Backpropagation

Autor und Herausgeber

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland (ORCID 0000-0001-7538-0990)
Editors: Konstantina Olioumtsevits (ORCID 0000-0003-2697-8659), Ekkehard Schnoor (ORCID 0000-0001-8252-1499), Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

Zusammenfassung

Backpropagation ist ein Algorithmus, der den Gradient des Verlust in Bezug auf die Modellparameter eines künstliches neuronales Netz (KNN) berechnet. Der Algorithmus besteht aus zwei aufeinanderfolgenden Phasen, die als Vorwärts- und Rückwärtsdurchlauf bezeichnet werden. Der Vorwärtsdurchlauf bewertet für die aktuellen Modellparameter die Aktivierungen und die resultierenden Vorhersagesfehler auf einem gegebenen Datensatz. Der Rückwärtsdurchlauf wendet dann die Kettenregel an, um die partielle Ableitungen des durchschnittlichen Verlust in Bezug auf die Modellparameter zu berechnen. Diese partielle Ableitungen bilden einen Gradient, der in einem Gradientenschritt verwendet werden kann, um die Modellparameter zu aktualisieren.

Definition

Betrachten Sie ein künstliches neuronales Netz (KNN) mit einer Milliarde Modellparameter und dessen Vorhersage für einen neuen Datenpunkt. Ist diese Vorhersage falsch, können die Modellparameter durch einen Gradientenschritt aktualisiert werden. Diese Aktualisierung erfordert die partielle Ableitungen des Verlust auf dem neuen Datenpunkt in Bezug auf alle Modellparameter.

Backpropagation ist ein Algorithmus zur Berechnung dieser partielle Ableitungen mit so wenig doppelter Berechnung wie möglich (Goodfellow et al., 2016, Sect. 6.5.3; Rumelhart et al., 1986). Allgemeiner berechnet sie den Gradient $\nabla f(\mathbf{w})$ einer Zielfunktion $f(\mathbf{w})$ in Bezug auf die Modellparameter $\mathbf{w}$ des KNN. Die Modellparameter bestehen aus den Kantengewichte und den Offset-Termen an den Knoten des KNN. Die Zielfunktion ist typischerweise der durchschnittliche Verlust

$$f(\mathbf{w}) = \frac{1}{m} \sum_{r=1}^{m} L\left(\left(\mathbf{x}^{(r)}, y^{(r)}\right), h^{(\mathbf{w})}\right),$$

der auf einem Datensatz von m Datenpunkte $\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right)$ anfällt. Backpropagation berechnet die partielle Ableitungen von $f(\mathbf{w})$ Schicht für Schicht (Fig. 1). Ihre Ausführung besteht aus zwei aufeinanderfolgenden Phasen, dem Vorwärtsdurchlauf und dem Rückwärtsdurchlauf.

Im Vorwärtsdurchlauf wird das KNN auf jeden Datenpunkt des Datensatz angewendet: Die Neuronen jeder Schicht bewerten ihre Aktivierungen aus den

Aktivierungen der vorherigen Schicht. Die letzte Schicht erzeugt dann den Ausgabe, der die Vorhersage für das Label eines Datenpunkt mit Merkmale $\mathbf{x}$ ist. Die Qualität dieser Vorhersage wird durch eine Verlustfunktion $L\left((\mathbf{x}, y), h^{(\mathbf{w})}\right)$ gemessen, wobei $h^{(\mathbf{w})}$ die Hypothese-Abbildung ist, die durch das KNN mit Modellparameter $\mathbf{w}$ repräsentiert wird. Im Rückwärtsdurchlauf werden die partielle Ableitungen von $f(\mathbf{w})$ in Bezug auf die Kantengewichte jeder Schicht rekursiv berechnet, vom Ausgabe zurück zum Input. Diese partielle Ableitungen bilden den gesamten Gradient $\nabla f(\mathbf{w})$. Der Rückwärtsdurchlauf erfordert ungefähr so viel Berechnung wie der Vorwärtsdurchlauf: seine Kosten wachsen proportional zur Anzahl der Kanten des KNN, eine partielle Ableitung, eine Multiplikation und eine Addition pro Kante (Goodfellow et al., 2016, Sect. 6.5.3).

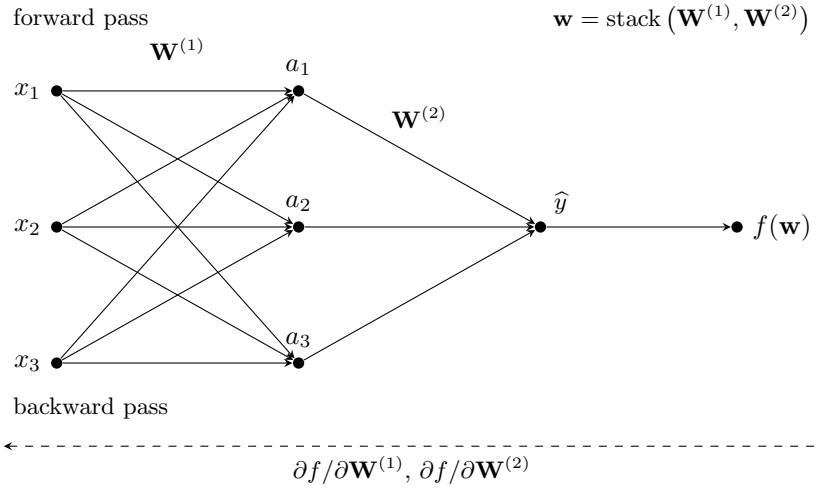

Abbildung 1: Backpropagation durch ein KNN mit einer versteckten Schicht. Die durchgezogenen Pfeile zeigen den Vorwärtsdurchlauf an: Die Merkmale x_1, x_2, x_3 erreichen die Neuronen a_1, a_2, a_3 über die Kantengewichte $\mathbf{W}^{(1)}$, dann den Ausgabe $\widehat{y} = h^{(\mathbf{w})}(\mathbf{x})$ über $\mathbf{W}^{(2)}$. Der Ausgabe ist eine Vorhersage für das Label eines Datenpunkt und verursacht den Verlust $f(\mathbf{w})$. Der gestrichelte Pfeil zeigt den Rückwärtsdurchlauf an, der in die andere Richtung verläuft und die partielle Ableitungen von $f(\mathbf{w})$ in Bezug auf die Kantengewichte beider Schichten liefert. Die Modellparameter $\mathbf{w}$ fassen die Kantengewichte beider Schichten zusammen; die Offset-Terme der Neuronen sind hier weggelassen.

Um die Details der Backpropagation zu veranschaulichen, betrachten Sie ein einfaches KNN: Es besteht aus drei Schichten mit jeweils einem Neuron. Seine Modellparameter sind die drei Kantengewichte $w^{(1)}, w^{(2)}, w^{(3)}$. Der Gradient eines Durchschnitts ist der Durchschnitt der Gradienten seiner Summanden, sodass ein Datensatz mit einem einzelnen Datenpunkt bereits die Rekursion zeigt. Die Zielfunktion $f(\mathbf{w})$ ist dann der Verlust dieses Datenpunkt. Schreiben

Sie $a^{(0)} = x$ für dessen Merkmal. Schicht $\ell = 1, 2, 3$ berechnet

$$z^{(\ell)} = w^{(\ell)} a^{(\ell-1)}, \qquad a^{(\ell)} = \sigma\big(z^{(\ell)}\big),$$

mit der Aktivierungsfunktion σ und ohne Offset-Terme. Der Ausgabe ist $h^{(\mathbf{w})}(x) = a^{(3)}$. Das Kantengewicht $w^{(3)}$ geht nur über $z^{(3)}$ in $f(\mathbf{w})$ ein, $w^{(2)}$ über $z^{(2)}$ und dann $z^{(3)}$, und $w^{(1)}$ über alle drei.

Die Kettenregel ergibt daher

$$\begin{aligned}
\frac{\partial f}{\partial w^{(3)}} &= \frac{\partial f}{\partial a^{(3)}}\, \sigma'\big(z^{(3)}\big)\, a^{(2)}, \\
\frac{\partial f}{\partial w^{(2)}} &= \frac{\partial f}{\partial a^{(3)}}\, \sigma'\big(z^{(3)}\big)\, w^{(3)}\, \sigma'\big(z^{(2)}\big)\, a^{(1)}, \\
\frac{\partial f}{\partial w^{(1)}} &= \frac{\partial f}{\partial a^{(3)}}\, \sigma'\big(z^{(3)}\big)\, w^{(3)}\, \sigma'\big(z^{(2)}\big)\, w^{(2)}\, \sigma'\big(z^{(1)}\big)\, x.
\end{aligned}$$

Jede Zeile wiederholt die darüber liegende und fügt zwei Faktoren hinzu. Würde man die drei Zeilen separat auswerten, würden die wiederholten Faktoren zweimal neu berechnet. Backpropagation berechnet sie einmal, vom Ausgabe rückwärts. Die wiederholten Faktoren bilden die partielle Ableitung von $f(\mathbf{w})$ in Bezug auf $z^{(\ell)}$. Diese partielle Ableitung folgt ihrer eigenen Rekursion,

$$\frac{\partial f}{\partial z^{(3)}} = \frac{\partial f}{\partial a^{(3)}} \sigma'\big(z^{(3)}\big), \qquad \frac{\partial f}{\partial z^{(\ell)}} = \frac{\partial f}{\partial z^{(\ell+1)}} w^{(\ell+1)} \sigma'\big(z^{(\ell)}\big) \quad \text{für } \ell = 2, 1.$$

Jedes Kantengewicht kostet dann eine Multiplikation,

$$\frac{\partial f}{\partial w^{(\ell)}} = \frac{\partial f}{\partial z^{(\ell)}} a^{(\ell-1)}.$$

Der Rückwärtsdurchlauf ist diese Rekursion, und die Aktivierungen $a^{(\ell)}$, die er benötigt, sind diejenigen, die der Vorwärtsdurchlauf bereits berechnet hat.

Die von der Backpropagation gelieferten partielle Ableitungen bilden den Gradient, der in einem Gradientenschritt verwendet wird:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla f(\mathbf{w}^{(t)}), \text{ für } t = 0, 1, \ldots,$$

mit der Schrittweite η. Jeder Gradientenschritt erfordert einen neuen Durchlauf der Backpropagation, da sich die Modellparameter nach jedem Schritt ändern. Diese partielle Ableitungen werden nicht nur für einen Gradientenschritt verwendet. Eine class activation map (CAM) erklärt eine Vorhersage eines Convolutional Neural Network (CNN), indem sie dessen Merkmalsabbildungen mit partielle Ableitungen des Ausgabe gewichtet (Selvaraju et al., 2017).

Siehe auch: KNN, tiefes Netz, Gradient, Gradientenabstieg, Gradientenschritt, Schicht, Kantengewicht, Modellparameter, Aktivierungsfunktion, Verlustfunktion, Neuron.

Querverweise

Literatur

1. Goodfellow et al. (2016). Deep Learning. MIT Press, Cambridge, MA, USA.

2. Selvaraju et al. (2017). Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In: 2017 IEEE Int. Conf. Comput. Vis., pp. 618–626.

3. Rumelhart et al. (1986). Learning Representations by Back-Propagating Errors. Nature, 323, pp. 533–536.