Praktikumsbericht

DLO Deep Learning und Objekterkennung

Modul von Prof. Dr. Jan Salmen
Tutor: Luca Uckermann, Matthias Bullert, Jinxin Eisenhut

Technology
Arts Sciences
TH Köln

Arthur Kehrwald
Matrikelnummer: 11135125
Datum: 25. April 2025

Aufgabenstellung

Thema ist die Klassifzierung von handschriftlichen Ziffern auf Rasterbildern durch ein neuronales Netz anhand des MNIST Datensatzes. Aufgabe ist die Beschreibung des Trainingsprozesses mittels Gradientenabstieg anhand der genauen Struktur des Inputs, der Zwischenergebnisse und des Outputs. Außerdem soll mit verschiedenen Schichtstrukturen, Fehlerfunktionen, Startgewichten, Epochenanzahlen und Batchgrößen experimentiert werden.

Eingaben, Zwischenergebnisse und Ausgaben

Der Trainingsdatensatz besteht aus Graustufenbildern jeweils einer handschriftlichen Ziffer mit einer Auflösung von 28×28. In der gegebenen Implementierung wird der Datensatz mit der PyTorch utility `DataLoader` geladen. Das Laden in der Funktion `get_data` wird durch die Batchgröße parameterisiert. Daraus ergibt sich bei einer Batchgröße von 32 für jede Trainingsepoche ein zweidimensionaler Input-Tensor mit 32×784 Elementen. Es ist naheliegend, dass die 28×28 Grauwerte jedes der 32 Bilder in einer einzigen Dimension mit 784 Elementen dargestellt wird.

Dementsprechend muss die erste Schicht 784 Neuronen haben. In der Beispielimplementierung folgt darauf ein verborgene Schicht mit 50 Neuronen und drei weitere mit jeweils 10 Neuronen weniger, sodass die letze Schicht 10 Neuronen hat. Das Netz gibt also für jedes Trainingsbeispiel zehn Werte aus. Jeder Wert bezieht sich auf eine der zehn Ziffern und beschreibt, wie sicher sich das Netz ist, dass auf dem Bild die entsprechende Ziffer dargestellt ist.

Experimente

Batchgröße

Eine Erhöhung der Batchgröße beschleunigt auf meinem System mit CUDA das Training, vermutlich weil die Trainingsdaten weniger oft und in größeren Mengen auf die GPU übertragen werden. Gleichzeitig verlangsamt sich aber auch der Fortschritt pro Epoche.

Aktivierungsfunktion

Ich habe zusätzlich zur voreingestellten Funktion `Tanh` die Aktivierungsfunktion `PReLu` und `Softmax` verwendet. `PReLu` führt zu keiner deutlichen Veränderung. `Softmax` verschlechtert das Trainingsergebnis leicht.

Fehlerfunktion

Im Vergleich zur voreingestellten `CrossEntropyLoss` Funktion führen weder `MultiLabelMarginLoss` noch `BCELoss` zu einer deutlichen Veränderung.

Epochen

Mehr Epochen führt wie erwartet grundsätzlich zu besseren Trainingsergebnissen. Bei den meisten Testläufen erreichen die Testergebnisse nach ca. 50 Epochen eine Genauigkeit von 97% und verbessern sich danach kaum noch.

Startgewichte

Die Initialisierung aller Gewichte mit dem Wert 1 macht das Training vollkommen ineffektiv. Eine Initialisierung mit gleichmäßiger anstelle der voreingestellten Normalverteilung führt zu schlechteren Trainingsergebnissen.