Apprentissage statistique
TP4 : Réseaux de neurones convolutifs

Éloi Zablocki - Arthur Pajot (inspiration par Thomas Robert)

2020-2021

L'objectif de ce TD est de se familiariser avec les réseaux de neurones convolutifs, très adaptés aux images. Pour ce faire, nous étudierons les couches classiques de ce type de réseaux et nous mettrons en place un premier réseau appris sur le jeu de données standard CIFAR-10.

Exercice 1 *Introduction aux réseaux convolutifs*

Les réseaux de neurones convolutionnels (CNN) sont devenus les architectures état-de-l'art dans la quasi-totalité des tâches de machine learning appliquées aux images. Ces réseaux diffèrent des réseaux plus classiques par les couches de base utilisées dans leur architecture. On y retrouve souvent deux types de couches différents : la convolution et le pooling.

Couches de convolution *Entrée* : Ces couches prennent en entrée un ensemble de D feature maps (c'est-à-dire un tenseur soit l'équivalent en 3 dimensions d'une matrice), chaque feature map étant une matrice de taille $n_x \times n_y$. On a donc une entrée de taille $n_x \times x_y \times D$

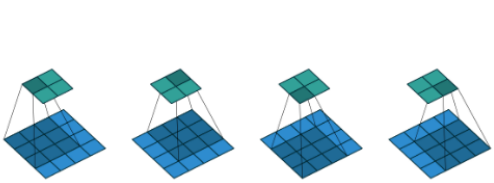
Sortie : Sur cette entrée, on applique C convolutions, chacune avec un filtre de convolution de taille $w \times h \times D$, on a généralement $w = h = k$ qu'on appelle *taille du kernel*. Ce sont ces C filtres qui constituent les paramètres que l'on va apprendre. Chaque convolution avec un filtre produit une feature map de sortie, on a donc en sortie un ensemble de feature maps de taille $n'_x \times n'_y \times C$, où n'_x et n'_y dépendent de la façon dont on réalise la convolution.

Hyperparamètres : En plus du nombre de filtres et de la taille du kernel, il y a deux hyperparmètres courants pour la convolution, le **padding** et le **stride**. Le padding indique combien de rangées de 0 on ajoute autour de l'entrée. Le stride indique de combien de pas on se déplace entre deux calculs de la convolution. La convolution "classique" a une sortie plus petite que l'entrée à cause de la taille du filtre que l'on doit placer à l'intérieur de la feature map d'entrée. Ajouter du padding permet par exemple de retrouver la taille initiale. Ajouter un stride permet de sauter des valeurs, cela correspond à faire du sous-échantillonnage de la sortie.

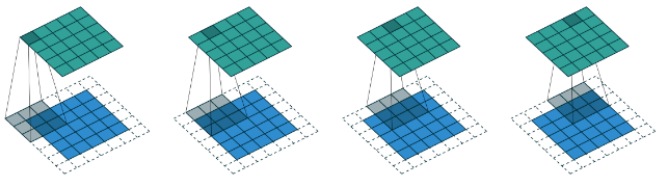
Références : Une démonstration de la convolution est visible à l'adresse http://cs231n.github.io/assets/conv-demo/index.html

Max Pooling *Principe* : Le pooling est une fonction de sous-échantillonnage spatial. Elle prend toujours en entrée des feature maps de taille $n_x \times n_y \times D$, et réduit les deux premières dimensions spatiales. Le calcul s'effectue selon le même principe qu'une convolution, mais s'applique sur chaque feature map indépendamment. On parcourt donc chaque feature map avec une fenêtre glissante, mais au lieu de réaliser un produit de convolution entre la fenêtre et un filtre, on réalise une opération de pooling sur la fenêtre d'entrée pour produire une valeur. On obtient donc une sortie de taille $n'_x \times n'_y \times D$ avec n'_x et n'_y significativement plus petits que n_x et n_y (souvent d'un facteur 2).

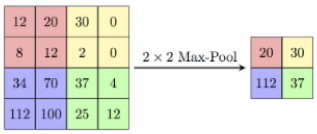
Hyperparamètres : Une couche de pooling a une taille de kernel (définissant la taille de la fenêtre), un stride et du padding qui se comportent comme pour la convolution, et une opération de pooling, les plus courantes étant le *max pooling* (on garde la valeur maximale dans la fenêtre) et l'*average pooling* (on



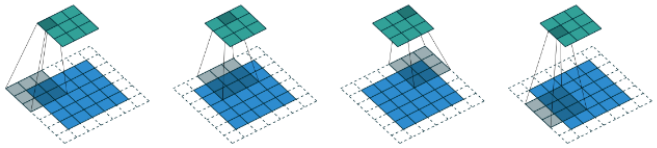
(a) Convolution simple, pas de padding, stride de 1, kernel de taille 3.



(b) Convolution avec padding de 1, stride de 1, kernel de taille 3.



(c) Exemple de max pooling, pas de padding, stride de 2, kernel de taille 2.



(d) Convolution avec padding de 1, stride de 2, kernel de taille 3.

FIGURE 1 – Exemples de convolution et de pooling

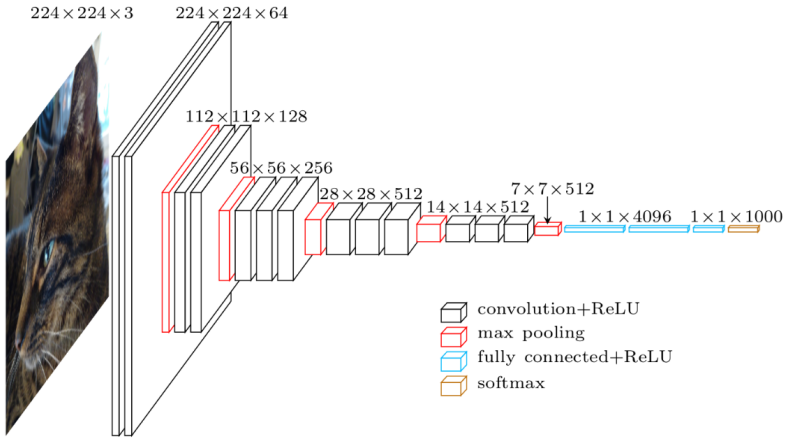


FIGURE 2 – Réseau VGG16

prend la valeur moyenne de la fenêtre). Un pooling très courant est un max pooling avec kernel de taille 2, stride de taille 2 et sans padding.

Architectures convolutives courantes Les réseaux de neurones convolutionnels classiques sont généralement composés d'une succession de couches de convolutions (avec ReLU) avec de plus en plus de filtres, et dont la dimension spatiale est progressivement réduite par des couches de max pooling possiblement jusqu'à aggregation totale des dimensions spatiales, il ne reste donc plus que la "profondeur" correspondant au nombre de filtres appliqués par la dernière convolution ($1 \times 1 \times C$). On y ajoute enfin généralement une ou quelques couches linéaires (appelées *fully-connected*). Un exemple de ce type d'architecture est le réseau VGG16. Depuis, des architectures plus complexes se sont développées, notamment les architectures Inception ou ResNet, et leurs dérivés et combinaisons.

Question 1

Considérant un seul filtre de convolution de padding p, de stride s et de taille de kernel k, pour une entrée de taille $x \times y \times z$, quelle sera la taille de sortie ?

Combien y a-t-il de poids à apprendre ?

Combien de poids aurait-il fallu apprendre si une couche fully-connected devait produire une sortie de la même taille ?

Question 2
Quels avantages apporte la convolution par rapport à des couches *fully-connected* ?

Quelle est sa limite principale ?

Question 3
Quel interêt voyez-vous à l'usage du pooling spatial ?

Question 4
Supposons qu'on essaye de calculer la sortie d'un réseau convolutionnel classique (par exemple celui en Figure 2) pour une image d'entrée plus grande que la taille initialement prévue (224×224 dans l'exemple). Peut-on (sans modifier l'image) calculer tout ou une partie des couches du réseau ?

Question 5
Montrer que l'on peut voir les couches fully-connected comme des convolutions particulières.

Question 6
Supposons que l'on remplace donc les fully-connected par leur équivalent en convolutions, répondre à nouveau à la question 4. Si on peut calculer la sortie, quelle est sa forme et son intérêt ?

Question 7
On appelle champ récepteur (*receptive field*) d'un neurone l'ensemble des pixels de l'image dont la sortie de ce neurone dépend. Quelles sont les tailles des receptive fields des neurones de la première et de la deuxième couche de convolution ? Pouvez-vous imaginer ce qu'il se passe pour les couches plus profondes ? Comment l'interpréter ?