Introduction aux réseaux de neurones convolutifs

Axel Carlier, Sandrine Mouysset

2ème Année Science du Numérique - ENSEEIHT

1 Apprentissage par représentation

Le perceptron monocouche représenté ci-dessous constitue un réseau de neurones simpliste, qui permet d'obtenir de bons résultats pour des problèmes simples comme la reconnaissance de chiffres manuscrits (sur la base de données MNIST), mais qui échoue à résoudre des problèmes plus compliqués.

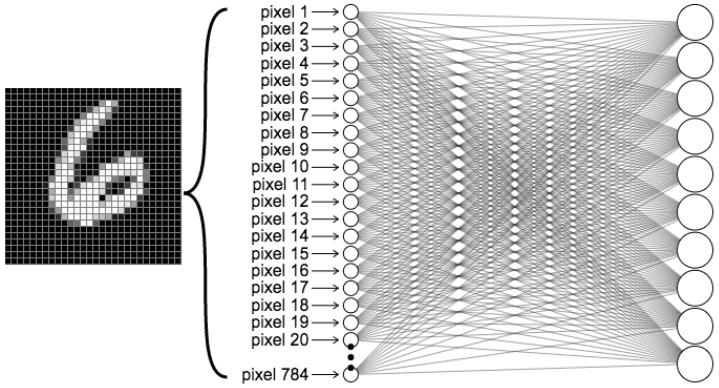

Figure 1: Intérêt du Deep Learning : architecture naïve 1 couche pour un chiffre *MNIST*

Les applications opérationnelles nécessitent souvent des structures de réseaux complexes, avec un nombre de couches conséquent. Le *deep learning*, ou apprentissage profond, est un terme utilisé pour décrire les approches basées sur des réseaux de neurones de grande profondeur, c'est-à-dire avec un grand nombre de couches cachées. [1, 3].

Un intérêt majeur des architectures profondes est que les différentes couches constituent une structure de représentation hiérarchique ; en d'autres termes, les premières couches du réseau permettent de détecter des structures locales, à petite échelle, et les couches suivantes combinent les structures de plus faible échelle pour détecter des structures à une échelle supérieure.

La force de l'apprentissage profond est que cette représentation hiérarchique émerge naturellement pendant la phase d'apprentissage ; c'est une différence majeure entre les réseaux de neurones et d'autres algorithmes d'**apprentissage par représentation**, où les représentations intermédiaires nécessitent d'être élaborées par un humain.

Cependant, avec l'augmentation du nombre de couches surviennent les problèmes de surajustement, ou surapprentissage (*overfitting*) et de disparition des gradients (*vanishing gradient*), sans parler du temps de calcul qui explose [2]. Ces problèmes sont dûs, pour partie, au grand nombre de paramètres des réseaux de neurones profonds.

2 Réseaux de neurones convolutifs

Les réseaux de neurones convolutifs sont, comme leur nom l'indique, construits à l'aide de l'opération de convolution. Ces réseaux sont très utilisées pour le traitement et la reconnaissance d'images.

L'idée centrale des réseaux convolutifs est basée sur le principe de "diviser pour régner" : leur architecture inclut des couches dédiées qui vont apprendre à extraire des caractéristiques de l'image. Celles-ci sont ensuite transmises à un ensemble de couches complètement connectées, plus classiques, qui effectuent la phase de reconnaissance.

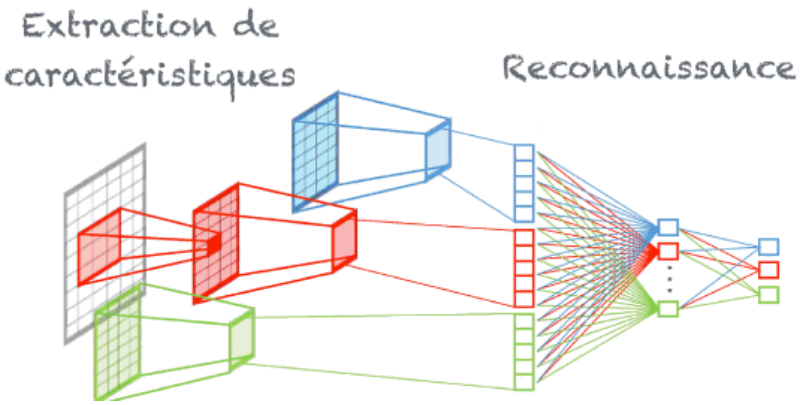

Figure 2: Exemple de CNN sur une image

Une autre façon de décrire cette approche est de la voir comme une décomposition hiérarchique du processus de reconnaissance, où chaque couche participe à la création des représentations de plus en plus abstraites et conceptuelles.

2.1 Opération de convolution sur des images

On appelle filtre de convolution (ou masque de convolution) une matrice carrée à coefficients réels. Étant donné une image, on peut calculer la **réponse du filtre** (c'est-à-dire, le résultat de la convolution entre l'image et le filtre) par le procédé illustré sur l'image ci-dessous.

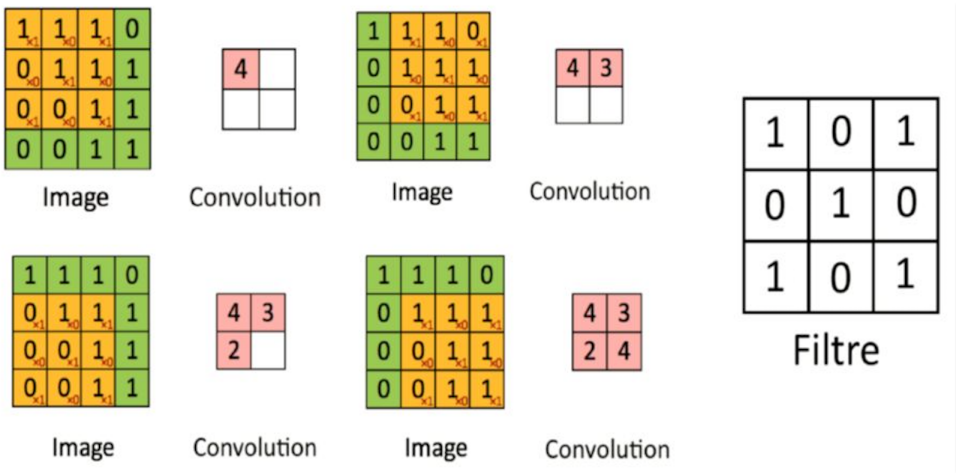

Figure 3: Principe de la convolution sur des images

La réponse d'un pixel donné est obtenue en tenant compte du voisinage défini par le masque et ses coefficients. Plus précisément, on multiplie la valeur de chaque pixel pris en compte par la valeur correspondante du masque de convolution et on additionne l'ensemble.

L'idée est donc de "paver" l'image de départ, c'est-à-dire de la découper en petites zones appelées "tuiles". Chaque tuile est traitée individuellement par un neurone, qui effectue une opération de filtrage classique en associant un poids à chaque pixel de la tuile. Tous les neurones ont donc les mêmes paramètres, ce qui permet d'obtenir le même traitement pour tous les pixels de la tuile. Ainsi, une convolution est assimilable à une couche complètement connectée, dans laquelle on aurait **répété les mêmes poids**. Cela permet de réduire drastiquement le nombre de paramètres du réseau, en bénéficiant du fait que l'on recherche souvent les mêmes structures dans toute l'image (par exemple, contours horizontaux et verticaux).

Remarque. Pour obtenir une réponse du filtre de convolution de taille égale à l'image d'entrée, on peut augmenter artificiellement la taille de l'image de départ (par exemple, avec des 0) pour compenser la perte induite par la convolution. On appelle ce procédé *padding*.

2.2 Sous-échantillonnage : couche de *Pooling*

Après avoir obtenu des caractéristiques à l'aide de la convolution, nous aimerions ensuite les utiliser pour la classification. En théorie, on pourrait utiliser toutes les caractéristiques extraites avec un classifieur tel que la fonction *softmax*, mais pour des images de grande taille, le nombre de paramètres est tel que l'apprentissage peut poser des problèmes.

Prenons par exemple des images de 96x96 pixels, et supposons que nous ayons appris 400 caractéristiques sur 8x8 entrées. Chaque convolution donne une sortie de taille (96 - 8 + 1) * (96 - 8 + 1) = 7921, et comme nous avons 400 caractéristiques, cela donne un vecteur de 7921 * 400 = 3 168 400 caractéristiques par exemple. L'apprentissage d'un classifieur avec des entrées ayant plus de 3 millions de caractéristiques peut être difficile à manier et peut aussi être sujet au surapprentissage.

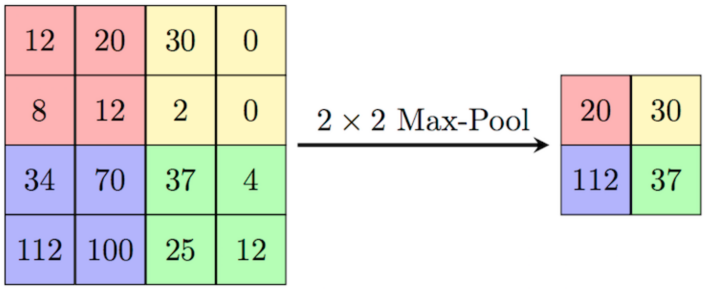

Figure 4: Exemple de *Max Pooling*

Pour y remédier, on peut agréger les statistiques de ces caractéristiques à une échelle locale. Par exemple, on pourrait calculer la valeur moyenne (ou maximale) d'une caractéristique particulière sur une sous-région de l'image. Ces statistiques sommaires sont beaucoup moins volumineuses (comparativement à l'utilisation de toutes les caractéristiques extraites) et peuvent également améliorer les résultats (moins de surapprentissage). Nous appelons cette opération *pooling*, ou parfois *mean pooling* ou *max pooling* (selon l'opération de pooling appliquée).

2.3 Architecture typique d'un réseau de neurones convolutif

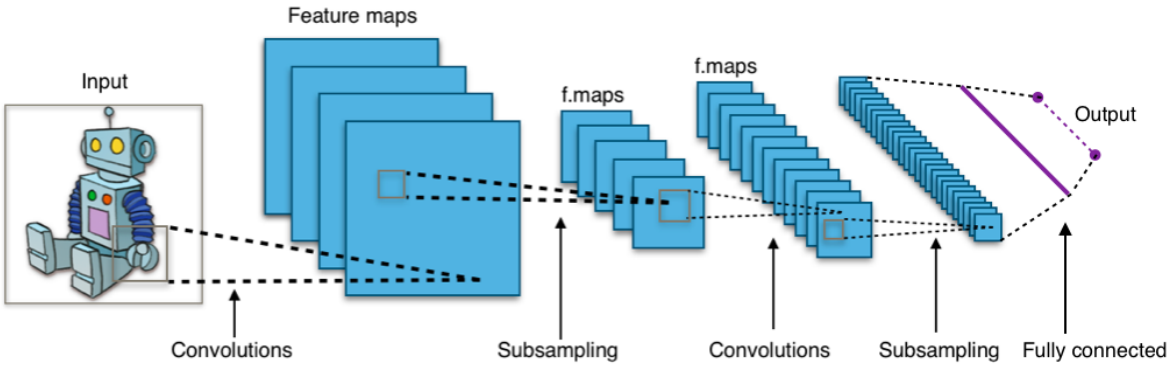

Figure 5: Exemple de CNN sur une image

Voici un schéma représentant un réseau de neurones convolutif classique ; on y trouve les composants suivants :

1. les couches de *convolution* qui calculent la réponse à des filtres de convolution. Il faut noter qu'une fonction d'activation est appliquée à la sortie de ces couches (par exemple, la fonction ReLU), exactement comme pour les neurones classiques ;

2. les couches de *pooling* (*max*, *mean* ou *sum pooling*) qui compressent l'information en réduisant la taille de l'image intermédiaire par sous-échantillonnage (*subsampling*) ;

3. les couches *fully connected* qui correspondent à des couches classiques de neurones formels totalement connectés ;

4. une couche de sortie, qui est la dernière du réseau. Elle porte notamment une fonction d'activation qui est spécifique au problème. Par exemple, la fonction *softmax* est utilisée en classification..

La figure ci-dessous illustre bien la hiérarchie des informations apprises par un réseau de neurones convolutifs. Les filtres de convolution appris dans différentes couches sont ici affichés. La première couche détecte des microstructures locales, combinées dans les filtres de la 2nde couche pour faire apparaître des parties de visage, elles-mêmes combinées dans la dernière couche pour faire apparaître des visages complets.

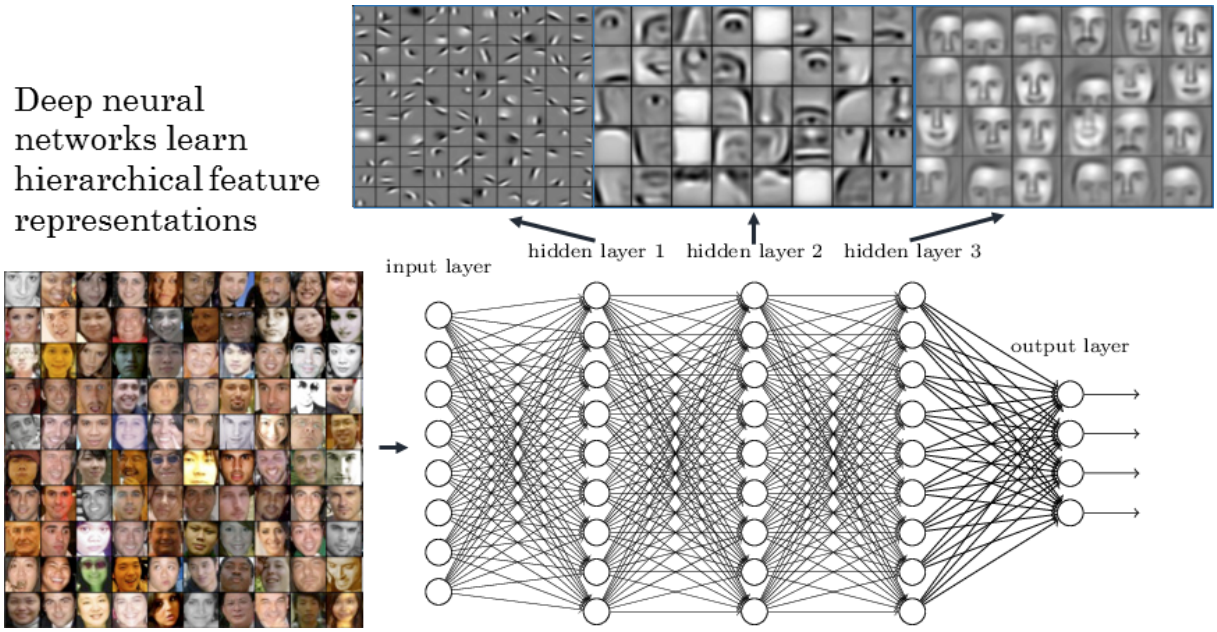

Figure 6: Exemple de CNN sur des visages : visualisation des caractéristiques extraites par chaque couche de neurones dans un réseau convolutif

References

[1] Aurélien Géron. *Neural networks and deep learning.* O'Reilly, 2018.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT press, 2016.

[3] Jean-Claude Heudin. *Comprendre le Deep Learning: Une introduction aux réseaux de neurones.* Science eBook, 2016.