Introduction aux réseaux de neurones – exercices

Question 1. Calculer $\nabla_{\mathbf{a}} L_{\text{nlv}}(\hat{y}_y, y)$, c'est-à-dire le gradient qui modifiera les poids associés à la couche de sortie. Pour ce faire, procédons en trois étapes

(a) Calculer la dérivée partielle associée au neurone de sortie correspondant à la bonne classe $(y = i)$:

$$\frac{\partial}{\partial a_y} L_{\text{nlv}}(\hat{y}_y, y) = \frac{\partial}{\partial a_y} \ln \left[\frac{1}{\text{softmax}(a_y)} \right].$$

<u>SOLUTION:</u>

$$\begin{aligned}
\frac{\partial}{\partial a_y} L_{\text{nlv}}(\hat{y}_y, y) &= \frac{\partial}{\partial a_y} \ln \left[\frac{1}{\text{softmax}(a_y)} \right] \\
&= \frac{\partial}{\partial a_y} \ln \left[\frac{\sum_{j=1}^{C} e^{a_j}}{e^{a_y}} \right] \\
&= \frac{1}{\sum_{j=1}^{C} e^{a_j}} \times \frac{\partial}{\partial a_y} \left[\sum_{j=1}^{C} e^{a_j} \right] - \frac{\partial}{\partial a_y} a_y \\
&= \frac{1}{\sum_{j=1}^{C} e^{a_j}} \times a_y - 1 \\
&= \text{softmax}(a_y) - 1 \\
&= \hat{y}_y - 1
\end{aligned}$$

(b) Calculer la dérivée partielle associée au neurone de sortie correspondant à la bonne classe $(y \neq i)$:

$$\frac{\partial}{\partial a_i} L_{\text{nlv}}(\hat{y}_y, y) = \frac{\partial}{\partial a_i} \ln \left[\frac{1}{\text{softmax}(a_y)} \right], \text{ avec } a_i \neq a_y \,.$$

<u>SOLUTION:</u>

$$\begin{aligned}
\frac{\partial}{\partial a_i} L_{\text{nlv}}(\hat{y}_y, y) &= \frac{\partial}{\partial a_i} \ln \left[\frac{1}{\text{softmax}(a_y)} \right] \\
&= \frac{\partial}{\partial a_i} \ln \left[\frac{\sum_{j=1}^{C} e^{a_j}}{e^{a_y}} \right] \\
&= \frac{1}{\sum_{j=1}^{C} e^{a_j}} \times \frac{\partial}{\partial a_i} \left[\sum_{j=1}^{C} e^{a_j} \right] - \frac{\partial}{\partial a_i} a_y \\
&= \frac{1}{\sum_{j=1}^{C} e^{a_j}} \times a_i \\
&= \text{softmax}(a_i) \\
&= \hat{y}_i
\end{aligned}$$

(c) À partir des réponses (a) et (b), déduire le gradient $\nabla_{\mathbf{a}} L_{\text{nlv}}(\hat{y}_y, y)$.

<u>SOLUTION:</u>

$$\nabla_{\mathbf{a}} L_{\text{nlv}}(\hat{y}_y, y) = \hat{\mathbf{y}} - \mathbf{y}\,.$$

Question 2. Considérons un réseau de neurones qui reçoit en entrée des images en teintes de gris représentées par des matrices de 20×20 pixels. Chacune de ces images illustre un des fruits suivant : banane, pomme, poire ou ananas. Il s'agit donc d'un problème de classification multi-classes ; le réseau possède quatre neurones de sorties.

(a) Si le réseau possède une seule couche cachée pleinement connectée de n neurones, combien de paramètres devrons-nous optimiser par descente de gradient ?

— Si on n'utilise aucun paramètre de biais, ni pour la couche cachée et ni pour la couche de sortie ?

<u>SOLUTION:</u>
Paramètres de la couche cachée : $400n$;
Paramètres de la couche de sortie : $n4$.
Total : $404n$ paramètres.

— Si on inclut les paramètres de biais pour la couche cachée et pour la couche de sortie ?

<u>SOLUTION:</u>
Paramètres de la couche cachée : $(400 + 1)n$;
Paramètres de la couche de sortie : $(n + 1)4$.
Total : $405n + 4$ paramètres.

(b) Si le réseau possède une première couche de m filtres de convolutions de taille 5×5, suivie d'une couche cachée pleinement connectée de n neurones, et d'aucun paramètre de biais, combien de paramètres devrons-nous optimiser par descente de gradient ?

<u>SOLUTION:</u>
Paramètres de la couche de convolution : $5^2m = 25m$;
Paramètres de la couche pleinement connectée : $(20 - 4)^2mn = 256mn$.
Paramètres de la couche de sortie : $n4$.
Total : $256mn + 25m + 4n$ paramètres.

(c) Si le réseau possède une première couche de m filtres de convolutions de taille 5×5, suivit d'une couche de *MaxPooling* avec une taille de pas de 4, suivit d'une couche cachée pleinement connectée de n neurones, et d'aucun paramètre de biais, combien de paramètres devrons-nous optimiser par descente de gradient ?

> SOLUTION:
> Paramètres de la couche de convolution : $5^2m = 25m$;
> Paramètres de la couche pleinement connectée : $(\frac{20-4}{4})^2mn = 16mn$.
> Paramètres de la couche de sortie : $n4$.
> **Total :** $16mn + 25m + 4n$ paramètres.

(d) Enfin, considérons l'architecture suivante :

— Entrée du réseau : images de tailles 20×20 ;
— 1ère couche : m_1 filtres de convolutions de taille 5×5, sans biais.
— *MaxPooling* avec taille de pas 2
— 2e couche : m_2 filtres de convolutions de taille $3 \times 3 \times m_1$, sans biais.
— *MaxPooling* avec taille de pas 2
— 3e couche : n_1 neurones pleinement connectés, sans biais
— 4e couche : n_2 neurones pleinement connectés, sans biais
— 5e couche : 4 neurones de sortie, avec biais.

> SOLUTION:
> — 1ère couche : $5^2m_1 = 25m_1$
> — 2e couche : $3^2m_1m_2 = 9m_1m_2$
> — 3e couche : $\left(\frac{(20-4)/2-2}{2}\right)^2 m_2n_1 = 9m_2n_1$
> — 4e couche : n_1n_2
> — 5e couche : $(n_2 + 1)4$
> **Total :** $25m_1 + 9m_1m_2 + 9m_2n_1 + n_1n_2 + 4n_2 + 4$ paramètres.

Question 3. Reprendre la question 2(a-d) en considérant cette fois-ci que les images sont en couleurs. Le réseau reçoit donc en entrée des matrices de taille $20 \times 20 \times 3$.

<u>SOLUTION:</u>

(a)
- Sans paramètres de biais :
  Paramètres de la couche cachée : $20 \times 20 \times 3 \times n = 1200n$;
  Paramètres de la couche de sortie : $n4$.
  Total : $1204n$ paramètres.

- Avec paramètres de biais :
  Paramètres de la couche cachée : $(1200 + 1)n$;
  Paramètres de la couche de sortie : $(n + 1)4$.
  Total : $1205n + 4$ paramètres.

(b) Paramètres de la couche de convolution : $5^2 \times 3m = 75m$;
Paramètres de la couche pleinement connectée : $(20 - 4)^2 mn = 256mn$.
Paramètres de la couche de sortie : $n4$.
Total : $256mn + 75m + 4n$ paramètres.

(c) Paramètres de la couche de convolution : $5^2 \times 3m = 75m$;
Paramètres de la couche pleinement connectée : $(\frac{20-4}{4})^2 mn = 16mn$.
Paramètres de la couche de sortie : $n4$.
Total : $16mn + 75m + 4n$ paramètres.

(d)
- 1ère couche : $5^2 \times 3m_1 = 75m_1$
- 2e couche : $3^2 m_1 m_2 = 9m_1 m_2$
- 3e couche : $\left(\frac{(20-4)/2-2}{2}\right)^2 m_2 n_1 = 9m_2 n_1$
- 4e couche : $n_1 n_2$
- 5e couche : $(n_2 + 1)4$

Total : $75m_1 + 9m_1 m_2 + 9m_2 n_1 + n_1 n_2 + 4n_2 + 4$ paramètres.