8. Übungsblatt

Aufgabe 31 Momentterm

Wir betrachten die Variante des Gradientenabstiegs, die einen Momentterm verwendet, um das Training zu beschleunigen, d.h., wir betrachten Gewichtsänderungen nach der Formel $\Delta w_t = -\frac{\eta}{2}\nabla_w e|_{w_t} + \alpha\ \Delta w_{t-1}$, $t = 0, 1, 2, 3, \ldots$. Nehmen Sie an, diese Trainingsregel werde für eine Fehlerfunktion angewandt, die eine unendlich ausgedehnte konstante Steigung aufweist. Wachsen in diesem Fall die Gewichtsänderungen über jede Grenze hinaus? Falls nicht: Was ist die Grenze für die Gewichtsänderungen?

Aufgabe 32 Deep Learning: Autokodierer

Ein Autokodierer ("autoencoder") ist, wie in der Vorlesung besprochen, ein dreischichtiges Perzeptron mit so vielen Ausgabe- wie Eingabeneuronen, das seine Eingaben auf (eine Rekonstruktion) diese(r) Eingaben abbilden soll. Wir nehmen an, daß sowohl in der versteckten Schicht als auch in der Ausgabeschicht die Aktivierungsfunktion ein gleichgerichtetes Maximum (*rectified maximum*) bzw. eine Rampenfunktion ist und daß weder Dropout-Training noch eine Begrenzung der Zahl der aktiven Neuronen verwendet wird. Warum ist es in diesem Fall unangebracht, in der versteckten Schicht genausoviele Neuronen zu verwenden wie in der Eingabe- bzw. Ausgabeschicht?
(Hinweis: Überlegen Sie, mit welchen (sehr einfachen) Parametern so ein Netz die Eingaben unverändert auf die Ausgaben abbilden kann.)

Aufgabe 33 Deep Learning: Autoenkodierer

Aufgabe 32 zeigte ein Problem auf, das man beim Training eines Autokodierers ("autoencoder") antrifft. In der Vorlesung wurden (bis zu) vier Ansätze angesprochen, mit denen man dieses Problem angehen und vermeiden kann. Was sind diese Ansätze und warum helfen sie, dieses Problem zu lösen?

Aufgabe 34 Lernen von Spielen

In der Vorlesung wurde kurz betrachtet, wie Deep-Learning-Neuronale-Netze zu einem Programm geführt haben, das in der Lage ist, das asiatische Brettspiel Go zu spielen und einen in der Weltrangliste führenden menschlichen Spieler überzeugend zu besiegen. Als grobe Analogie betrachten wir in dieser Aufgabe das einfache Brettspiel Tic-Tac-Toe. Wie kann man ein künstliches neuronales Netz trainieren, so daß es in die Lage versetzt wird, dieses Spiel zu spielen? Wie kann man das Brett als Eingabe und den auszuführenden Zug als Ausgabe kodieren? Ist es sinnvoll, ein Faltungsnetz ("convolutional neural network") zu verwenden?