Übungsblatt 3: Neural Netzwerke und Backpropagation
BA-INF 153: Einführung in Deep Learning für Visual Computing

Deadline:	15.05.2024 - 14:00 via eCampus	
Tutoren:	Alina Pollehn	s6aapoll@uni-bonn.de
	Johannes van de Locht	s6jovand@uni-bonn.de
Übungsgruppenleitung:	Jan Müller	muellerj@cs.uni-bonn.de

Theoretische Aufgaben (*15 Punkte + 4 Bonuspunkte*)

a) Auswirkungen der Aktivierungsfunktion auf den Gradienten *(4 Punkte)* Die Wahl der Aktivierungsfunktion kann einen großen Einfluss auf den Gradienten eines künstlichen Neuronalen Netzwerks haben. Eine ungünstige getroffene Auswahl kann zu einem verschwindenden Gradienten führen. Das bedeutet, dass die Gradientenwerte gefährlich nahe an die Maschinenpräzision herankommen und daher Gewichtsänderungen zur. Hier untersuchen wir dieses Problem anhand eines vereinfachten Beispiels.

Aufgabenstellung: Betrachten Sie ein vereinfachtes Netzwerk, das aus einer Eingabe $x \in \mathbb{R}$, n versteckten Schichten $h_i = \sigma(w_i \cdot h_{i-1})$ besteht, wobei w_i das Gewicht der Schicht ist, $\sigma : \mathbb{R} \to \mathbb{R}$ eine beliebige elementweise Aktivierungsfunktion ist, $h_0 = x$ und h_n die Ausgabe des Netzes ist.

1. (*2 Punkte*) Berechnen Sie die Ableitung der Ausgabe nach den Gewichten der versteckten Schichten, d.h. berechnen Sie $\frac{\partial h_n}{\partial w_i}$ für $i \in \{1, ..., n\}$.
2. (*2 Punkte*) Betrachten Sie σ als die Sigmoid-Funktion. Wie groß ist der maximale Wert der Steigung des Gewichts w_i in Bezug auf i, wenn $|w_i| < 1$ für alle $i \in \{1, ..., n\}$? Wie ändert sich Ihr Ergebnis, wenn Sie stattdessen die ReLU-Aktivierung $\sigma(x) = \max(0, x)$ betrachten?

b) Graphen und Kettenregeln für Backpropagation *(5 Punkte)* Auf dem vorherigen Übungsblatt haben wir uns auf die automatischen Differenzierung von Funktion in PyTorch zur Berechnung unserer Ableitungen verlassen. Wenn wir eine Funktion ausführen, erstellt PyTorch einen gerichteten, azyklischen Graph, der die Funktion beschreibt. Wenn wir die Funktion `backward` aufrufen, wird der Gradient unserer Funktion mit Hilfe des Graphen und der Kettenregel evaluiert. In dieser Übungsaufgabe wollen wir die Funktionsweise der automatischen Differenzierung anhand eines einfachen Beispiels veranschaulichen.

Betrachten wir die Rosenbrock-Funktion - die wir schon vom letzten Übungsblatt kennen - um ihre Ableitung mit Hilfe eines DAG und der Kettenregel zu bestimmen.

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2. \tag{1}$$

Aufgabenstellung: Zeichnen Sie einen gerichteten azyklischen Graphen der die Rosenbrook-Funktion in Gleichung 1 als Aneinanderreihung von Operationen $+, -, *$ und die Potenz $x\hat{}e$ mit natürlichem Exponenten e darstellt. Nehmen Sie an, dass $x_1, x_2 \in \mathbb{R}$ reelle Werte sind und verwenden Sie die Kettenregel und Ihren DAG um den Gradienten ∇f der Rosenbrock-Funktion zu berechnen. Geben Sie auch hier alle Ableitung an den Kanten des DAG explizit an.

Hinweis: Da das Erstellen von Graphen in LaTex etwas anstrengend ist, können Sie gerne ein Illustrationstool Ihrer Wahl verwenden und das Ergebnis in Ihr PDF einbinden.

c) Backpropagation für Multivariate Funktionen *(6 Punkte + 4 Bonuspunkte)*
Betrachten Sie ein MLP, das einen Eingabevektor mit 784 Dimensionen auf einen Vektor mit 10 Dimensionen abbildet. Das MLP hat zwei verborgenen Schichten, die jeweils 256 Neuronen haben und die Tanh-Aktivierungsfunktion verwenden. Die Ausgabeschicht des MLPs verwendet die Softmax-Aktivierungsfunktion. Das MLP kann rekursive definiert werden als

$$z^{(l)} := \phi^{(l)}(\underbrace{W^{(l)} \cdot z^{(l-1)} + b^{(l)}}_{o^{(l)}(z^{(l-1)})}) \text{ mit } z^{(0)} := x$$

wobei $W^{(l)}$ die Gewichtsmatrix, $b^{(l)}$ der Biaswert und $\phi^{(l)}$ die Aktivierungsfunktion der Schicht mit Index l ist und x die Eingabe in das MLP ist. Das MLP wird trainiert die Kreuzentropie

$$L_{\text{ce}}(z^{(3)}, y) = -\log z_y^{(3)}$$

zu minimieren, wobei y ein ganzzahliges Label $y \in [1:10]$ ist.

Repräsentieren sie die Berechnung der Zielfunktion als DAG, in dem Sie die Symbole x, $z^{(1)}$, $z^{(2)}$, $z^{(3)}$, y und L verwenden. Verwenden Sie dann die Kettenregel in mehreren Dimensionen um die Gradienten $\nabla_x L$ und $\nabla_{b^{(1)}} L$ als Produkt aus Jacobi-Matrizen aufzustellen. Geben Sie anschließend explizit die Jacobi-Matrizen der einzelnen Schritte an und leiten Sie die Jacobi-Matrizen der gegebenen Aktivierungsfunktionen aus der Definition der Aktivierungsfunktionen ab. Achten Sie darauf, dass das Produkt der Jacobi-Matrizen tatsächlich berechnet werden kann. Um dies zu überprüfen, geben Sie die Größe aller Jacobi-Matrizen an, die zur Berechnung der Gradienten $\nabla_x L$ und $\nabla_{b^{(1)}} L$ benötigt werden.

Bonus: *(4 Punkte)* Verwenden Sie Tensorprodukte um den Gradienten $\nabla_W^{(1)} L$ bzgl. der Gewichtsmatrix $W^{(1)}$ ebenfalls als Produkt aus Vektoren, Matrizen und Tensoren anzugeben. Beschreiben Sie dabei insbesondere die Form und Einträge der Jacobi-Matrix $J_{o^{(1)}}(W^{(1)})$.