Artificial intelligence
University of Rennes 1
ESIR
2020/21

Backpropagation Let us consider the neural network on Figure 1.

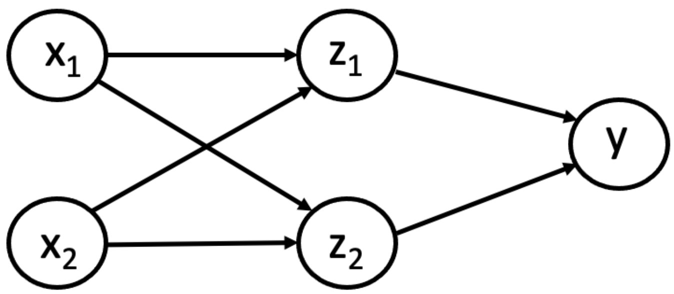

Figure 1: Neural network

In this example we try to understand using a simple numerical example, how the gradient descent algorithm works in the context of neural networks and how the network itself can compute iteratively the derivatives of the cost function.

- Let us assume that our training set contains a single example, with input $x_1 = 2$, $x_2 = 3$ and output $y = 1$. For simplicity, let us assume that the activation function is the linear function.
- Let us initialize the weights θ of the network as follows:
  $\theta^1_{01} = 0$, $\theta^1_{11} = 0.11$, $\theta^1_{21} = 0.21$, $\theta^1_{12} = 0.12$, $\theta^1_{22} = 0.08$,
  $\theta^2_{01} = 0$, $\theta^2_{11} = 0.14$, $\theta^2_{21} = 0.15$
- Compute the output of the network on the training set. Also, express the predicted value (the output of the network) as a single formula of the input and weights.
- Let us assume that we use the following loss function: $J(\theta) = \frac{1}{2}(y_{predicted} - y_{traininig})^2$
- Let us assume that the learning rate $\alpha = 0.05$ and we realize a gradient descent algorithm to learn the weights of the network. Let us now try to understand the update of the weight θ^2_{21}, in the first iteration of the algorithm. Recall, that we compute it as

  $$\theta^2_{21} \leftarrow \theta^2_{21} - \alpha \frac{\partial J(\theta)}{\partial \theta^2_{21}}$$

  Compute the partial derivative $\frac{\partial J(\theta)}{\partial \theta^2_{21}}$ using the expression (question above)

- Let $\Delta = y_{predicted} - y_{traininig}$. Express the partial derivative $\frac{\partial J(\theta)}{\partial \theta^2_{21}}$ now using Δ and the value of z_2 (that we can remember if we do a forward calculation, with the training input)
- Compute also the partial derivative $\frac{\partial J(\theta)}{\partial \theta^2_{11}}$.
- Compute now the partial derivative $\frac{\partial J(\theta)}{\partial \theta^1_{11}}$ (that is, let us try to understand how the error function depends on a weight that is between a node in the input layer and another node in the hidden layer).
- Express the updates of θ^1_{11} in the gradient descent, with the help of θ^2_{11} and Δ.
- Express the updates of the other weights (between the input and hidden layer) in the same way.
- Explain how can we realize an iteration of the gradient descent through forward and backward computation in a neural network.