Student Name: Kevin Song
Student ID: 21042881

**Comp4211**
**Assignment 1**

---

## Question 1

Our perceptron will have weights of $[-1, 1, 1]$ for inputs $a, b, c$ respectively, a bias term of $\tau = 1.5$, and the activation function will be:

$$O = \text{step}(\sum_{i=1}^{n} I_i w_i + \tau)$$

Where $I_i$ is the $i$th input, and $n = 3$, and $step(x)$ returns 1 if $x \geq 0$, and 0 otherwise.

Recall that our expression is:

$$\neg a \vee b \neg c$$

Observe the truth table for our expression:

| $\neg a \vee b \neg c$ | a | b | c |
|---|---|---|---|
| T | T | F | F |
| T | F | T | F |
| T | F | F | T |
| T | T | T | F |
| F | T | F | T |
| T | T | T | T |
| T | F | F | F |

We note that our output, when expanded, is as follows:

$$O = \text{step}(-a + b - c + \tau)$$

$$O = \text{step}(-a + b - c + 1.5)$$

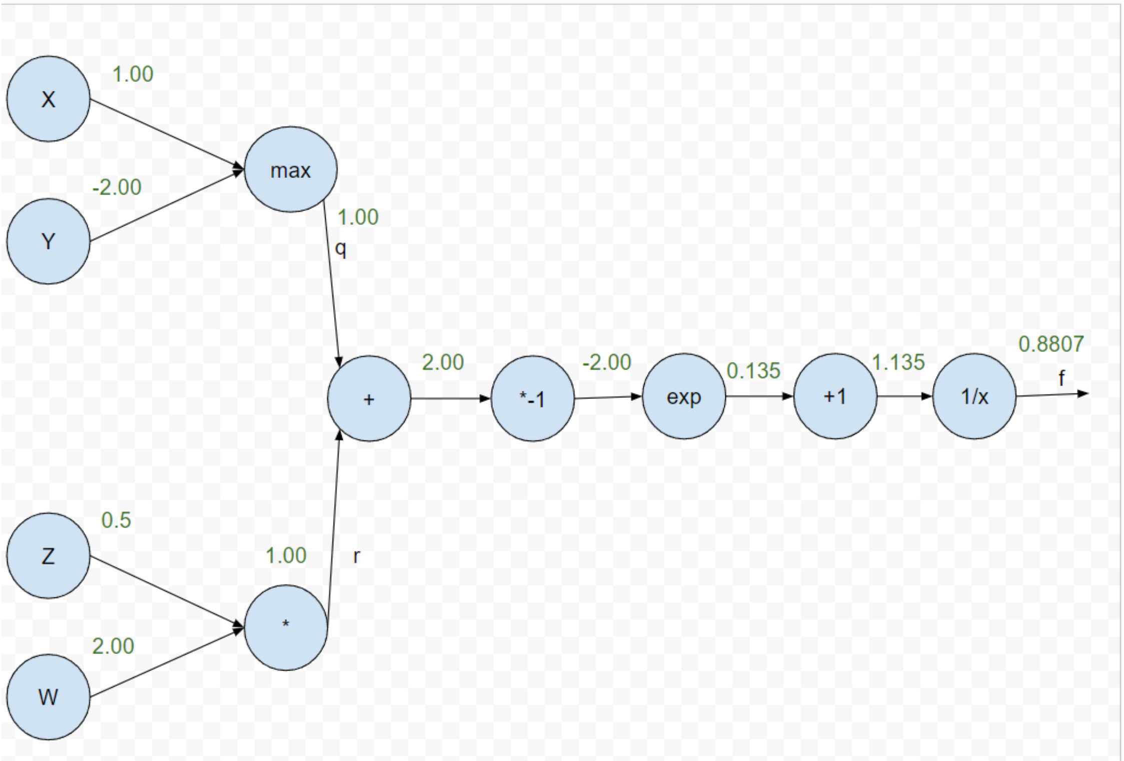
As $a = 0$ if $a$ is false, and $a = 1$ if $a$ evaluates to true, then we can insert this back into the table and show that they are logically equivalent:

| $\neg a \vee b \neg c$ | a | b | c | (-a + b - c + 1.5) | O |
|---|---|---|---|---|---|
| T | T | F | F | 0.5 | T |
| T | F | T | F | 2.5 | T |
| T | F | F | T | 0.5 | T |
| T | T | T | F | 1.5 | T |
| F | T | F | T | -0.5 | F |
| T | T | T | T | 1.5 | T |
| T | F | F | F | 1.5 | T |

Thus, our perceptron has correctly modeled the boolean expression $\neg a \vee b \vee \neg c$.

## Question 2

We begin by computing the actual output for this set of parameters, and drawing its associated computational graph:

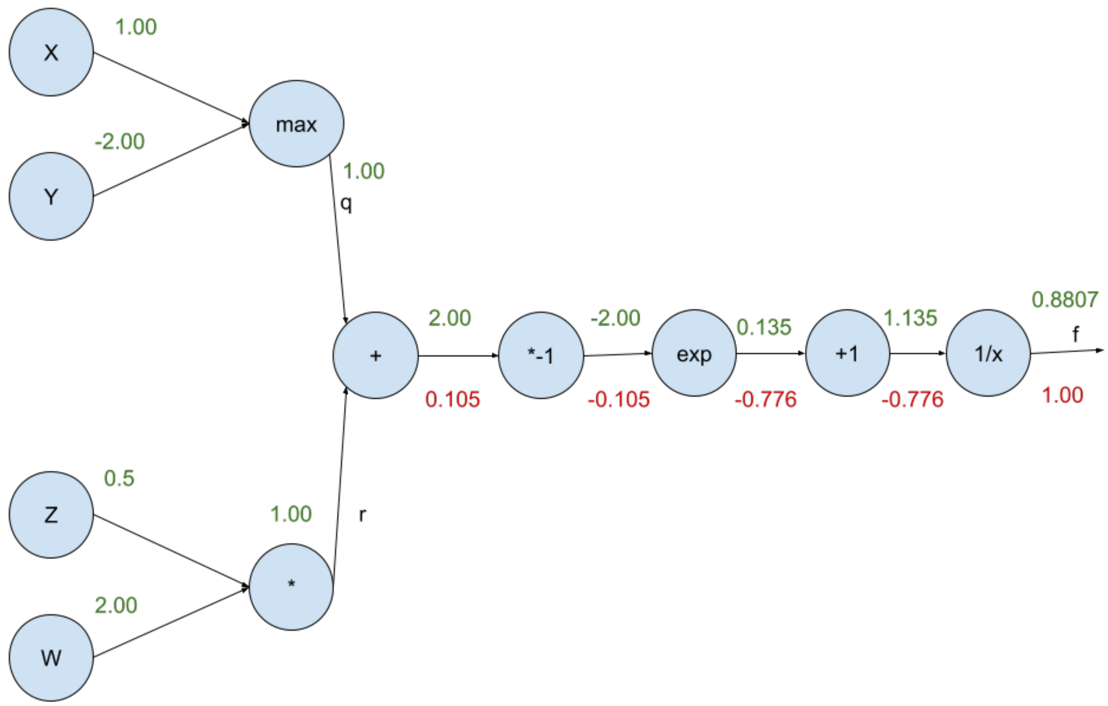



Where $f$ is the output of the function, respectively.

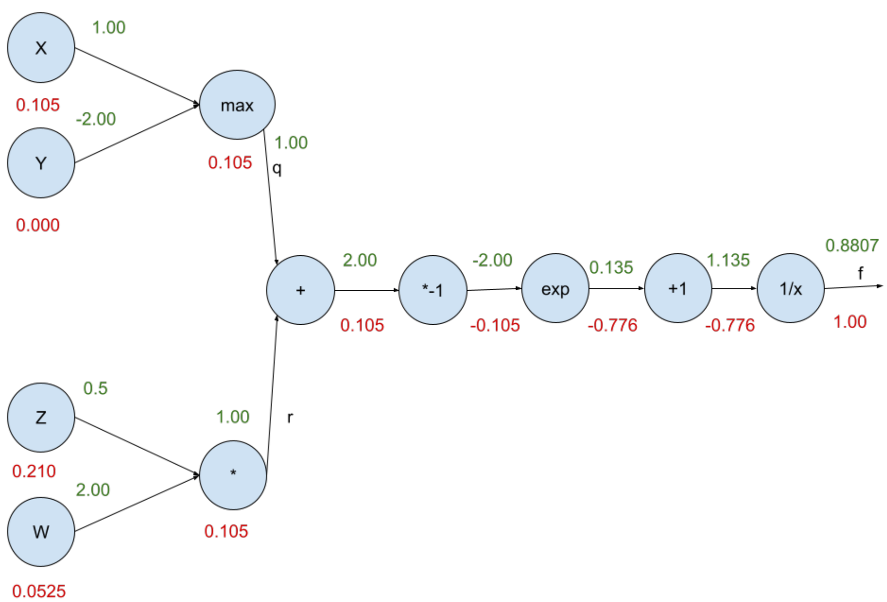
To make notation easier, we will say that each of the outputs between nodes of the sigmoid gate is also $f$. So in order of right to left, we have:

1. $f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = \frac{-1}{x^2}$
2. $f(x) = a + x \rightarrow \frac{df}{dx} = 1$
3. $f(x) = e^x \rightarrow \frac{df}{dx} = e^x$
4. $f(x) = ax \rightarrow \frac{df}{dx} = a$

As a result, we obtain the following graph, with the backwards gradients marked in red:




Finally, we compute the remaining gradients:



## Question 3