Deep Learning (IST, 2022-23)

Practical 2: Perceptron

André Martins, Andreas Wichert, Luis Sá-Couto, Margarida Campos

Pen-and-Paper Exercises

The following questions should be solved by hand. You can use, of course, tools for auxiliary numerical computations.

Question 1

Consider the following linearly separable training set:

$$\boldsymbol{x}^{(1)} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \boldsymbol{x}^{(2)} = \begin{bmatrix} 0 \\ 0.25 \end{bmatrix}, \boldsymbol{x}^{(3)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \boldsymbol{x}^{(4)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$y^{(1)} = -1, y^{(2)} = +1, y^{(3)} = +1, y^{(4)} = -1.$$

1. Initialize all weights to zero (including the bias). Assume $\text{sign}(z) = +1$ iff $z \geq 0$, and -1 if $z < 0$. Use a learning rate of one. Compute two epochs of the perceptron learning algorithm.
2. After applying the algorithm until convergence the final trained weights were: $\boldsymbol{w} = [0, 1, 1.75]^\top$. Draw the separation hyperplane.
3. What is the perceptron output for the query point $\begin{bmatrix} 0 & 1 \end{bmatrix}^\top$?
4. If we were to change the initialization of weights and biases to be random with a standard normal distribution $\mathcal{N}(0, 1)$, would it always converge?

Question 2

The perceptron can learn a relatively large number of functions. In this exercise, we focus on simple logical functions.

1. Show graphically that a perceptron can learn the logical NOT function. Give an example with specific weights.
2. Show graphically that a perceptron can learn the logical AND function for two inputs. Give an example with specific weights.
3. Show graphically that a perceptron can learn the logical OR function for two inputs. Give an example with specific weights.
4. Show graphically that a perceptron can not learn the logical XOR function for two inputs.

Programming Exercises

The following exercises should be solved using Python, you can use the corresponding practical's notebook for guidance.

1. Let us consider **Question 1**.
   (a) Initialize all weights to zero (including the bias). Assume $\text{sign}(z) = +1$ iff $z \geq 0$, and -1 if $z < 0$. Use a learning rate of one. Apply the perceptron learning algorithm until convergence. How many epochs does it take to converge?
   (b) Change the initialization of weights and biases to be random with a standard normal distribution $\mathcal{N}(0,1)$. Try multiple times. Does it always converge?

2. Generate a balanced dataset with 30 examples in $\mathbb{R}^2$ and 3 classes. Assume each of the 10 inputs associated to class $k \in \{0,1,2\}$ is generated as $x \sim \mathcal{N}(\mu_k, \sigma_k^2 I)$, with $\sigma_0 = \sigma_1 = \sigma_2 = 1$, $\mu_0 = [0,0]^\top$, $\mu_1 = [0,3]^\top$, and $\mu_2 = [2,2]^\top$. Plot the data.
   (a) Implement the multi-class perceptron algorithm and run 100 iterations. Initialize all the weights to zero and use a learning rate of one. What is the training accuracy (fraction of points that are correctly classified)?

3. Now it's time to try the perceptron on real data and see what happens. Load the UCI handwritten digits dataset using `scikit-learn`
   (a) Randomly split this data into training (80%) and test (20%) partitions.
   (b) Create and run your implementation of the multi-class perceptron algorithm on this dataset. Measure the training and test accuracy.
   (c) Use `scikit-learn`'s implementation of the perceptron algorithm. Compare the resulting accuracies.