Fiche TP1 : Training an MLP from scratch

1- Write the function **init_params(nx, nh, ny)**that initialize the weights of an MLP based on nx, nh, and ny that presents the number of neurons in each layer : input (x), hidden (h), and output (y). All weights must be initialized following the normal distribution with an average of 0 and a standard deviation of 0.3.

2- Write the function **forward(params, X)** that receive a batch of data X with a size of $n_{batch} \times n_x$ and the different parameters in **params.** The function returns $\hat{Y}$ and intermediate values**.**

3- Write the function **loss_accuracy(Yhat, Y)**that returns the loss and accuracy.

4- Write the function **backward**(x,params, outputs, Y)that compute the loss gradients regarding parameters and stock them in a dictionary.

5- Write the function **sgd**(params, grads, eta) that apply mini batch sgd and update the network parameters based on gradients and the learning rate (eta).

6- Write the training steps based on the previous function and display Accuracy and loss graphs.

7- Train the network on the Hand written digit recognition dataset MNIST.

Ps: Use matrix multiplications. Activation functions: Tanh (for hidden layers) and Softmax for the Output. Loss function: Cross entropy.

Parameters: 0.1, 128, 50.

- Load data
- Initialize the Network
- For I = 1…n_{epoach}
  - Random the data
  - For J = 1…N/n_{batch}
    - Load a batch of data
    - Forward Propagation on the batch
    - Compute the loss on the batch
    - Backpropagation on the batch
    - Apply SGD to update parameters

- Plot the accuracy and loss histories.