Introduction to Neural Networks

The Perceptron

M. Alexander Perez

2021

Galileo University – BiomedLab

Outline

Introduction to Neural Networks

Pitt Neuron

Neural Network

How should we change the weights?

Introduction to Neural Networks

Objective

Our grand objective is the idea of strong AI. Create a simulation of a single neuron, a biological one, and join 10^{11} of them and get something like the human brain. Of course this is an objective from which we are very far away, at least for now (2021).

Our Model

But many scientist got inspired by the human brain and the idea of the neuron. One of the great principles that many neuroscience help to discover is that of plasticity. We can say that the brain doesn't build new neurons; but changes the connection between the neurons to learn something.

Hebb's Rule

Another basic principle from the human brain is the Hebb's Rule, although I think Freud write something about this before. Hebb's rule is simple: "Neurons that fire together wire together." If you receive an stimulus like a song, when something sad happens, both set of neurons fire together. The ones that make you feel bad and the ones that process the sound of the song. Next time you listen to the song you get sad.

Simple Model: McCullochg and Pitts Neuron

A simple neuron model is based on the following:

- $x_1, x_2, \ldots, x_n$ the set of inputs.
- $w_1, w_2, \ldots, w_n$ the set of weights.
- $h(t)$ an activation function, this decides if the neuron fires or not for a given input.

The output of the neuron is modeled by:

$$y = h(w_1 \, x_1 + w_2 \, x_2 + \cdots + x_n \, w_n).$$

Originally the Heaviside function was used as an activation function:

$$h(t) = \begin{cases} 1 & t > 0 \\ 0 & t \leq 0 \end{cases}.$$

Basic Scheme of Neural Networks

As you might imagined a single neuron is not that interesting. So we use many of them. The next step is how we make them learn something (recall the basics expectations of learning). Our approach is the one of supervised learning:

Change the weights of the neurons so the output is more like the one we want.

The first type of neural network is a single layer neural network called Perceptron.

Examples

In a supervised learning problem, in particular classification, we have:

- A set of examples: (x_i, t_i).
  - $\mathrm{x}_i = (x_i^1, x_i^2, \ldots, x_i^m)$ is the feature vector.
  - t_i is the target value, usually encoded in a one hot vector i.e.:

$$t_i = (t_i^1, t_i^2, \ldots, t_i^n).$$

  Here if the example x_i is of the class k then $t_i^k = 1$ and $t_j^k = 0$ for $j \neq k$.

The Neural Network

- A neural network, that is a set of neurons in the case of the perceptron n neurons, one for each input.
  - What characterizes the neural network is the weights $w_i j$.
  - The output is a vector that we calculate as:

$$\mathrm{y} = \mathrm{h}\left(w\,\mathrm{x}\right).$$

The Loss Function

- Our objective is to make the y equal to the t so we make a change proportional to the error:

$$\Delta w_{ij} = -(y^i - t^i)\, x^j .$$

The Update Algorithm

- We make several iterations altering the weights, trying to get a better approximation, at each iteration we do:

  $$w_{ij} \leftarrow w_{ij} - \alpha(y^i - t^i)\, x^j.$$

  The value α is called the learning rate and it controls how much will change the weights in each iteration. We will see his effect on a notebook.

Implementations

- Numpy
- Torch
- Tensorflow