TP réseaux de neurones

Implémentation de l'algorithme de perceptron multicouche

Luong Phat NGUYEN

Remarques préliminaires:

— Ce TP est noté.

end

— Il faut rendre : vos codes sources (avec commentaires) + compte rendu, par mail : luong.nguyen@univ-tours.fr

Le but de ce TP est d'implémenter un perceptron multi-couche. Nous allons utiliser l'algorithme de descente de gradient pour l'optimisation.

Algorithme 1 : L'algorithme de perceptron multicouche

```
Input : L-nombre de couches cachées, X-données numériques, Y-Sortie du
                      réseau (une matrice sparse, etc.)
Input: W^l, 1 \le l \le L, matrice de poids.
Input : b^l, 1 \le l \le L, vecteur de poids.
\mathbf{Output}: \mathbf{Paramètres} \ \mathbf{ajust\'{e}s} \ \boldsymbol{W}^l \ \mathrm{et} \ \boldsymbol{b}^l
Initialisation;
for epoch= 1, 2, ..., nombre d'epoch do
        \boldsymbol{a}^0 = \boldsymbol{X}:
        for l = 1, 2, 3, ..., L - 1 do
            \begin{vmatrix} \mathbf{z}^{l} = \mathbf{W}^{l} \mathbf{a}^{l-1} + \mathbf{b}^{l}; \\ \mathbf{a}^{l} = f(\mathbf{z}^{l}); \end{vmatrix}
        \boldsymbol{z}^{L} = \boldsymbol{W}^{L} \boldsymbol{a}^{L-1} + \boldsymbol{b}^{L};
        \hat{m{Y}} = \mathcal{F}(m{z}^L) ;
        Compute J(\hat{\boldsymbol{Y}}, \boldsymbol{Y});
        oldsymbol{g} \leftarrow \nabla_{\hat{oldsymbol{Y}}} J(\hat{oldsymbol{Y}}, \hat{oldsymbol{Y}}) \, ;
        for l = L, L - 1, ..., 2, 1 do
             egin{aligned} oldsymbol{g} &\leftarrow oldsymbol{B}, \ oldsymbol{g} \leftarrow oldsymbol{g} \odot 
abla_{oldsymbol{a}^l} f'(oldsymbol{a}^l), \ si \ l = L, f = \mathcal{F} \ ; \ 
abla_{oldsymbol{b}^l} &= oldsymbol{g} \ ; \ 
abla_{oldsymbol{W}^l} &= oldsymbol{g} oldsymbol{a}^{(l-1)^T} \ ; \ 
oldsymbol{g} &\leftarrow 
abla_{oldsymbol{a}^{l-1}} J = oldsymbol{W}^{l^T} oldsymbol{g} \ ; \end{aligned}
```