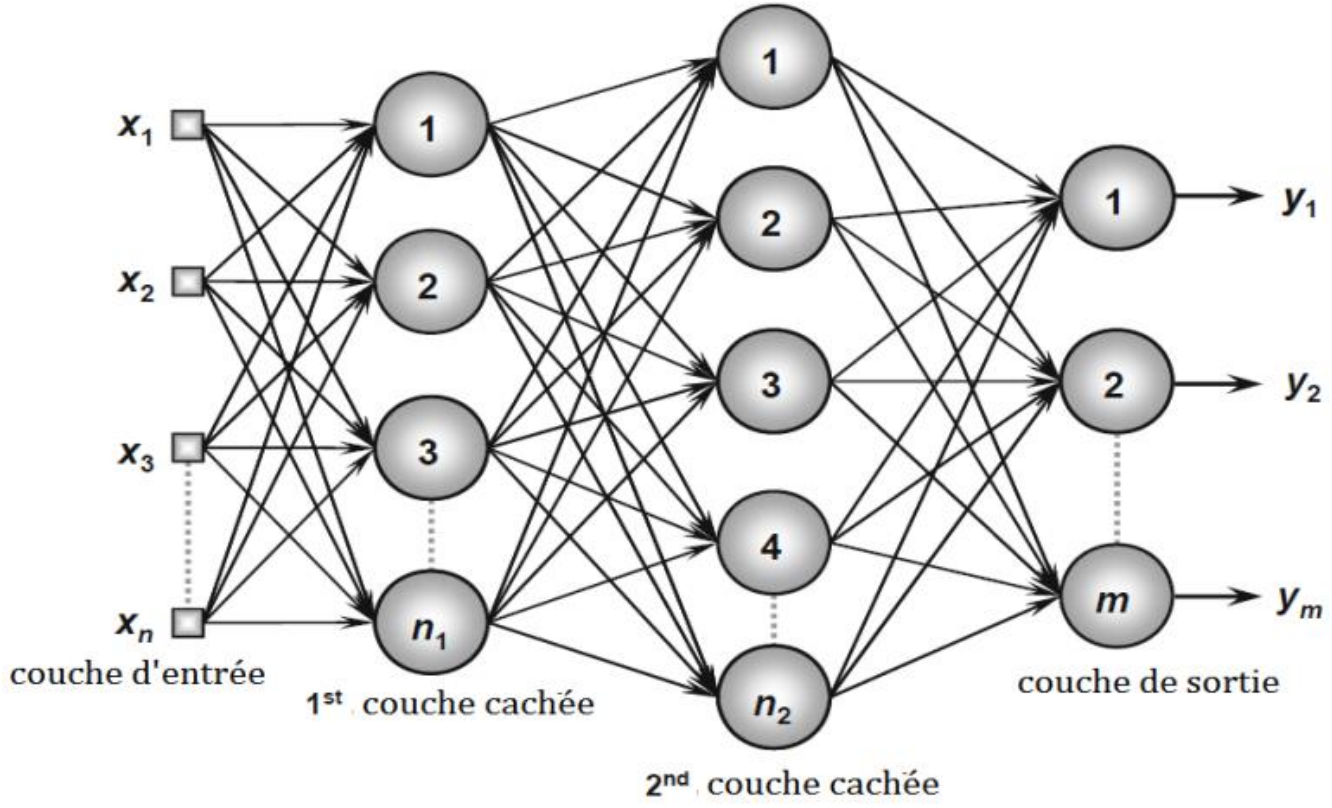
Réseau monocouche

Ce réseau neuronal artificiel a juste une couche d'entrée et une seule couche neuronale, qui est aussi la couche de sortie.
L'information circule toujours dans une seule direction (unidirectionnelle), qui va de la couche d'entrée à la couche de sortie. Dans la figure, on peut voir que dans les réseaux appartenant à cette architecture, le nombre de sorties de réseau coïncide toujours avec sa quantité de neurones.
Ces réseaux sont habituellement employés dans la classification des modèles et les problèmes de filtrage linéaire.



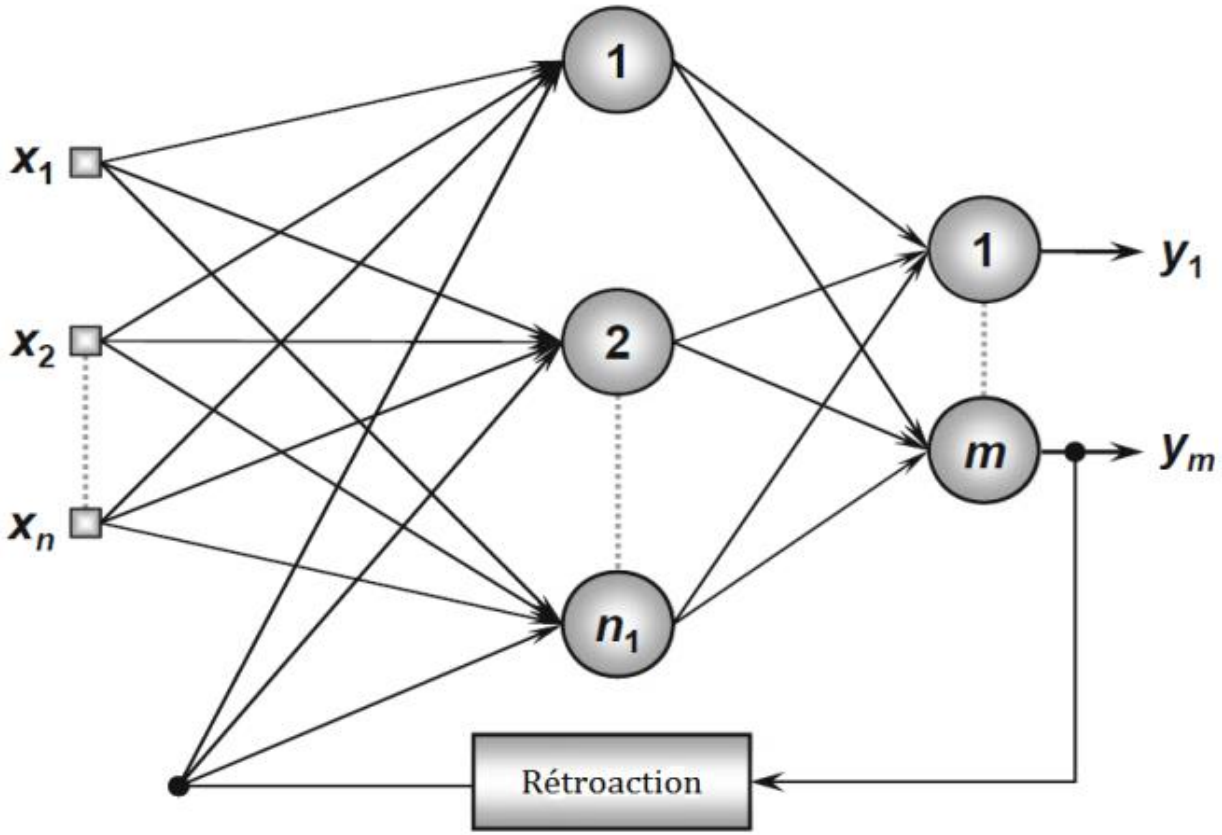
le réseau multicouche

les réseaux multicouches sont composés d'une ou plusieurs couches neuronales cachées. La figure montre un réseau multicouche composé d'une couche d'entrée avec signaux, deux couches neuronales cachées constituées de n1 et n2 neurones respectivement, et enfin une couche neuronale de sortie composée de *m* neurones représentant les valeurs de sortie.



Réseau de neurones récurrent

Dans ce réseau, les sorties des neurones sont utilisées comme entrées de rétroaction pour d'autres neurones. La caractéristique de rétroaction qualifie ces réseaux pour le traitement dynamique de l'information, ce qui signifie qu'ils peuvent être utilisés sur des systèmes variant dans le temps, tels que la prévision des séries chronologiques, l'identification et l'optimisation du système, le contrôle du processus, etc. La figure illustre un exemple d'un réseau récurrent, où l'un de ses signaux de sortie est renvoyé à la couche intermédiaire. Ainsi, en utilisant le processus de rétroaction, les réseaux avec cette architecture produisent des sorties en tenant compte également des valeurs de sortie précédentes.



Deep learning: l'apprentissage en profondeur

Le Deep Learning est un nouveau domaine de recherche de la machine Learning (ML), qui a été introduit dans le but de rapprocher le ML de son objectif principal à savoir : l'intelligence artificielle. Il concerne les algorithmes inspirés par la structure et du fonctionnement du cerveau.

Ils peuvent apprendre plusieurs niveaux de représentation dans le but de modéliser des relations complexes entre les données.

L'apprentissage en profondeur : définition

L'apprentissage en profondeur est un ensemble d'algorithmes d'apprentissage automatique qui tentent d'apprendre à plusieurs niveaux, correspondant à différents niveaux d'abstraction. Il a la capacité d'extraire des caractéristiques à partir des données brutes grâce aux multiples couches de traitement composé de multiples transformations linéaires et non linéaires et apprendre sur ces caractéristiques petites à petit à travers chaque couche avec une intervention humaine minime.

L'apprentissage en profondeur : historique

L'idée du Deep Learning n'est pas une idée récente, mais elle date en réalité des années 1980, plus particulièrement suite aux travaux de réseaux de neurones multicouches et aux travaux de certains pionniers du machine Learning et du Deep Learning comme le français Yann Le Cun. En collaboration avec deux autres informaticiens, Kunihiko Fukushima et Geoffrey Hinton, ils mettent au point un type d'algorithme particulier appelé Convolutional neural network.

Bien que cette approche donne des résultats, ses progrès et son évolution sont limités par les progrès technologiques en matière de micro-processeurs, de puissance de calculs, et du manque d'accessibilités à des données afin de pouvoir entraîner les neurones. Cependant certains chercheurs ont continué à travailler sur ce modèle pendant environ deux décennies et, avec l'aide des évolutions en matière de technologies, mais surtout avec la disponibilité toujours plus grande de données, ont pu améliorer cette technique.
Afin de développer un système d'apprentissage performant, il faut pouvoir l'exercer et cela requiert un nombre important de données à tester. C'est dans ce contexte qu'en 2007 le STANFORD VISION LAB, avec Fei-Fei Li à sa tête, développent un agrégateur d'images où sont consignés et étiquetés quelques millions de photos : ImageNet. En 2010, ImageNet regroupe 15 000 000 d'images toutes catégorisées en fonction de leurs caractéristiques propres (véhicules, animaux, ...).

L'apprentissage en profondeur : historique

Afin de développer un système d'apprentissage performant, il faut pouvoir l'exercer et cela requiert un nombre important de données à tester. C'est dans ce contexte qu'en 2007 le STANFORD VISION LAB, avec Fei-Fei Li à sa tête, développent un agrégateur d'images où sont consignés et étiquetés quelques millions de photos : ImageNet. En 2010, ImageNet regroupe 15 000 000 d'images toutes catégorisées en fonction de leurs caractéristiques propres (véhicules, animaux, ...)

En 2012, le Deep Learning est remis au goût du jour avec un succès retentissant au ImageNet Large Scale Visual Recognition Challenge (ILSVRC) qui est un concours annuel de reconnaissance d'image fondée par l'université de Stanford, dans le cadre de son laboratoire STANFORD VISION LAB. Plusieurs équipes de chercheurs en informatique s'affrontent dans ce concours tous les ans afin de décerner la victoire au programme ayant eu le plus faible taux d'échec. Et alors que les algorithmes d'apprentissage profond sont absents de la compétition, en 2012 c'est bel et bien un algorithme de Deep Learning qui va remporter l'édition 2012 à la surprise générale,

L'apprentissage en profondeur : Pourquoi?

Tout d'abord les différents algorithmes du deep Learning ne sont apparus qu'à l'échec de l'apprentissage automatique tentant de résoudre une grande variété de problèmes de l'intelligence artificielle (l'IA) :

- Afin d'améliorer le développement des algorithmes traditionnels dans de telles tâches de l'IA.
- De développer une grande quantité de données telle que les big data.
- De s'adapter à n'importe quel type de problème
- D'extraire les caractéristiques de façon automatique

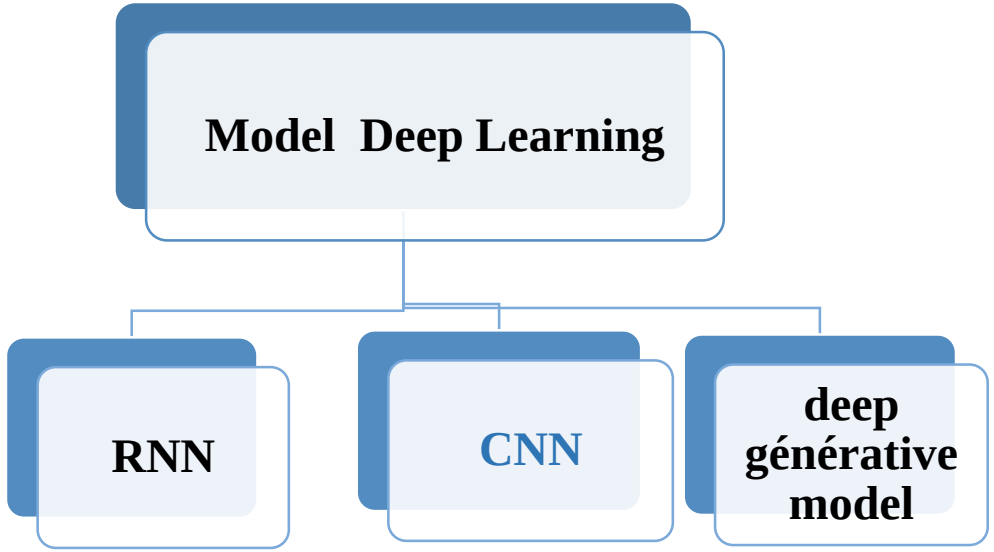
Flux d'apprentissage automatique

Flux d'apprentissage profond

L'apprentissage en profondeur : les architectures

Bien qu'il existe un grand nombre de variantes d'architectures profondes. Il n'est pas toujours possible de comparer les performances de toutes les architectures, car elles ne sont pas toutes évaluées sur les mêmes ensembles de données. Le Deep Learning est un domaine à croissance rapide, et de nouvelles architectures, variantes ou algorithmes apparaissent toutes les semaines.

- **Réseaux de neurones convolutifs**
- **Réseau de neurones récurrents**
- **Réseaux antagonistes génératifs**



Réseau de neurones convolutif

Un réseau de neurones convolutifs ou réseau de neurones à convolution (en anglais CNN ou ConvNet pour Convolutional Neural Networks) est un réseau neuronal multicouche inspiré biologiquement du cortex visuel animal. L'architecture est particulièrement utile dans les applications de traitement d'image. Le premier CNN a été crée par Yann LeCun ; l'architecture était axée sur la reconnaissance de caractères manuscrits, comme l'interprétation de codes postaux. En tant que réseau profond, les premières couches reconnaissent les entités (telles que les arêtes) et les couches ultérieures recombinent ces entités en attributs de niveau supérieur de l'entrée.

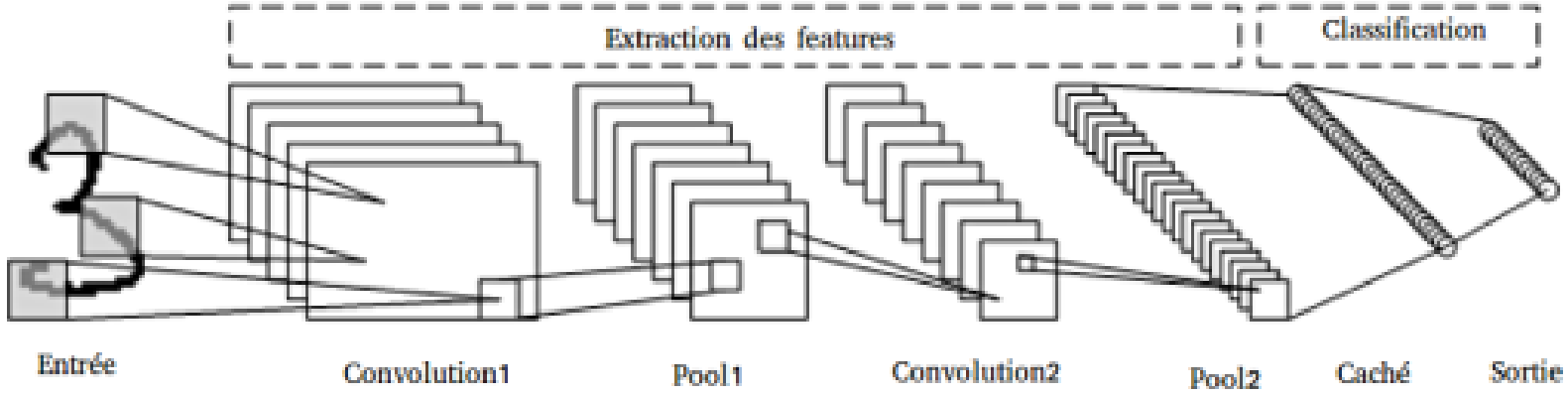
Réseau de neurones convolutif

L'architecture CNN est composée de plusieurs couches qui implémentent l'extraction de caractéristiques, puis la classification. L'image est divisée en champs réceptifs qui alimentent une couche convolutionnelle, qui extrait ensuite des entités de l'image d'entrée.

L´étape suivante est la mise en commun, qui réduit la dimensionnalité des entités extraites (par le biais d'un échantillonnage vers le bas) tout en conservant les informations les plus importantes (généralement via la mise en pool maximale). Une autre étape de convolution et de mise en commun est ensuite effectuée qui alimente un perceptron multicouche entièrement connecté. La couche de sortie finale de ce réseau est un ensemble de nœuds qui identifient les caractéristiques de l'image.

Vous formez le réseau en utilisant la rétropropagation.

Réseau de neurones convolutif

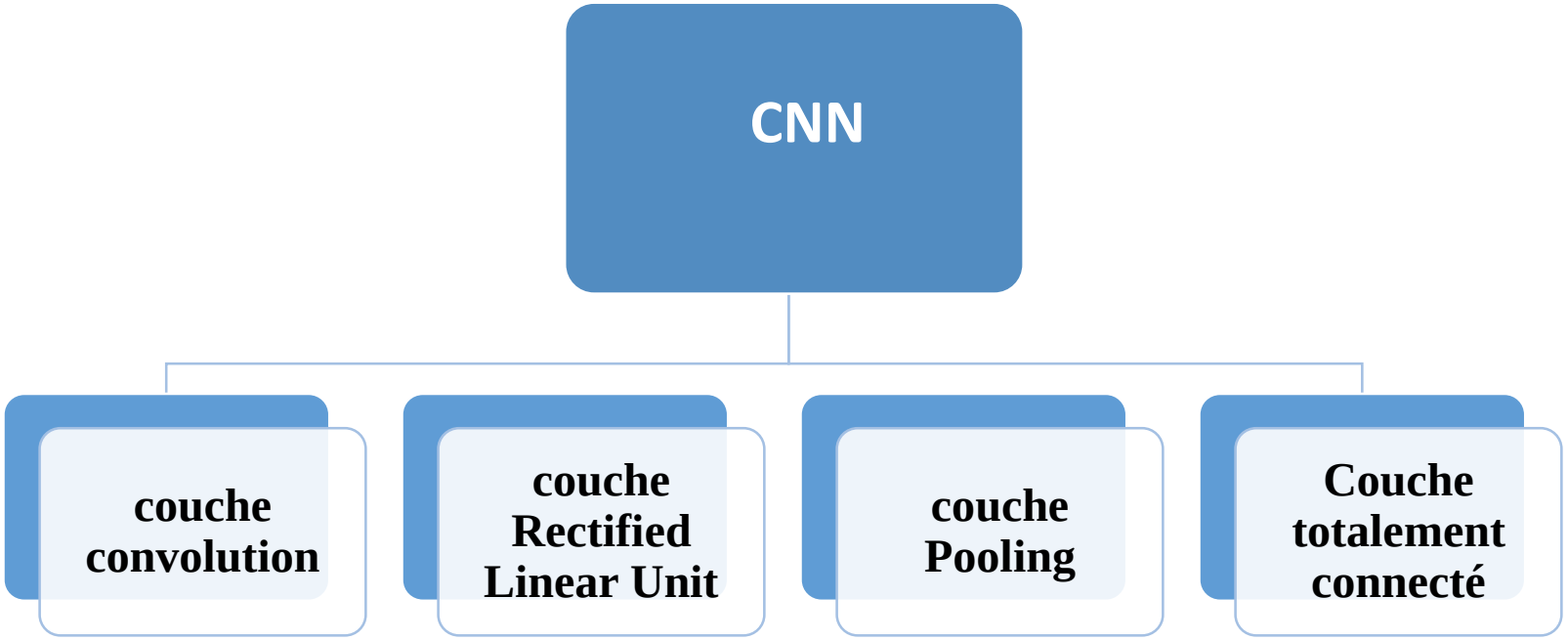


L'utilisation de couches profondes de traitement, de convolutions, de mise en commun et d'une couche de classification entièrement connectée a ouvert la porte à diverses nouvelles applications des réseaux neuronaux d'apprentissage profond. En plus du traitement de l'image, le CNN a été appliqué avec succès à la reconnaissance vidéo et à diverses tâches dans le traitement du langage naturel.

Il existe quatre (4) principales opérations illustrées dans le CNN à savoir :

- La couche convolution
- La couhe Rectified Linear Unit
- La couche Pooling
- La couche entièrement connectée

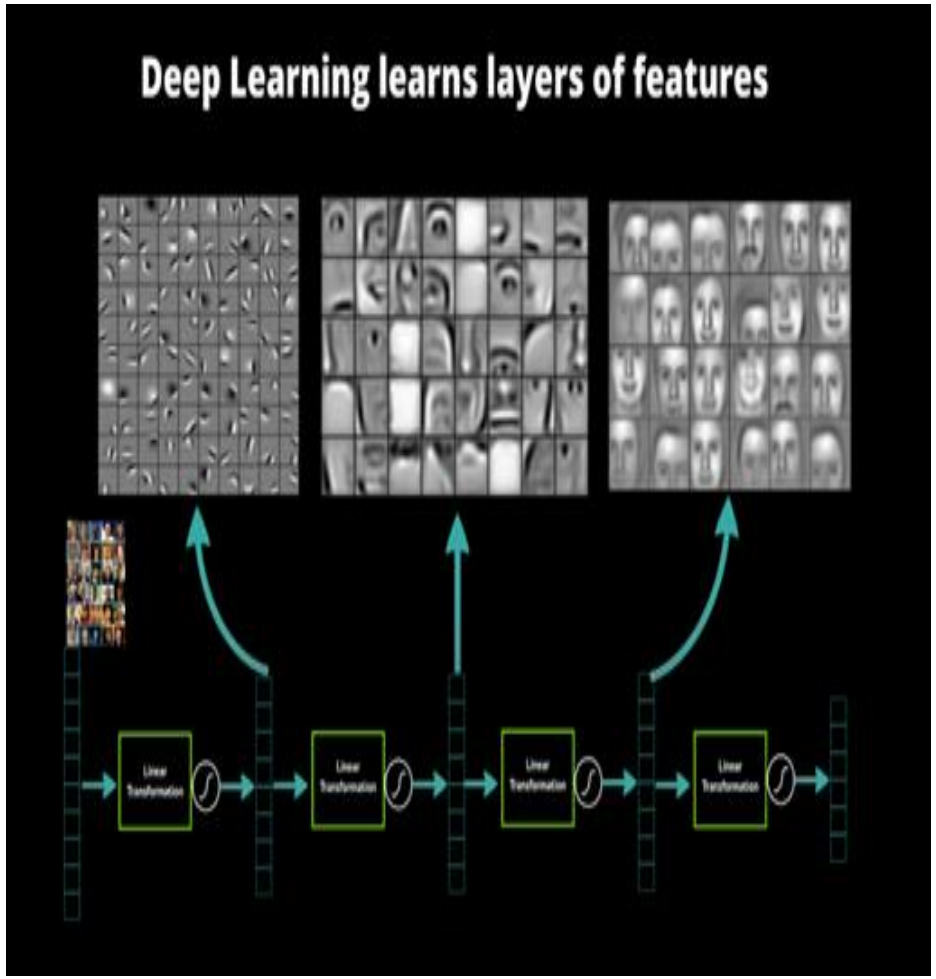
Réseau de neurones convolutif



Réseau de neurones convolutif: la convolution

La **convolution** est le cœur du réseau de neurones convolutif. À l'origine, une convolution est un outil mathématique (on parle de produit de convolution) très utilisé en retouche d'image**,** car il permet d'en faire ressortir l'extraction des caractéristiques à partir des images d'entrées, afin d'appliquer un bon **filtre**.

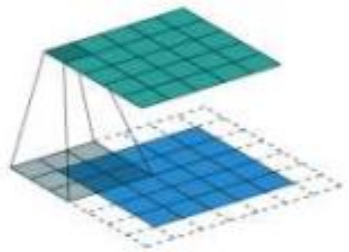
En fait, une convolution prend simplement en entrée une image et un filtre (qui est une autre image),effectue un calcul, puis renvoie une nouvelle image (généralement plus petite);



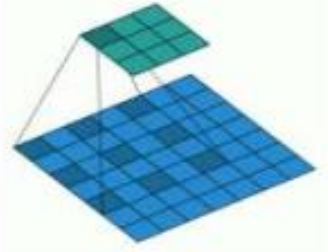
Les différents types de convolution

Il existe plusieurs types de convolutions, même si en général on utilise celle de base, il peut s'avérer utile de connaître les outils à notre disposition.

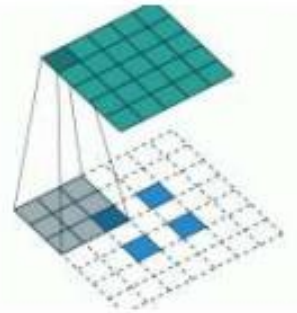
- La **convolution classique** qui représente le décalage du kernel entre chaque calcul, et le **p**adding qui est la manière dont on peut « dépasser » de l'image pour appliquer la convolution.
- La **dilated convolution**, identique à la convolution à ceci près que le kernel est éclaté (on prend, par exemple, un pixel sur deux pour calculer la convolution). Il y a un paramètre supplémentaire : la dilation **rate**, qui est le nombre de pixels à ignorer.
- La **transposed convolution**, qui construit la sortie comme si on inversait une convolution sur l'image
- La **séparable convolution**, qui est une convolution décomposable en convolutions plus simples



Une convolution de kernel



Une dilated convolution



Une transposed convolution

Fonction d'activation

Une fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Il dérive de l'équivalent biologique qui signifie "**potentiel d'activation**, lorsque le seuil de stimulation aura été atteint entraîne une réponse du neurone. Son but principal est de pouvoir permettre aux réseaux de neurones d'apprendre des fonctions plus complexes qu'une simple régression linéaire, car le simple fait de multiplier les poids d'une couche cachée est juste une transformation linéaire :

Rectified Linear Unit (ReLU) :

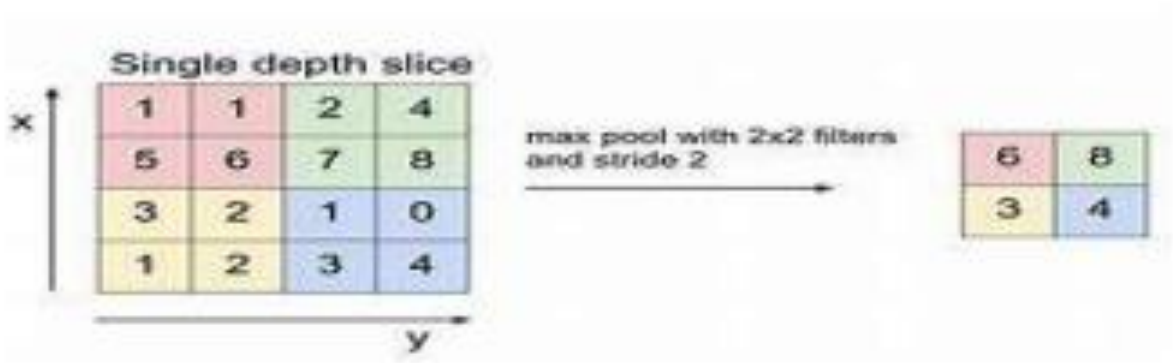
Elle est utilisée après chaque opération de convolution, ou toutes les valeurs de pixels négatifs sont mises à zéro. Le but de ReLU est d'introduire la non-linéarité dans notre CNN, puisque la plupart des données du monde réel, puisque la plupart de caractéristiques appliquées à l'une des cartes d'entrée donne une carte de sortie qui est également appelée carte de caractéristiques rectifiées.

Pooling

Elle permet de réduire la dimension de chaque carte de caractéristiques, mais conserve l'information la plus importante.IL peut être de différents types : max, moyenne, somme, etc.
En cas de Pooling max, on définit un voisinage spatial (par exemple, une fenêtre2×2) et de prendre le plus grand élément dans cette fenêtre.
La fonction de Pooling consiste à réduire progressivement la taille de la carte de caractéristiques rectifiée, en particulier, pooling :
- Rend les représentations d'entrée plus petites et plus faciles à gérer,
- Réduit le nombre de paramètres et les calculs dans le réseau,
- Rend le réseau invariant aux petites transformations, les distorsions et les translations dans l'image d'entrée (une faible distorsion en entrée ne change pas la sortie de la Pooling - car nous prenons le maximum / valeur moyenne dans un voisinage local).

→ Nous aide à arriver à une représentation presque invariante à l'échelle de notre image.

Pooling



Il existe plusieurs types de pooling :

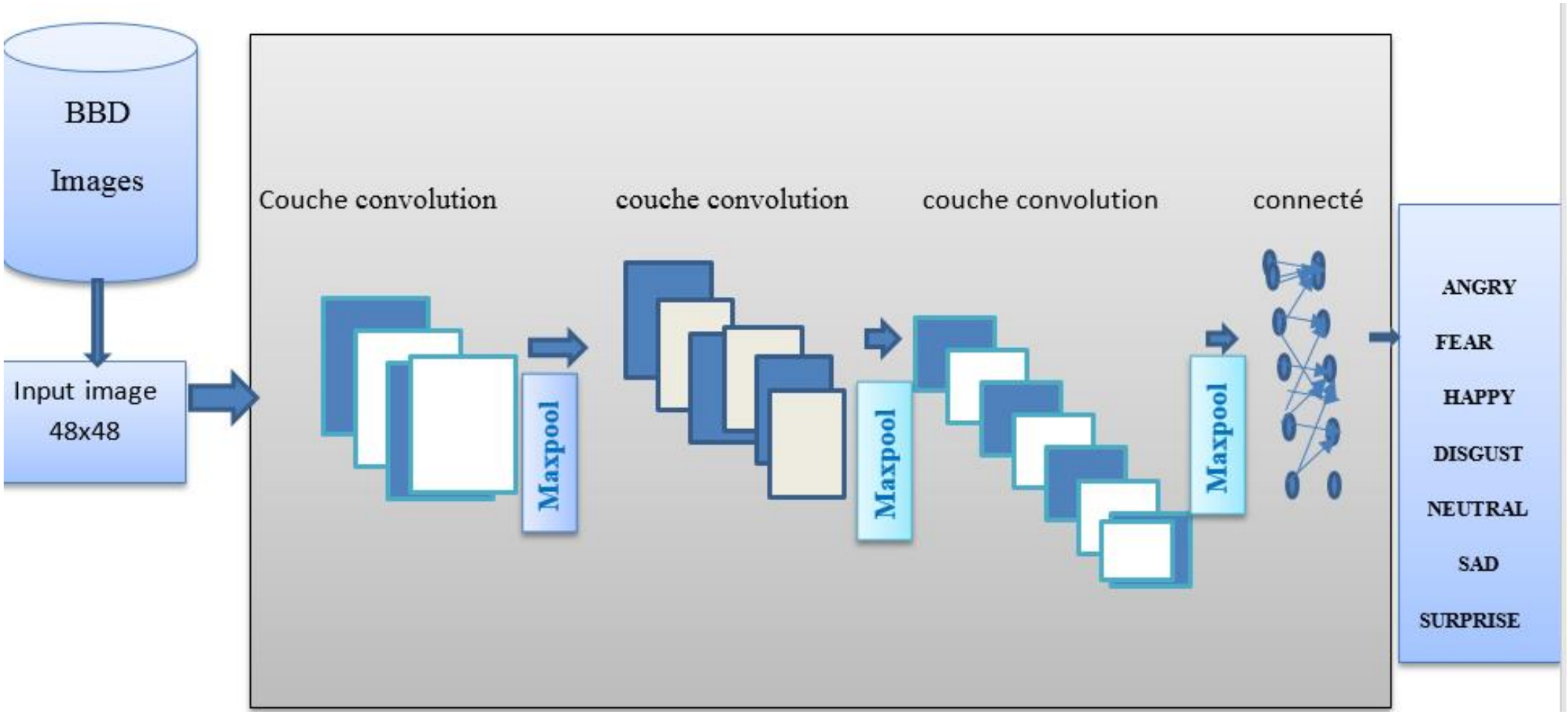
- Le « max pooling », qui revient à prendre la valeur maximale de la sélection. C'est le type le plus utilisé, car il est rapide à calculer (immédiat), et permet de simplifier efficacement l'image
- Le « mean pooling » (ou average pooling), soit la moyenne des pixels de la sélection : on calcule la somme de toutes les valeurs et on divise par le nombre de valeurs. On obtient ainsi une valeur intermédiaire pour représenter ce lot de pixels
- Le « sum pooling », c'est la moyenne sans avoir divisé par le nombre de valeurs (on ne calcule que leur somme),

Couches totalement connecté (Fully Connected Layer))

La couche entièrement connectée est un traditionnel perceptron multicouche (Multi Layer Perceptron) utilisant une fonction d'activation notamment appelée « softmax » dans la couche de sortie (d'autres classificateurs comme SVM peuvent également être utilisés). Le terme «entièrement connecté » implique que chaque neurone dans la couche précédente est connecté à chaque neurone sur la couche suivante.

La sortie des couches de convolution et de Pooling représente les fonctions de haut niveau de l'image d'entrée. Le but de la couche entièrement connectée est de pouvoir utiliser ces fonctions pour classer l'image d'entrée dans différentes classes en fonction de l'ensemble de données d'apprentissage.

PFE Master ADD_2019



PFE Master ADD_2019

- La base de données d'images des expressions faciales que nous avons eu à utiliser est celle de la de Fer2013 elle contient 35887 images expressions faciales qui sont reparties en deux parties à savoir :
- 28709 représentent les images d'apprentissage.
- 3589 images représentant les images de test

Figure. Echantillon d'images BDD de Fer2013

PFE Master ADD_2019

```
Initialization du model
model = Sequential ()
Adding Input Layer
model.add (Conv2D (32, (3, 3), padding='same', activation='relu',
input_shape=input_shape))
Adding more layers
model.add (Conv2D (32, (3, 3), activation='relu'))
model.add (MaxPooling2D (pool_size= (2, 2)))
model.add (Dropout (0.25))
model.add (Conv2D (64, (3, 3), padding='same', activation='relu'))
model.add (Conv2D (64, (3, 3), activation='relu'))
model.add (MaxPooling2D (pool_size= (2, 2)))
model.add (Dropout (0.25))
model.add (Conv2D (128, (3, 3), padding='same', activation='relu'))
model.add (Conv2D (64, (3, 3), activation='relu'))
model.add (MaxPooling2D (pool_size= (2, 2)))
model.add (Dropout (0.25))
Flattening
model.add (Flatten ())
Adding fully connected layer
model.add (Dense (512, activation='relu'))
model.add (Dropout (0.4))
Adding Output Layer
model.add (Dense (nClasses, activation='softmax'))
```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 32)	320
conv2d_2 (Conv2D)	(None, 46, 46, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 23, 23, 32)	0
dropout_1 (Dropout)	(None, 23, 23, 32)	0
conv2d_3 (Conv2D)	(None, 23, 23, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 11, 11, 64)	0
dropout_2 (Dropout)	(None, 11, 11, 64)	0
conv2d_4 (Conv2D)	(None, 11, 11, 128)	73856
max_pooling2d_3 (MaxPooling2	(None, 5, 5, 128)	0
dropout_3 (Dropout)	(None, 5, 5, 128)	0
flatten_1 (Flatten)	(None, 3200)	0
dense_1 (Dense)	(None, 512)	1638912
dropout_4 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 7)	3591

Total params: 1,744,423
Trainable params: 1,744,423

Non-trainable params: 0

PFE Master ADD_2019

```
batch_size = 256

epochs =200

model1.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])

model1.summary()


history = model1.fit(train_data, train_labels_one_hot, batch_size=batch_size, epochs=epochs, verbose=1,
                     validation_data=(test_data, test_labels_one_hot))
model1.evaluate(test_data, test_labels_one_hot)
```

Epoch 1/200

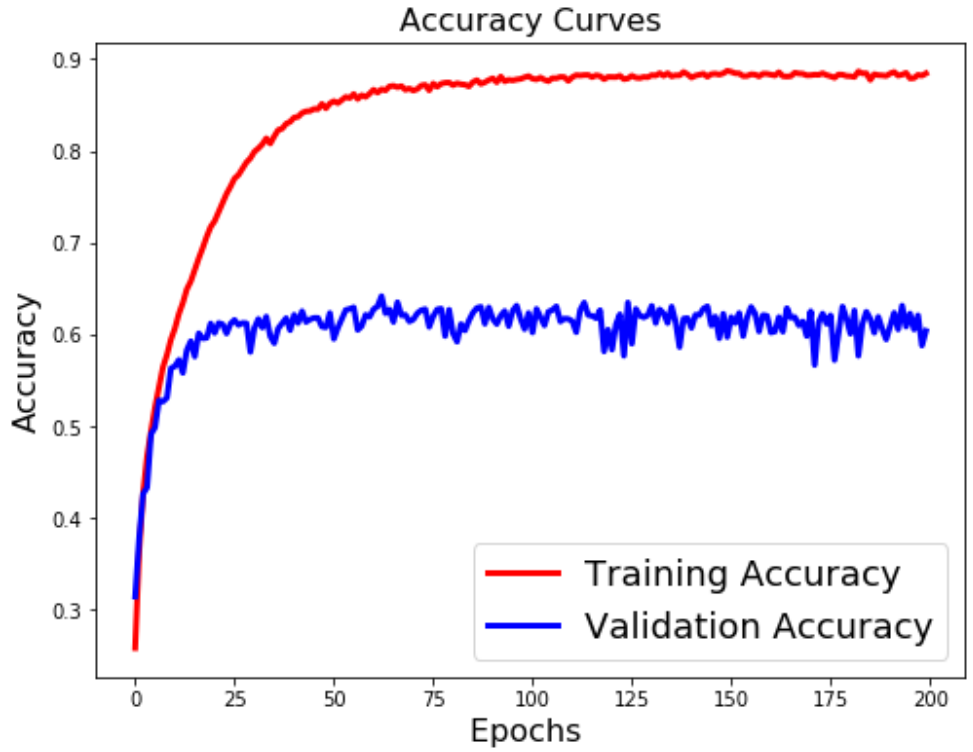
28709/28709 [==============================] - 6s 192us/step - loss: 1.8165 - acc: 0.2483 - val_loss: 1.7077 - val_acc: 0.2984

Epoch 200/200

28709/28709 [==============================] - 4s 152us/step - loss: 0.3451 - acc: 0.8903 - val_loss: 1.2742 - val_acc: 0.6124

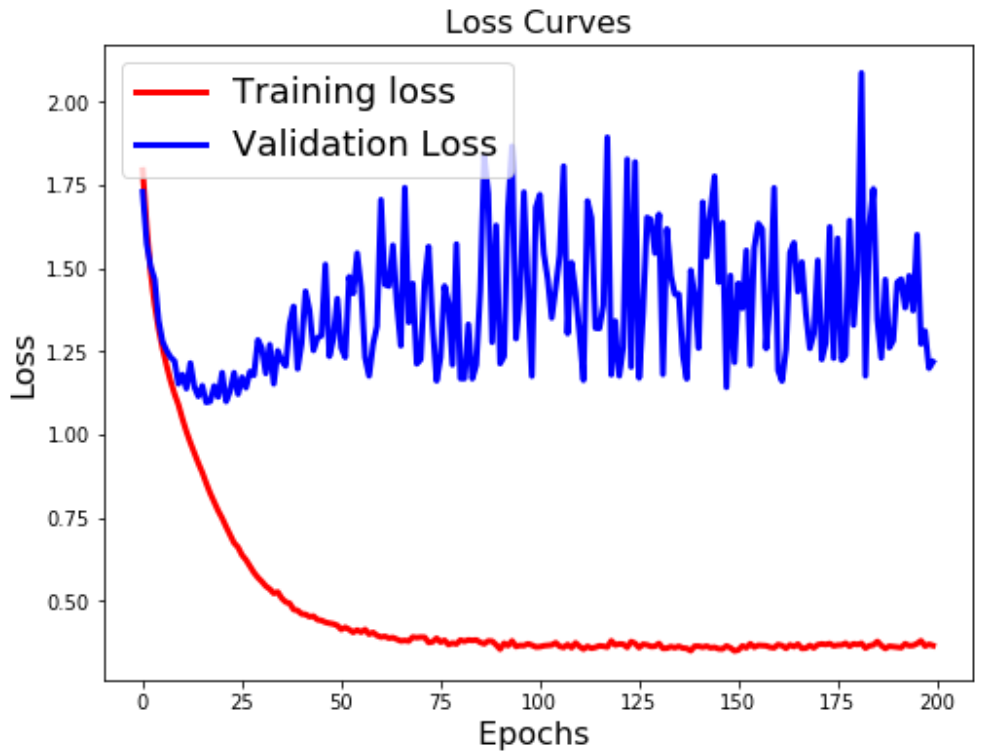
PFE Master ADD_2019

```
# Accuracy Curves
plt.figure(figsize=[8,6])
plt.plot(history.history['acc'],'r',linewidth=3.0)
plt.plot(history.history['val_acc'],'b',linewidth=3.0)
plt.legend(['Training Accuracy', 'Validation Accuracy'],fontsize=18)
plt.xlabel('Epochs ',fontsize=16)
plt.ylabel('Accuracy',fontsize=16)
plt.title('Accuracy Curves',fontsize=16)
```



PFE Master ADD_2019

```
# Loss Curves
plt.figure(figsize=[8,6])
plt.plot(history.history['loss'],'r',linewidth=3.0)
plt.plot(history.history['val_loss'],'b',linewidth=3.0)
plt.legend(['Training loss', 'Validation Loss'],fontsize=18)
plt.xlabel('Epochs ',fontsize=16)
plt.ylabel('Loss',fontsize=16)
plt.title('Loss Curves',fontsize=16)
```



PFE Master ADD_2019

Modèle	Training accuracy	Validation accuracy
SVM	43.36%	38.61%
CNN	72.25%	63.86%
CNN	90.11%	67.68%
CNN (modèle mémoire)	89,03%	61 ,24%

Le réseau de neurone récurrent RNN

Le réseau de neurone récurrent (recursive neural networks RNN en anglais) est l'une des architectures de réseau fondamentales à partir desquelles d'autres architectures d'apprentissage profond sont construites. La principale différence entre un réseau multicouche typique et un réseau récurrent est que les connexions sont complètement anticipées, un réseau récurrent peut avoir des connexions qui sont réinjectées dans des couches précédentes (ou dans la même couche). Cette rétroaction permet aux réseau de neurone récurrent de conserver la mémoire des entrées passées et des problèmes de modèle dans le temps.
Les réseaux de neurone récurrent sont constitués d'un ensemble riche d'architectures (une topologie populaire appelée LSTM).
Le différentiateur clé est la rétroaction dans le réseau, qui pourrait se manifester à partir d'une couche cachée, la couche de sortie, ou une combinaison de ceux-ci.
Ils existent plusieurs type de réseau de neurone récurent comme : fully recurrent networks, recursive neural networks, neural history compressor, gated recurrent unit neural networks et long short-term memory networks (LSTM).

Long short-term memory networks (LSTM)

En français réseau récurrent à mémoire court et long terme ou plus explicitement réseau de neurones récurrents à mémoire court-terme et long terme, est l'architecture de réseau de neurones récurrents la plus utilisée en pratique qui permet de répondre au problème de disparition de gradient. Le réseau LSTM a ´été proposé par Sepp Hochreiter et J¨urgen Schmidhuber en 1997. L'idée associée au LSTM est que chaque unité computationnelle est liée non seulement à un état caché mais également à un état de la cellule qui joue le rôle de mémoire.
Le passage à se fait par transfert à gain constant et égal à 1.

De cette façon les erreurs se propagent aux pas antérieurs (jusqu'`a 1000 étapes dans le passé) sans phénomène de disparition de gradient. L'état de la cellule peut être modifie a travers une porte qui autorise ou bloque la mise à jour (input gate). De même une porte contrôle si l'état de cellule est communiqué en sortie de l'unité LSTM (output gate). La version a plus répandue des LSTM utilise aussi une porte permettant la remise à zéro de l'état de la cellule (forget gate).