1. (2 Punkte) Ein Netz gibt für drei Bilder die Wahrscheinlichkeit $\hat{y}$ für die Klasse 1 aus. Berechne jeweils den Fehler $L = -\big(y \ln(\hat{y}) + (1-y)\ln(1-\hat{y})\big)$. Bei welchem Bild liegt das Netz am meisten daneben?
   (a) $\hat{y} = 0.8$, $y = 1$
   (b) $\hat{y} = 0.8$, $y = 0$
   (c) $\hat{y} = 0.3$, $y = 0$

2. (4 Punkte) (a) Zeige, dass für $\sigma(z) = \dfrac{1}{1+e^{-z}}$ gilt: $\sigma'(z) = \sigma(z)\big(1-\sigma(z)\big)$.
   Tipp: Schreibe $\sigma(z) = (1+e^{-z})^{-1}$ und benutze die Kettenregel.
   (b) Für den Fehler gilt $\frac{\partial L}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$ mit $\hat{y} = \sigma(z)$. Zeige, dass $\frac{\partial L}{\partial z} = \hat{y} - y$ ist, getrennt für $y = 1$ und $y = 0$.
   (c) Wo ist $\sigma'(z)$ am größten und wie groß ist es dort?

3. (3 Punkte) Gegeben ist $f(x_1, x_2, x_3) = (x_1^2 + 3x_2) \cdot 2x_3$ an der Stelle $x_1 = 2$, $x_2 = -1$, $x_3 = 3$.
   (a) Zeichne den Rechengraphen mit den Zwischenergebnissen $u = x_1^2$, $v = 3x_2$, $s = u + v$, $t = 2x_3$ und $f = s \cdot t$. Trage die Werte vorwärts und die Ableitungen von f rückwärts ein.
   (b) Gehe mit $lr = 0.01$ einen Schritt gegen die Ableitungen: $x_i^{neu} = x_i - lr \cdot \frac{\partial f}{\partial x_i}$. Wie groß ist f an der neuen Stelle?

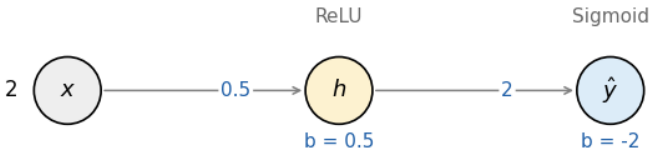
4. (3 Punkte) Gegeben ist $f(x_1, x_2, x_3) = (x_1 + 2x_2)(x_2 + 4x_3)$ an der Stelle $x_1 = -1$, $x_2 = 2$, $x_3 = 3$.
   (a) Zeichne den Rechengraphen mit $p = x_1 + 2x_2$, $q = x_2 + 4x_3$ und $f = p \cdot q$. Trage die Werte vorwärts und die Ableitungen rückwärts ein.
   (b) Bestimme $\frac{\partial f}{\partial x_1}$, $\frac{\partial f}{\partial x_2}$ und $\frac{\partial f}{\partial x_3}$. Was ist bei x_2 anders als bei den anderen beiden?

5. (3 Punkte) Ein Neuron mit einem Input, Sigmoid und Binary Cross-Entropy hat die Parameter $w = 1$, $b = 0.5$. Das Trainingsbeispiel ist $x = -1$, $y = 0$, die Lernrate $lr = 1$.
   (a) Berechne z, $\hat{y}$ und L.
   (b) Berechne $\delta = \hat{y} - y$, $\frac{\partial L}{\partial w}$ und $\frac{\partial L}{\partial b}$.
   (c) Berechne die neuen Parameter und damit noch einmal $\hat{y}$ und L. Ist das Netz besser geworden?

6. (4 Punkte) Ein Neuron mit zwei Inputs, Sigmoid und Binary Cross-Entropy hat die Parameter $w_1 = 0.5$, $w_2 = -1$, $b = 1$. Das Trainingsbeispiel ist $x_1 = 2$, $x_2 = 1$, $y = 0$, die Lernrate $lr = 0.5$.
   (a) Berechne z, $\hat{y}$ und L.
   (b) Berechne δ und die Ableitungen nach w_1, w_2 und b.
   (c) Berechne die neuen Parameter und damit $\hat{y}$ und L.
   (d) Warum ändert sich w_1 stärker als w_2?

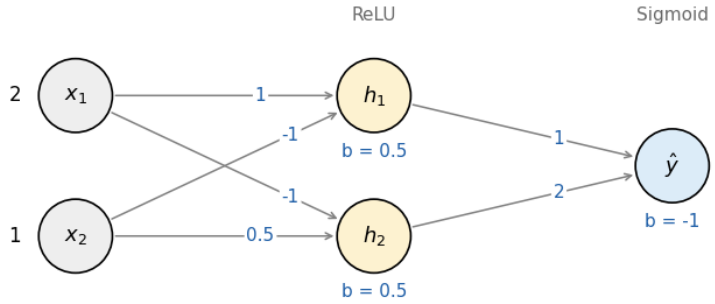
7. (3 Punkte) Ein Neuron mit einem Input und Sigmoid soll aus zwei Trainingsbeispielen lernen: $x = 1$ mit $y = 1$ und $x = -1$ mit $y = 0$. Am Anfang ist $w = 0$, $b = 0$, die Lernrate ist $lr = 1$. Der Fehler ist der Mittelwert der Fehler beider Beispiele.
   (a) Berechne für jedes Beispiel $\hat{y}$, δ, $\frac{\partial L}{\partial w}$ und $\frac{\partial L}{\partial b}$.
   (b) Berechne die gemittelten Ableitungen und die neuen Parameter.
   (c) Berechne den mittleren Fehler vorher und nachher.

8. (4 Punkte) Ein verstecktes Neuron h mit ReLU und ein Ausgabeneuron mit Sigmoid sind hintereinander geschaltet: $z_1 = w_1 x + b_1$, $h = ReLU(z_1)$, $z_o = w_2 h + b_2$, $\hat{y} = \sigma(z_o)$.



Die Parameter sind $w_1 = 0.5$, $b_1 = 0.5$, $w_2 = 2$, $b_2 = -2$. Das Trainingsbeispiel ist $x = 2$, $y = 0$, die Lernrate $lr = 0.1$.
   (a) Rechne vorwärts: z_1, h, z_o, $\hat{y}$ und L.
   (b) Rechne rückwärts: δ_o, δ_h und die Ableitungen nach allen vier Parametern.
   (c) Berechne die neuen Parameter und damit h, $\hat{y}$ und L.

9. (6 Punkte) Das Netz hat zwei versteckte Neuronen mit ReLU und ein Ausgabeneuron mit Sigmoid.



Die Parameter sind $w_{11} = 1$, $w_{12} = -1$, $b_1 = 0.5$, $w_{21} = -1$, $w_{22} = 0.5$, $b_2 = 0.5$, $v_1 = 1$, $v_2 = 2$, $c = -1$. Dabei ist w_{ji} das Gewicht von x_i zu h_j. Das Trainingsbeispiel ist $x_1 = 2$, $x_2 = 1$, $y = 0$, die Lernrate $lr = 0.1$.

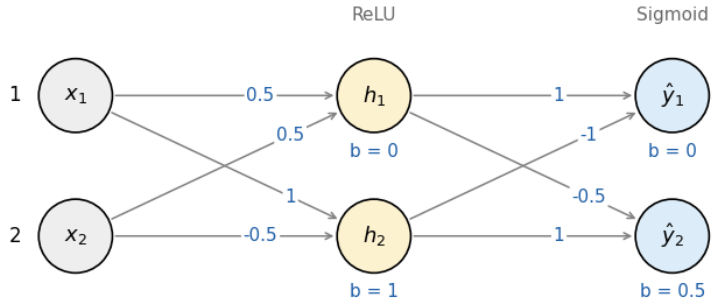
(a) Rechne vorwärts: z_1, h_1, z_2, h_2, z_o, $\hat{y}$ und L.

(b) Rechne rückwärts: δ_o, δ_1, δ_2 und die Ableitungen nach allen neun Parametern.

(c) Berechne die neuen Parameter und damit $\hat{y}$ und L.

(d) Drei Parameter ändern sich nicht. Welche und warum?

10. (5 Punkte) Das Netz hat zwei Ausgaben, der Fehler ist die Summe der Fehler beider Ausgaben.



Die Parameter sind $w_{11} = 0.5$, $w_{12} = 0.5$, $b_1 = 0$, $w_{21} = 1$, $w_{22} = -0.5$, $b_2 = 1$, $v_{11} = 1$, $v_{12} = -1$, $c_1 = 0$, $v_{21} = -0.5$, $v_{22} = 1$, $c_2 = 0.5$. Dabei ist v_{kj} das Gewicht von h_j zu $\hat{y}_k$. Das Trainingsbeispiel ist $x_1 = 1$, $x_2 = 2$, $y_1 = 1$, $y_2 = 1$.

(a) Rechne vorwärts: h_1, h_2, $\hat{y}_1$ und $\hat{y}_2$.

(b) Berechne δ_{o1} und δ_{o2} und die Fehlersignale δ_1 und δ_2 der versteckten Neuronen.

(c) Berechne $\frac{\partial L}{\partial w_{11}}$, $\frac{\partial L}{\partial w_{22}}$ und $\frac{\partial L}{\partial v_{21}}$.

11. (4 Punkte) Ein verstecktes Neuron h mit ReLU und ein Ausgabeneuron mit Sigmoid sind hintereinander geschaltet wie in der Aufgabe mit der Kette: $z_1 = w_1 x + b_1$, $h = ReLU(z_1)$, $z_o = w_2 h + b_2$, $\hat{y} = \sigma(z_o)$. Die Parameter sind $w_1 = 1$, $b_1 = 0.5$, $w_2 = 2$, $b_2 = -1$. Das Trainingsbeispiel ist $x = 1$, $y = 0$, die Lernrate $lr = 1$.
    (a) Rechne einen Schritt vorwärts und rückwärts und berechne die neuen Parameter.
    (b) Berechne mit den neuen Parametern z_1, h und $\hat{y}$.
    (c) Warum ändern sich w_1 und b_1 bei diesem Trainingsbeispiel nie mehr?
    (d) Für welche Inputs $x \geq 0$ ist das Neuron jetzt tot? Was bedeutet das für die Ausgabe des Netzes?

12. (3 Punkte) Für die Kette aus verstecktem Neuron (ReLU) und Ausgabeneuron (Sigmoid) mit $w_1 = -1$, $b_1 = 0.5$, $w_2 = 1$, $b_2 = 0.5$. Das Trainingsbeispiel ist $x = 1$, $y = 1$, die Lernrate $lr = 0.5$. hat jemand so gerechnet:

$z_1 = -1 \cdot 1 + 0.5 = -0.5$, $h = 0$	$z_o = 1 \cdot 0 + 0.5 = 0.5$, $\hat{y} = 0.6225$
$\delta_o = 0.6225 - 1 = -0.3775$	$\delta_h = \delta_o \cdot w_2 = -0.3775$
$\frac{\partial L}{\partial b_2} = \delta_o = -0.3775$	$b_2^{neu} = 0.5 + 0.5 \cdot (-0.3775) = 0.3112$

Finde die beiden Fehler und gib die richtigen Werte für δ_h und b_2^{neu} an.

13. (3 Punkte) Wir nehmen das Neuron aus der Aufgabe mit einem Input: $x = -1$, $w = 1$, $b = 0.5$, $y = 0$. Backpropagation liefert dort $\frac{\partial L}{\partial w} = -0.3775$.

    (a) Berechne den Fehler L für $w = 1.01$ und für $w = 0.99$ (die anderen Werte bleiben gleich) und daraus

    $$\frac{L(w + 0.01) - L(w - 0.01)}{0.02}$$

    Vergleiche mit dem Wert aus Backpropagation.

    (b) Ein Netz hat eine Million Parameter. Wie viele Durchgänge vorwärts braucht man, um alle Ableitungen auf diese Weise zu berechnen? Wie viele Durchgänge braucht Backpropagation?

14. (4 Punkte) (a) Was gibt das Fehlersignal δ eines Neurons an?

    (b) Warum berechnet man die Ableitungen rückwärts, von der Ausgabe zum Input?

    (c) Was ist eine Epoche? Wie geht man vor, wenn es mehrere Trainingsbeispiele gibt?

    (d) Warum benutzt man in den versteckten Schichten tiefer Netze ReLU statt Sigmoid?

15. (4 Punkte) Erweitere die Klasse `Wert` aus dem Notebook um eine Methode `tanh`. Die Funktion $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ liefert Werte zwischen -1 und 1, ihre Ableitung ist $1 - \tanh(z)^2$. In Python berechnet `math.tanh` den Funktionswert.

    Teste die Methode mit $f = \tanh(w \cdot x + b)$ für $w = 0.5$, $x = 2$, $b = -0.5$: Berechne `w.grad` und vergleiche mit der numerischen Ableitung.