UNIVERSITÉ TOULOUSE III PAUL SABATIER Université de Toulouse

TD 2 — Convolution, CNN

Exercice 1 : VGG16

Question 1. On considère un réseau de neurones convolutif qui prend en entrée des images couleur RGB de taille 224×224 pixels. Si la première couche de convolution a 16 filtres de dimension 3×3, combien de poids au total comporte cette couche ?

Question 2.

- Pour réaliser une classification binaire, quelle fonction d'activation va-t-on utiliser pour l'unique neurone de la dernière couche d'un réseau ? Peut-on utiliser deux neurones en sortie pour faire la même tâche ?
- Même question pour une classification avec 10 classes mutuellement exclusives à distinguer.
- Toujours avec 10 classes différentes, comment peut-on coder/représenter en pratique ces 10 classes pour un réseau de neurones ? Comment s'appelle ce type d'encodage d'étiquettes ?
- Donner deux pré-traitements possibles sur des images données en entrée d'un réseau de neurones.

Question 3. On considère le réseau de classification d'objets dans les images appelé VGG16. Son architecture est illustrée dans la figure 1.

- Combien de classes d'objets ce réseau peut-il prédire ?
- Sachant qu'il permet de détecter des classes mutuellement exclusives, avec quelle fonction de coût peut-on entraîner ce modèle ?
- Quel est le rôle des couches « maxpooling » ? Détaillez leur fonctionnement.
- Ce réseau est constitué de deux blocs : le premier composé d'une succession de couches de convolutions, le deuxième de couches denses (MLP). Quel rôle a chacun de ces blocs ?

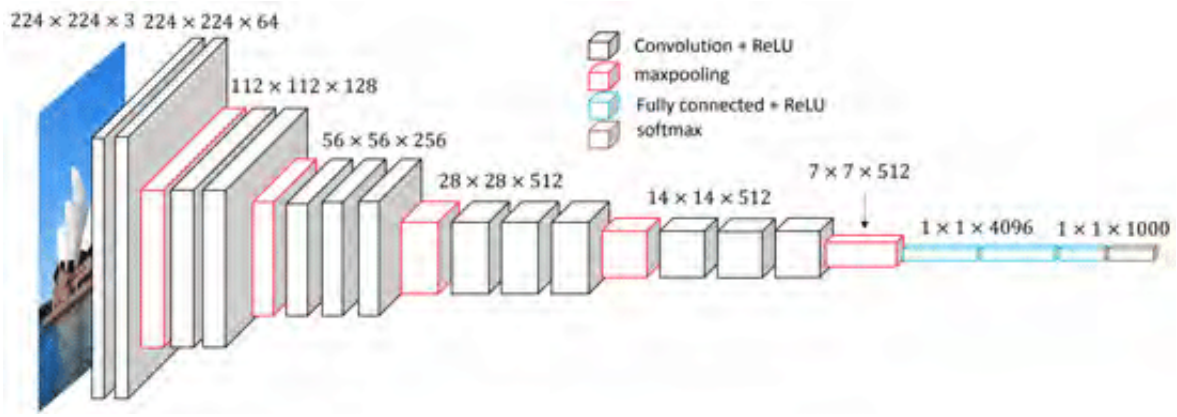

FIGURE 1 – Architecture de VGG16.

Exercice 2 : convolution 1-d

Question 1. Exemple
Soit le tenseur x = [2, 3, 0, -1] et le kernel w = [1, 2, -1]

- Calculer le résultat de la convolution sur x avec le kernel w.
- Donner la matrice W utilisant les poids de w qui permet de faire l'opération équivalente par une multiplication matricielle $y = Wx$.
- Quelle est la longueur du résultat de cette convolution ?

Question 2. Transposons
Considérons maintenant le résultat précédent y comme une entrée

- Calculer le résultat de la multiplication matricielle $z = W^t y^t$.
- Quelle est la longueur du résultat z ? Commenter.

Exercice 3 : convolutions 2d standard, depthwise et pointwise

Notations

Les tenseurs sont aplatis pour pouvoir être lus facilement. Nous avons les canaux de sortie (C_{out}) de haut en bas et les canaux d'entrée de gauche à droite (C_{in}), et une ligne horizontale (resp. verticale) délimite les deux.

Question 1. Conv2d standard
Voici la définition du noyau (*kernel*) w de notre première convolution :
w.size() = (C_{out}, C_{in}, k_h, k_w) = (2, 2, 2, 2)
et ses valeurs :

$$w = \left[\begin{array}{cc|cc} 1 & \frac{1}{2} & -1 & -\frac{1}{2} \\ \frac{1}{3} & \frac{1}{4} & -\frac{1}{3} & -\frac{1}{4} \\ \hline 1 & 2 & -1 & -2 \\ 3 & 4 & -3 & -4 \end{array}\right]$$

On considère un tenseur « input » (batch de taille 1), avec :
input.size() = (B, C_{in}, h_{in}, w_{in}) = (1,2,4,4)
et dont les valeurs des "pixels" sont les suivantes :

$$input = \left[\begin{array}{cccc|cccc} 1 & 2 & 3 & 4 & -1 & -2 & -3 & -4 \\ 5 & 6 & 7 & 8 & -5 & -6 & -7 & -8 \\ 9 & 10 & 11 & 12 & -9 & -10 & -11 & -12 \\ 13 & 14 & 15 & 16 & -13 & -14 & -15 & -16 \end{array}\right]$$

Soit m une couche Conv2D telle que :

```
m = nn.Conv2d(Cin, Cout, (kh,kw), bias=False, stride=2)
```

- Quelles sont les dimensions du résultat de cette convolution appliquée à l'input ?
- Quel est le tenseur résultat m(input, w) ?
- Nous avons vu en cours que l'implantation de la convolution dans PyTorch est faite à l'aide d'une multiplication matricielle. Cet algorithme, appelé Im2Col, redimensionne à la fois le kernel et l'input pour obtenir le résultat désiré (un reshape final est nécessaire pour revenir à un tenseur 4-d en sortie).
  - Le batch d'inputs subit le redimensionnement suivant : (B, C_{in}, h_{in}, w_{in}) en (C_{in} * k_h * k_w , B * H_{out} * W_{out})
  - Et pour le kernel : (C_{out}, C_{in}, k_h, k_w) en (C_{out}, C_{in} * k_h * k_w)

  Donner les deux matrices input et kernel redimensionnées par Im2Col pour notre exemple.

Question 2. Conv2d séparable en profondeur ou *Depthwise Conv2d*
Pour pouvoir utiliser ce type de convolution, on utilise l'option de PyTorch *groups* = C_{in}. On a en outre besoin que C_{out} soit divisible par la valeur de *groups* choisie. Dans notre exemple, nous aurions une possibilité :
w.size() = (C_{out}, C_{in}/groups, k_h, k_w) = (2,1,2,2)
Considérons le noyau suivant :

$$w = \left[\begin{array}{cc} 1 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{4} \\ \hline 1 & 2 \\ 3 & 4 \end{array}\right]$$

Soit m la couche depthwise Conv2D utilisant ce noyau :

```
m = nn.Conv2d(Cin, Cout, (kh,kw), groups=Cin, bias=False, stride=2)
```

— Quelles sont les dimensions et le tenseur résultat de m(input, w) ?

Question 3. Conv2d Pointwise
Dans ce cas, les dimensions du kernel kh et kw valent 1 :

```
wp.size() = (Cout, Cin, kh, kw) = (2, 2, 1, 1)
```

Soit le noyau suivant :

$$wp = \left[\begin{array}{c|c} 1 & -1 \\ \hline 0 & 0 \end{array}\right]$$

Appliquons cette convolution pointwise sur la sortie de la convolution depthwise précédente (ce qui est fait en pratique dans les CNN utilisant ces deux types de couches) :
Soit n la couche de convolution pointwise avec :

```
m = nn.Conv2d(Cin, Cout, (kh,kw), groups=Cin, bias=False, stride=2)
n = nn.Conv2d(2, 2, (1,1), bias = False)
```

— Quelles sont les dimensions et le tenseur résultat de n(m(input,w),wp) ?

Question 4. Peut-on approcher une convolution standard par la combinaison d'une convolution depthwise suivie d'une convolution pointwise ?
L'idée serait de voir si on peut trouver, pour notre exemple, un noyau wp qui permette d'approximer le résultat trouvé à la question 1 :

$$wp = \left[\begin{array}{c|c} a & b \\ \hline c & d \end{array}\right]$$