HOCHSCHULE COBURG

Fakultät Elektrotechnik und Informatik
Prof. Dr. Florian Mittag

Deep Learning

Wintersemester 2024/25

Übungsblatt 5

Besprechung am 12.11.2024

Aufgabe 1 – Bildklassifikation auf dem MNIST-Datensatz

Auf dem letzten Übungsblatt haben wir lediglich zwei Ziffern des MNIST-Datensatzes ausgewählt, nun wollen wir im Notebook `image_classification.ipynb` alle zehn Ziffern klassifizieren können. Der MNIST-Datensatz besteht aus 60.000 Bildern im Trainingsdatensatz und 10.000 Bilden im Testdatensatz. Die Bilder sind ursprünglich als Grauwerte in 28×28 Pixeln gespeichert, wurden aber für die Übung jeweils zu 768-dimensionalen Vektoren konvertiert.

a) Implementieren Sie ein Modell zur logistischen Regression, also ein neuronales Netz mit nur einer Schicht. Statt der Sigmoid-Funktion müssen Sie jetzt aber die Softmax-Funktion verwenden, da wir 10 Ausgabe-Werte benötigen. (Softmax ist bereits in `edf.py` implementiert.) Als Loss-Funktion benutzen Sie Cross Entropy Loss.
   Der Fehler auf dem Testdatensatz sollte nach Epoche 10 ca. 8% betragen.

b) Angenommen, Sie initialisieren die Gewichte des Regressionsmodells zufällig und führen kein Training durch. Welchen Testfehler erwarten Sie und wieso?

c) Implementieren Sie nun ein MLP mit einer Hidden Layer zur Klassifikation der Daten. Nutzen Sie die Sigmoid-Funktion als Aktivierungsfunktion der Hidden Layer.
   Der Fehler auf dem Testdatensatz sollte nach dem Training bei ca. 2% liegen.

d) In der Vorlesung wurden weitere Aktivierungsfunktionen vorgestellt. Implementieren Sie
   - tanh
   - ReLU
   - LeakyReLU mit Faktor 0.01 für den negativen Bereich

   indem Sie dafür die im Notebook vorbereiteten Methoden `forward` und `backward` implementieren. Diese sollen die Aktivierungsfunktion bzw. deren Ableitung berechnen. Es ist bereits Code vorgegeben, der Ihre Implementierung plottet, so dass Sie testen können, ob alles richtig ist.

e) Definieren Sie für jede der drei Funktionen einen Computation Graph, der diese als Aktivierungsfunktion im Hidden Layer verwendet, und trainieren Sie das resultierende MLP. Welche Testfehler sehen Sie?

f) Eine weitere Möglichkeit die Ableitung zu bestimmen ist die numerische Differentiation. Dazu verwendet man den Differenzenquotienten, um den Differentialquotienten anzunähern, also $\frac{\partial f(x)}{\partial x} \approx \frac{f(x+h)-f(x-h)}{2h}$, wobei h ein kleiner Wert sein muss, hier $h = 10^{-4}$. Nutzen Sie den bereitgestellten Code zum Plotten der Ergebnisse, um die Näherung mit dem echten Wert zu vergleichen.

Aufgabe 2 – Lernrate und Anzahl und Größe der Hidden Layers

In dieser Aufgabe sollen Sie analysieren, wie verschiedene Lernraten und verschiedene Anzahlen von Hidden Layers die Performanz des neuronalen Netzes beeinflussen.

a) Der vorgegebene Code geht in einer Schleife über verschiedene Werte für die Lernrate (hier `[0.1, 0.5, 1.0]`) und setzt `edf.learning_rate` auf diesen Wert. Definieren Sie in der Schleife den Computation Graph, trainieren Sie das Modell und speichern danach den Trainingsfehler und den Testfehler in den dafür vorgesehenen Listen `train_err_per_lr` und `test_err_per_lr`.
   Finden Sie hierdurch heraus, welche der Lernraten die beste ist für die Sigmoid- und für die ReLU-Funktion.

b) Führen Sie das gleiche Experiment wie in Aufgabe a) durch, aber für eine verschiedene Anzahl von Hidden Layers (hier `[2, 4, 6]`). Auch hier sollen Sie die Ergebnisse in den dafür vorgegebenen Listen speichern und danach plotten.

c) Jetzt sollen Sie ausprobieren, welchen Einfluss die Breite der Hidden Layers haben, Anzahl der Neurone in einer Hidden Layer. Testen Sie hierfür die Werte 32, 64 und 128. Für die Lernrate und die Anzahl der Hidden Layers verwenden Sie die in Aufgabe a) und b) ermittelten besten Werte.

Viel Erfolg!