INF264 - Exercise 4

Pierre Gillot Natacha Galmiche*

Week 37 - 2021

In this group session you will first get familiar with neural networks and more specifically MLP (Multi-Layer Perceptron) using Tensorflow Playground, then apply the feedforward and backward algorithm by hand on a very simple neural network.

To help you go through the exercises, some key-points of the last lectures are recalled in the appendix.

1 Tensorflow playground

Get familiar with MLP (Multi-Layer Perceptron) by playing around with Tensorflow Playground `http://playground.tensorflow.org/` and then answer the following questions:

1. Preliminary:
   (a) Click multiple times on the "reset the network" button. What does this button do?
   (b) Sometimes if you reinitialize and re-train a model with exactly the same data and exactly the same configuration it converges to different weights values. Why?
   (c) What could happen if your learning rate is too low?
   (d) What could happen if your learning rate is too high?
   (e) If your learning rate is a bit high and your data noisy, would increasing the batch size improve or reduce the generalization performance of your model?

2. Look at Figure 1, and assume X_1, X_2 and Y are in $\mathbb{R}$.:
   (a) What are the dimensions of the hidden units $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$?
   (b) What are the dimensions of the weights $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}$ and $\mathbf{W}^{(3)}$?
   (c) Assume all activation functions are the identity function. Recalling $\mathbf{H}^{(1)} = \mathbf{W}^{(1)}\mathbf{X}$ where $\mathbf{X} = (X_1, X_2) \in \mathbb{R}^2$, express Y using only $\mathbf{W}^{(1)}$, $\mathbf{W}^{(2)}$, $\mathbf{W}^{(3)}$ and $\mathbf{X}$.
   (d) What is the type of the relation between Y and $\mathbf{X}$ in question 2.(c)?

*Not a TA for this course this year.

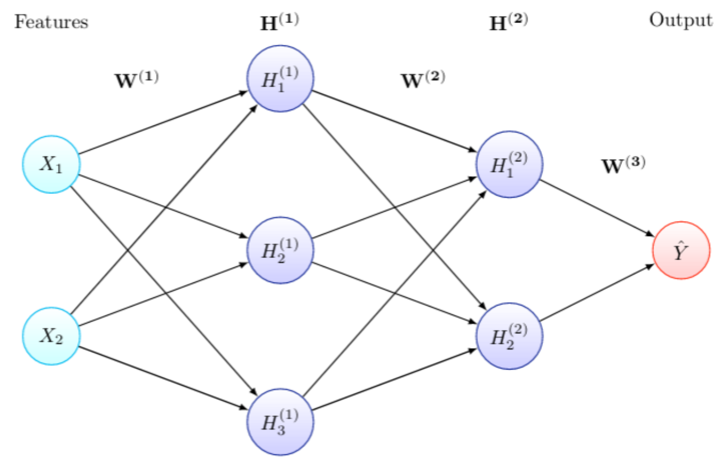


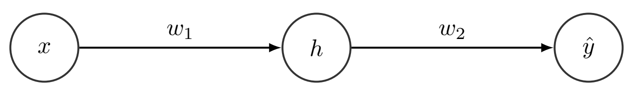
Figure 1: An example of a MLP with 2 hidden layers

3. MLP for regression:
   (a) Click on **this link** to get a specific model configuration or look at the figure in the appendix.
   (b) Train this model. Is this model suitable for this problem? Why?
   (c) Change the activation function of this model to 'Sigmoid' and observe the results.

4. MLP for classification:
   (a) What can you say if your test accuracy is close to 0.5?
   (b) Select the "exclusive or" dataset. If you had to select only one feature for this problem, which one would you choose?
   (c) Should you use a neural network to solve this problem using this feature?

2 Neural networks

Consider the following simple neural network:



We have a one-dimensional input $x \in \mathbb{R}$ and a one-dimensional target $y \in \mathbb{R}$. Furthermore, we have one hidden layer consisting of one neuron and ReLU activation function. The output layer is linear.

That is, we have $z = w_1 x$, $h = f(z)$ where $f(z) = \max(0, z)$ and $\hat{y} = w_2 h$. We consider squared loss $L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$.

Suppose that initial weights have values $w_1 = 3$ and $w_2 = 2$. We have observed one data point with $x = 1$ and $y = 5$.

1. Perform one update of parameters w_1 and w_2 using gradient descent with learning rate $\gamma = 0.1$

Hint: $f'(z) = 1$ when $z > 0$, $f'(z) = 0$ when $z < 0$ and undefined when $z = 0$.

APPENDIX

Tensorflow playground configuration

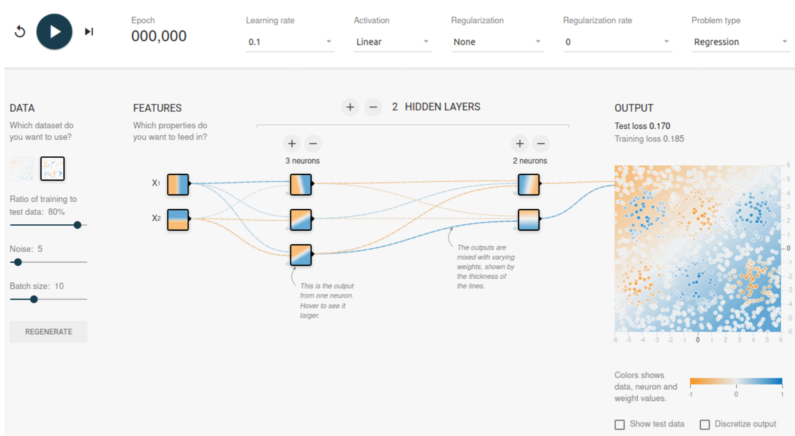


Figure 2: An example of a MLP configuration using Tensorflow playground

Reminders

This section is based on the lectures slides for more details directly look at the slides.

Gradient-based learning

Given a loss function $L(\theta)$ where θ are the parameters of your neural network

- The goal is to minimize the training loss by selecting appropriate values for the weights:

$$\min_{\theta} L(\theta).$$

- Update weights using gradient descent:

$$\theta_{t+1} = \theta_t - \gamma_t \nabla L(\theta) \qquad \text{where } \gamma_t \text{ is the learning rate.}$$

- The objective function is not convex, thus gradient descent converges towards a local minimum.

Chain rule of calculus

- Formula for computing the derivative of the composition of two or more functions:

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial x}.$$

Backpropagation

- First compute the gradient for the last set of weights.
- Then propagate the error to the previous layer and compute the gradient for the next set of weights.
- Repeat this process until you have processed the first set of weights (all weights have now been updated).