GSI-ACAD (2018-2019)
Machine Learning
TP 3 : réseaux de neurones

Remarques préliminaires :

— Ce TP est noté.
— Vos pouvez travailler en monôme ou en binôme.
— Votre travail est à rendre, au plus tard le 27/11 pour le groupe 1 et le 4/12 pour le groupe 2.
— Ce qu'il faut rendre : tous vos codes sources + compte rendu, par mail : `moncef.hidane@insa-cvl.fr`.
— Rappel : retard = pénalité

L'objectif de ce TP est d'implémenter, débugger et tester un perceptron multicouche complètement connecté. L'optimisation se fera par descente de gradient (batch ou mini-batch ou séquentielle). Les dérivées seront calculées par l'algorithme de propagation arrière (backward propagation).

Voici l'algorithme de propagation avant :

Entrée : l, profondeur du reseau
Entrée : $\mathbf{W}^{(i)}, 1 \leq i \leq l$, poids du modèle
Entrée : $\mathbf{b}^{(i)}, 1 \leq i \leq l$, biais du modèle
Entrée : $\mathbf{x}$, entrée du réseau
Sortie : $\mathbf{y}$, sortie du réseau

$\mathbf{h}^{(0)} = \mathbf{x}$
Pour $k = 1, \ldots, l-1$
$\quad \mathbf{a}^{(k)} = \mathbf{b}^{(k)} + \mathbf{W}^{(k)}\mathbf{h}^{(k-1)}$
$\quad \mathbf{h}^{(k)} = f(\mathbf{a}^{(k)})$
Fin pour
$\mathbf{a}^{(l)} = \mathbf{b}^{(l)} + \mathbf{W}^{(l)}\mathbf{h}^{(l-1)}$
$\hat{\mathbf{y}} = \mathcal{F}(\mathbf{a}^{(l)})$
$J = L(\hat{\mathbf{y}}, \mathbf{y}) + \lambda\Omega(\theta)$

Voici l'algorithme de propagation arrière :

$\mathbf{g} \leftarrow \nabla_{\hat{\mathbf{y}}} J = \nabla_{\hat{\mathbf{y}}} L(\hat{\mathbf{y}}, \mathbf{y})$
Pour $k = l, l-1, \ldots, 1$
$\quad \mathbf{g} \leftarrow \nabla_{\mathbf{a}^{(\mathbf{k})}} J = \mathbf{g} \odot f'(\mathbf{a}^{(k)})$
$\quad \nabla_{\mathbf{b}^{(k)}} J = \mathbf{g} + \lambda \nabla_{\mathbf{b}^{(k)}} \Omega(\theta)$
$\quad \nabla_{\mathbf{W}^{(k)}} J = \mathbf{g}\, \mathbf{h}^{(k-1)^{\mathsf{T}}} + \lambda \nabla_{\mathbf{W}^{(k)}} \Omega(\theta)$
$\quad \mathbf{g} \leftarrow \nabla_{\mathbf{h}^{(k-1)}} J = \mathbf{W}^{(\mathbf{k})^{\mathsf{T}}} \mathbf{g}$
Fin pour