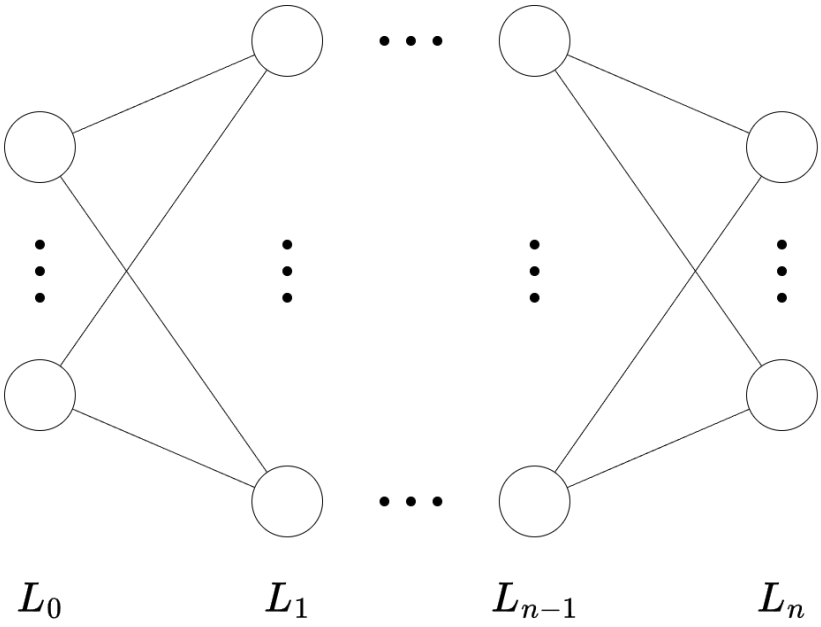
TIPE

Réseau de neurones

Définitions

Soit $n \in \mathbb{N}, n \geqslant 2$.

On peut représenter un réseau de neurones à $n+1$ couches à l'aide du schéma suivant:



Soit $k = 0, \ldots, n$.

On note L_k la k-ième couche du réseau. Cette couche L_k possède $\ell_k \in \mathbb{N}^*$ neurones auxquels on associe un réel appelé activation. On associe alors à la couche L_k un vecteur $X_k \in \mathbb{R}^{\ell_k}$ appelé activation de la couche L_k dont les coordonnées sont les activations des neurones de la couche L_k. Le vecteur X_0 est appelé vecteur d'entrée ou entrée du réseau et le vecteur X_n est appelé vecteur de sortie ou sortie du réseau.

Soit $k = 0, \ldots, n-1$.

Chaque neurone de la couche L_k est relié à chacun des neurones de la couche suivante L_{k+1} par une connection possédant un poids réel.

On peut représenter les connections des neurones de la couche L_k à ceux de la couche L_{k+1} à l'aide d'une matrice $W_k \in \mathcal{M}_{\ell_{k+1},\ell_k}(\mathbb{R})$.

On associe aussi à chaque neurone un réel appelé biais. On associe alors à la couche L_k un vecteur $B_k \in \mathbb{R}^{\ell_k}$ appelé biais de la couche L_k dont les coordonnées sont les biais des neurones de la couche L_k.

Propagation: calcul des activations $X_1, \ldots, X_n$ à partir de X_0

On définit la fonction réelle a telle que: $a : x \in \mathbb{R} \longmapsto \max(x, 0)$.

Pour $p \in \mathbb{N}$ et $X \in \mathbb{R}^p$, on note $a(X)$ le vecteur dont les coordonnées sont les images des coordonnées de X par a.

Pour un vecteur d'entrée $X_0 \in \mathbb{R}^{\ell_0}$ donné, on obtient par récurrence les activations $X_1, \ldots, X_n$ en appliquant pour $k = 1, \ldots, n-1$:

$$X_{k+1} = a(W_k X_k + B_k) \in \mathbb{R}^{\ell_{k+1}}$$

But: entraîner le réseau

Pour un vecteur d'entrée X_0 donné, on cherche à obtenir un vecteur de sortie X_n très "proche" d'un vecteur cible $Y \in \mathbb{R}^{\ell_n}$.

Il est donc nécessaire de disposer d'une fonction permettant de mesurer la distance entre le vecteur de sortie X_n et le vecteur cible Y.

On appelle cette fonction la fonction coût associée au vecteur cible Y et on la note C_Y. Elle est définie comme la distance au carré entre X_n et $Y = (y_i)_{1 \leqslant i \leqslant \ell_n}$ pour le produit scalaire canonique de $\mathbb{R}^{\ell_n}$:

$$C_Y : X = (x_i)_{1 \leqslant i \leqslant \ell_n} \in \mathbb{R}^{\ell_n} \longmapsto ||X - Y||^2 = \sum_{i=1}^{\ell_n} (x_i - y_i)^2$$

Notre but est "d'entraîner" le réseau pour différentes données d'entraînement constituées de couples $(X_0, Y) \in \mathbb{R}^{\ell_0} \times \mathbb{R}^{\ell_n}$. Le but de l'entraînement est de minimiser C_Y pour toutes les données d'entraînement.

Pour entraîner le réseau pour une donnée d'entraînement (X_0, Y), on fixe donc les X_k après avoir déterminé leur valeur par récurrence (voir précédemment) puis on cherche à minimiser C_Y en faisant varier les W_k et les B_k.

Pour cela, on calcule numériquement les gradients de C_Y par rapport à chaque W_k et par rapport à chaque B_k, que l'on note $\nabla_{W_k} C_Y$ et $\nabla_{B_k} C_Y$ respectivement.

On choisit ensuite un petit réel ε et pour $k = 0, \ldots, n-1$ on effectue les opérations:

- $W_k \longleftarrow W_k - \varepsilon \nabla_{W_k} C_Y$
- $B_k \longleftarrow B_k - \varepsilon \nabla_{B_k} C_Y$

Ce calcul de gradient et cette mise à jour des paramètres sont répétés un certain nombre de fois de manière à minimiser au maximum la fonction de coût pour la donnée d'entraînement (X_0, Y).

On répète ce processus appelé "descente de gradient" pour chaque donnée d'entraînement de manière à obtenir des paramètres permettant au réseau de pouvoir donner un vecteur de sortie proche de la sortie attendue pour les entrées présentes dans les données d'entraînement mais aussi pour des entrées qui n'y figurait pas: le réseau a "appris" à généraliser.

Mon problème

J'ai du mal à formaliser le calcul de gradient, dans ce qui suit je risque d'utiliser des termes que ne seront peut être pas adaptés car je ne connais pas très bien le sujet.

Ce que je cherche à faire est trouver une formule qui me permette de calculer numériquement les gradients $\nabla_{W_k} C_Y$ et $\nabla_{B_k} C_Y$ à l'aide de $\nabla_{X_n} C_Y, \nabla_{X_{n-1}} C_Y, \ldots, \nabla_{X_{k+1}} C_Y$ pour pouvoir automatiser ce calcul dans un programme informatique.

En effet, en faisant des recherches j'ai cru comprendre que dans le cas d'une fonction $\mathbb{R}^n \longrightarrow \mathbb{R}^m$, son gradient est une matrice Jacobienne.

Je crois aussi que l'on peut utiliser une sorte de règle de la chaîne pour la Jacobienne d'une composée de fonction:

$$J_{G \circ F}(X) = J_G(F(X)) J_F(X)$$

Je crois aussi avoir compris que lorsque l'on cherche à différentier une fonction $\mathcal{M}_{n,p} \longrightarrow R^m$, il suffit de considérer une fonction $\mathbb{R}^{np} \longrightarrow \mathbb{R}^m$ prenant en entrée un vecteur issu de la concaténation des vecteurs colonnes de la matrice de la fonction initiale (par exemple) et d'en calculer la matrice Jacobienne.

Mais je doute sur ces points et c'est sur ceux-ci que j'aurais besoin d'éclaircissements.

Merci beaucoup pour votre attention !