

**INF8225**

**Hiver 2020**

**TP No. 2**

**1878557 – Bernard Meunier**

**Soumis à :** Christopher Pal

Partie 1 :

A)

Avec :

* Z[] : liste de matrices de valeur pré-activation. Une matrice par couche de neurones Dimension de chaque matrice : *nombre d’Example* \* *neurones par couches*
* a[] : liste de matrices de valeur activées avec un biais. Une matrice par couche de neurones Dimension de chaque matrice : *nombre d’exemple* \* *neurones par couches + 1*.
* deltas[] : liste contenant les matrice de deltas de chaque couche :

Dimension de chaque matrice : *nombre d’exemple* \* *neurones par couches*.

* W[] : liste contenant les matrice de de poids entre les couches de neurones :

Dimension de chaque matrice : *neurone dans la couche précédentes + 1* \* *neurones dans la couche suivante.*

* f ’() : la fonction dérivée de la fonction d’activation de sortie
* h’() : la fonction dérivée de la fonction d’activation des couches cachées.
* Y : matrice des prédictions recherché. Composé d’un vecteur one-hot de la classe de l’exemple.

Dimension de chaque matrice : *nombre d’exemple* X *nombre de classe*.

* lr : constante du taux d’apprentissage
* @ : opération de multiplication de matrice.
* cut() : fonction qui retire la ligne correspondante au biais dans la matrice.

Backpropagation :

1. **deltas[**dernier**]** <= (**Y** – **a[**dernier**]**) \* f ’(**Z[**dernier**]**)
2. Pour chaque matrice **delta** dans **deltas** à partir de l’avant dernier et vers le bas:
   1. **delta** <= cut(**deltas[**suivant**]** @ **W[**suivant**]**) \* h’(**Z[**suivant**]**)
3. Pour chaque matrice **w** dans la matrice de poids **W**:
   1. **gradient** <= ( **a[**avant **w]** @ **deltas[**après **w]** ) / nombre d’exemple dans **a[**avant **w]**
   2. **w** <= **w** + lr \* **gradient**

B)

Implémentation du reseau neuronal multicouche.

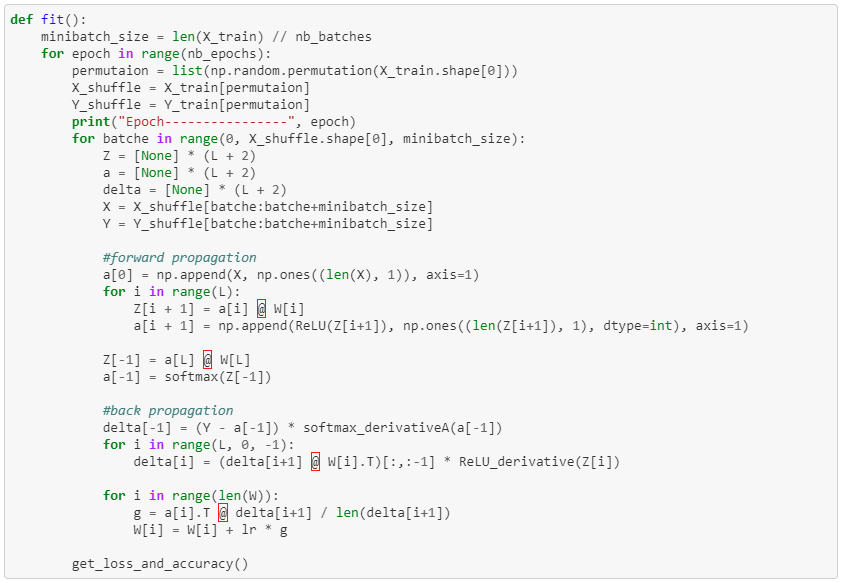
Préparation des données :



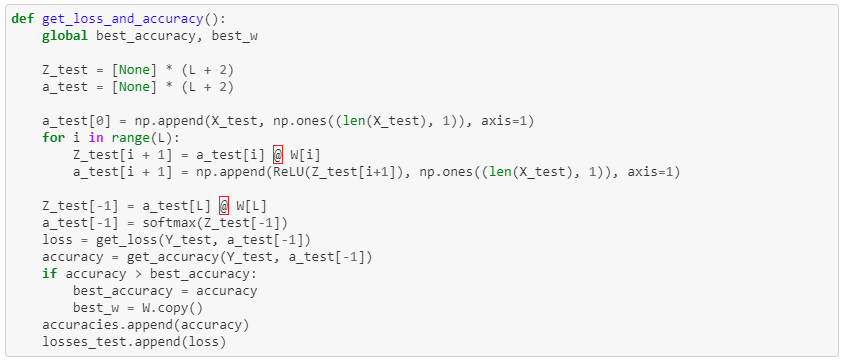
Déclaration de fonction helper :



Code d’entrainement :



Calcule du loss :

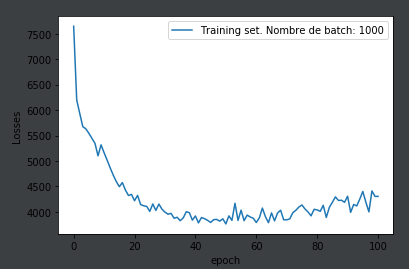


Code d’exécution :



Pour un entrainement avec 1000 mini batch et un taux d’apprentissage de 0.001 :

Le loss est calculer sur 10 000 données de test :



La précision est calculée sur les mêmes 10 000 données de tests :

