ASSIGNMENT 1 : Introduction to Neural Network

Détagnon Bernic GBAGUIDI

March 2, 2024

Problem 1: Simulate a boolean function

Let write a neural network NN with four inputs and one output such that

$$NN(x_1, x_2, x_3, x_4) = 1 + x_1x_2 + x_2x_3 + x_3x_4$$

for every $x_1, x_2, x_3, x_4 \in \{0, 1\}$

As we are dealing with four inputs we will have 16 rows in the truth table what become large . The following python code generate the truth table corresponding to our boolean function.

```
from itertools import product

# Define the variables
variables = ['x1', 'x2', 'x3', 'x4']

# Generate all possible combinations of truth values for the
    variables
truth_table = list(product([False, True], repeat=len(variables)))

# Print the header
header = '\t'.join(variables + ['NN(x1,x2,x3,x4)'])
print(header)

# Print the truth values
for row in truth_table:
# Evaluate your logical expression here, for example: x and y or
    (z and not t)
expression_result =
    bool((1+row[0]*row[1]+row[1]*row[2]+row[2]*row[3])%2)

# Print the truth values and the result of the expression
row_str = '\t'.join([str(value) for value in row] +
    [str(expression_result)])
print(row_str)
```

We will obtain the following table,considering that in the table "F" stand for "0" and "T" for "1" .

x_1	x_2	x_3	x_4	$NN(x_1, x_2, x_3, x_4)$
F	F	F	F	T
F	F	F	T	T
F	F	T	F	T
F	F	T	T	F
F	T	F	F	T
F	T	F	T	T
F	T	T	F	F
F	T	T	T	T
T	F	F	F	T
T	F	F	T	T
T	F	T	F	T
T	F	T	T	F
T	T	F	F	F
T	T	F	T	F
T	T	T	F	T
T	T	T	T	F

Table 1: Truth table.

From the table we can come with the Conjunctive Normal Form of the boolean function:

$$\begin{aligned} NN(x_1, x_2, x_3, x_4) =& (x_1 \vee x_2 \vee \rceil x_3 \vee \rceil x_4) \wedge (x_1 \vee \rceil x_2 \vee \rceil x_3 \vee x_4) \\ & \wedge (\rceil x_1 \vee x_2 \vee \rceil x_3 \vee \rceil x_4) \wedge (\rceil x_1 \vee \rceil x_2 \vee x_3 \vee x_4) \\ & \wedge (\rceil x_1 \vee \rceil x_2 \vee x_3 \vee \rceil x_4) \wedge (\rceil x_1 \vee \rceil x_2 \vee \rceil x_3 \vee \rceil x_4) \end{aligned}$$

With this form we can write the boolean circuit and from that generate the neural network equivalent. But here I directly go to the neural network.

Note that I wrote the Conjunctive Normal form not the Disjunctive normal form which we use to do in class. With this in the boolean circuit we will have a range of "OR" gates instead of "AND". I make this choice because in the truth table "F" are fewer that "T".

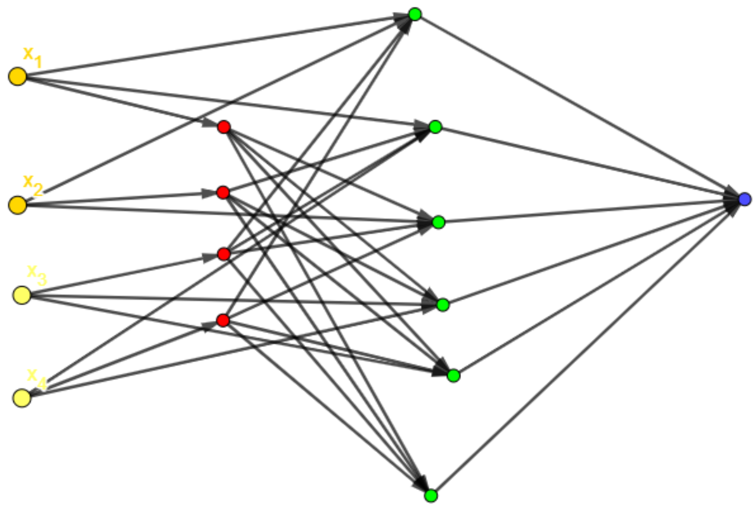

Figure 1: Neural Network

Let give explain the neural network.

- In yellow we have the inputs layers .
- In red we have a layers which neurons correspond to NOT in the boolean circuit.
- In green we have a layer which neurons correspond to OR in the boolean circuit.
- In purple we have a layer which neurons correspond to AND in the boolean circuit.
- Every arrow that targets a red neuron has weight -1 and biais 0.5 and the neurons use step function as activation function.
- Every arrow that targets a green neuron has weight 2 and biais -1 and the neurons use step function as activation function.
- Every arrow that targets a purple neuron has weight 2 and biais -3 and the neurons use step function as activation function.

Problem2: Equivalence of activations

a) Let for every $\epsilon > 0$, describe a neural

network NN with one input, one output and sigmoid activations such that

$$|NN(x) - H(x)| < \epsilon$$

for every x such that $|x| > \epsilon$

This in another words is by referring to definition of limit to find a neural network such that :

$$\lim_{x \longrightarrow +\infty} NN(x) = 1$$

and

$$\lim_{x \longrightarrow -\infty} NN(x) = 0$$

Let suppose our neural network has one input , no hidden layer but one output with sigmoid activation function.

So we can write $NN(x) = S(wx + b) = \dfrac{1}{1 + e^{-(wx+b)}}$.

We can see that the relation :

$$\lim_{x \longrightarrow +\infty} NN(x) = 1$$

and

$$\lim_{x \longrightarrow -\infty} NN(x) = 0$$

is satisfy if we choose $w > 0$ and $b \in \mathbb{R}$.

i.e $\forall \alpha > 0$ there exist $\delta_1 > 0$ such that for every $x > \delta_1$ we have

$|NN(x) - 1| < \alpha$ and there exist $\delta_2 > 0$ such that for every $x < -\delta_2$ we have $|NN(x) - 0| < \alpha$.

So for $\delta = max(\delta_1, \delta_2)$ we can say that $\forall \alpha > 0$ there exist $\delta > 0$ such that for every $|x| > \delta$ we have $|NN(x) - H(x)| < \epsilon$.

But choosing $\epsilon = max(\delta, \alpha)$ we will finally say

$\forall \epsilon > 0$ if $|x| > \epsilon$ we have $|NN(x) - H(x)| < \alpha$

With this we can conclude of the choice of a neural network $NN(x) = S(wx + b) = \dfrac{1}{1 + e^{-(wx+b)}}$ with $w > 0$ and $b \in \mathbb{R}$

b) Let describe a neural network NN with one input, one output, using step and identity activations, such that for every -1 $\leq$ x $\leq$ 1

$$|NN(x) - f(x)| < \epsilon$$

In order word we want to describe a neural network that approximate f on $[-1, 1]$.

For this purpose let suppose that with an error ϵ we can say that the value of f on $[-1, 1]$ is $W \in \mathbb{R}$

i.e. $|W - f(x)| \leq \epsilon$ for $x \in [-1, 1]$

With this let consider the neural network NN in the figure below.

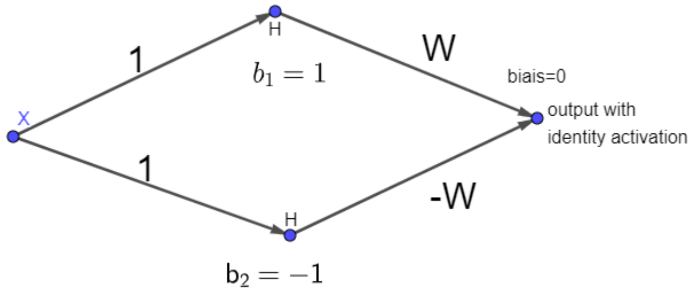

Figure 2:

We have $NN(x) = W[H(x - (-1)) - H(x - 1))]$.

As $[H(x - (-1)) - H(x - 1))] = \mathbb{1}(-1 \leq x \leq 1)$ then

$NN(x) = W\mathbb{1}(-1 \leq x \leq 1)$.

We can then conclude that for $x \in [-1, 1]$, $|NN(x) - W| \leq \epsilon$

Problem3: Gradients for one neuron

Let $x, w \in \mathbb{R}^2, b \in \mathbb{R}$

a) Let $f_R(x) = ReLU(w^T x + b)$ be a ReLU neuron and $Csq(y, \bar{y}) = (y - \bar{y})^2$ denote the square loss for $\bar{y} \in \mathbb{R}$.

Let calculate $\dfrac{\partial Csq(f_R(x), \bar{y})}{\partial w_i}$

$$
\begin{aligned}
\frac{\partial Csq(f_R(x),\bar{y})}{\partial w_i} =& \frac{\partial}{\partial w_i}(f_R(x)-\bar{y})^2\\
=& \frac{\partial}{\partial w_i}(ReLU(w^Tx+b)-\bar{y})^2\\
=& \frac{\partial}{\partial w_i}((w^Tx+b))\frac{\partial}{\partial(w^Tx+b)}(ReLU(w^Tx+b)-\bar{y})^2\\
=& x_i \times 2 \times ReLU'(w^Tx+b)(ReLU(w^Tx+b)-\bar{y})\\
=& 2x_i(ReLU(w^Tx+b)-\bar{y}) \times \begin{cases} 1 & \text{if } w^Tx+b>0\\ 0 & \text{if } w^Tx+b<0 \end{cases}
\end{aligned}
$$

b) Let compute $\frac{\partial Cross(f_s(x),\bar{y})}{\partial w_i}$

$$
\begin{aligned}
\frac{\partial Cross(f_s(x),\bar{y})}{\partial w_i} =& \frac{\partial}{\partial w_i}(-\bar{y}\ln(f_S(x))-(1-\bar{y})\ln(1-f_S(x)))\\
=& \frac{\partial}{\partial w_i}(-\bar{y}\ln(S(w^T+b))-(1-\bar{y})\ln(1-S(w^T+b)))\\
=& (-\bar{y}\frac{\partial(w^T+b)}{\partial w_i}\frac{\partial \ln(S(w^T+b)}{\partial w^T+b})\\
&-(1-\bar{y})\frac{\partial(w^T+b)}{\partial w_i}\frac{\partial \ln(1-S(w^T+b))}{\partial w^T+b})\\
=& -\bar{y}x_iS'(w^T+b)\frac{1}{S(w^T+b)}\\
&+(1-\bar{y})x_iS'(w^T+b)\frac{1}{1-S(w^T+b)}\\
=& -\bar{y}x_iS(w^T+b)(1-S(w^T+b))\frac{1}{S(w^T+b)}\\
&+(1-\bar{y})x_iS(w^T+b)(1-S(w^T+b))\frac{1}{1-S(w^T+b)}\\
=& -\bar{y}x_i(1-S(w^T+b))+(1-\bar{y})x_iS(w^T+b)\\
=& x_i(S(w^T+b)-\bar{y})
\end{aligned}
$$

c) Let compute $\frac{\partial C_{hinge}(f_T(x),\bar{y})}{\partial w_i}$

$$
\begin{aligned}
\frac{\partial C_{hinge}(f_T(x),\bar{y})}{\partial w_i} &= \frac{\partial(max(0, 1-\bar{y}\tanh(w^T+b)))}{\partial w_i} \\
&= \begin{cases} \dfrac{\partial(1-\bar{y}\tanh(w^T+b))}{\partial w_i} & \text{if } w^Tx+b>0 \\ 0 & \text{if } w^Tx+b<0 \end{cases} \\
&= \begin{cases} -x_i \times \bar{y}(1-\tanh^2(w^T+b)) & \text{if } w^Tx+b>0 \\ 0 & \text{if } w^Tx+b<0 \end{cases} \\
&\text{Because } \tanh'(x) = (1-\tanh^2(x))
\end{aligned}
$$

Problem 4: Accuracy at Initialization

Let assume there is a dataset with M points and labels $x^{(i)} \in \mathbb{R}^d, y^{(i)} \in \{0,1,2,3\}$ for $i = 1, ..., M$. What accuracy can we expect for this dataset at initialization?

At initialization the weights are set randomly so the output is a random guessing . The accuracy is therefore the probability to output the correct label among the four . Knowing that we just have one correct output this probability is then $\frac{1}{4} = 0.25$

The accuracy at initialization is 0.25.