INF264 - Exercise 4 - Solutions

Natacha Galmiche

1 Tensorflow playground

1. Preliminary:
   (a) This button re-initializes the weights.
   (b) Different weights initializations can result in different weights final values. Indeed, since the objective function is not convex the algorithm that solves the loss function minimization problem (such as a gradient descent method) can converge towards a local minimum
   (c) If the learning rate is too low, then more epochs can be necessary to obtain the same performance (the learning is slower) but more importantly, the gradient descent algorithm can be more likely stuck in a local minimum close to the initial weights values
   (d) If the learning rate is too high the gradient descent algorithm might not converge and is more sensitive to last inputs' noise.
   (e) Increasing the batch size can help improve generalization performance by attenuating the sensitivity to last inputs' noise.

2. Assuming X_1, X_2 and Y are in $\mathbb{R}$.:
   (a) $\mathbf{H}^{(1)} \in \mathbb{R}^3$ and $\mathbf{H}^{(2)} \in \mathbb{R}^2$.
   (b) $\mathbf{W}^{(1)} \in \mathbb{R}^{3\times 2}$, $\mathbf{W}^{(2)} \in \mathbb{R}^{2\times 3}$ and $\mathbf{W}^{(3)} \in \mathbb{R}^{1\times 2}$.
   (c) $Y = \mathbf{W}^{(3)}\mathbf{W}^{(2)}\mathbf{W}^{(1)}\mathbf{X}$.
   (d) Linear, because there exists a matrix $\mathbf{W} = \mathbf{W}^{(3)}\mathbf{W}^{(2)}\mathbf{W}^{(1)}$ such that $Y = \mathbf{W}\mathbf{X}$

3. MLP for regression:
   (a) Click on **this link** to get a specific model configuration or look at the figure in the appendix.
   (b) No, this model is linear whereas the data is not
   (c) Now that we use a non linear activation function our model is able to fit to the data. We can draw a parallel between this and the non-linear basis functions we use in linear regresion

4. MLP for classification:
   (a) Since there are only 2 classes and since there is the same number of datapoints in each class, if you assume that all the datapoints are blue for instance then you get a 0.5 accuracy. So it we expect our neural network to perform better than this extremely basic model.
   (b) The X_1X_2 feature matches perfectly this problem
   (c) If we already know that there is a feature matching perfectly our data then there is no need for a neural network...

2 Neural networks

Forward pass

$$\begin{aligned} z &= w_1 x = 3 \cdot 1 = 3 \\ h &= f(z) = \max(0, 3) = 3 \\ \hat{y} &= w_2 h = 2 \cdot 3 = 6 \end{aligned}$$

Backward pass We are going to need the following derivatives:

$$\begin{aligned}
\frac{\partial L}{\partial \hat{y}} &= -(y - \hat{y}) = \hat{y} - y \\
\frac{\partial \hat{y}}{\partial w_2} &= h \\
\frac{\partial \hat{y}}{\partial h} &= w_2 \\
\frac{\partial h}{\partial z} &= \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{if } z \leq 0 \end{cases} \\
\frac{\partial z}{\partial w_1} &= x
\end{aligned}$$

The compute the gradient, we use backpropagation. Using chain rule, we get the following partial derivatives:

$$\begin{aligned}
\frac{\partial L}{\partial w_1} &= \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h} \frac{\partial h}{\partial z} \frac{\partial z}{\partial w_1} \\
&= (\hat{y} - y) \cdot w_2 \cdot 1 \cdot x \\
&= (6 - 5) \cdot 2 \cdot 1 \cdot 1 \\
&= 2
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial L}{\partial w_2} &= \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2} \\
&= (y - \hat{y}) \cdot h \\
&= (6 - 5) \cdot 3 \\
&= 3
\end{aligned}$$

Updates with gradient descent:

$$\begin{aligned}
w_1 &\leftarrow w_1 - \gamma \frac{L}{\partial w_1} \\
&= 3 - 0.1 \cdot 2 \\
&= 2.8
\end{aligned}$$

$$\begin{aligned}
w_2 &\leftarrow w_2 - \gamma \frac{L}{\partial w_2} \\
&= 2 - 0.1 \cdot 3 \\
&= 1.7
\end{aligned}$$