Aufgabenblatt 6

Praktikum Computer Vision
WiSe 2023/24

Christian Wilms

06. Dezember 2023

Abgabe bis 13. Dezember 2023, 08:00

Hinweis: Sollte Tensorflow keinen Speicher auf der GPU mehr allokieren können (`ran out of memory`), startet zunächst den Kernel neu und lasst alle Zellen erneut ausführen. Besteht das Problem weiterhin, ist euer Netz/sind eure Hyperparameter für die aktuelle Speicherkonfiguration zu groß.

Aufgabe 1 — CIFAR-10 mit einem Convolutional Neural Network

1. Auf den RZ-Rechnern öffnet eine Konsole (Terminal) und wechselt in das Verzeichnis (Befehl `cd`), in dem normalerweise eure Notebooks liegen. Der Befehl `cd /infhome` wechselt etwa in euer Home-Verzeichnis (Wurzelverzeichnis). Aktiviert die virtuelle Pythonumgebung mit Tensorflow/Keras mit dem Befehl:

   ```
   export PATH=/opt/pyenv/versions/3.11.6/bin:\$PATH
   ```

   und startet Jupyter Notebook über den Befehl `jupyter notebook`.
2. Ladet den CIFAR-10 Datensatz wie in Aufgabe 3.1 der letzten Woche beschrieben über Keras herunter. Nutzt die dortige Aufteilung in Trainings- und Testdaten. Führt zudem eine Vorverarbeitung durch, indem ihr die Datentypen der Bilder auf `'float32'` anpasst und den Wertebereich der Bilder auf $0 \dots 1$ verändert.
   Tipp: Nutzt eine eigene Zelle für das Laden des Datensatzes, die ihr idealerweise nur einmal ausführt.
3. Erzeugt nun ein CNN als sequentielles Modell. Fügt dem Modell einen Conv-Layer mit 32 3×3 Filtern und ReLU-Aktivierung hinzu, einen Flatten-Layer, einen Dense-Layer mit 256 Neuronen und ReLU-Aktivierung sowie einen Dense-Layer als Output-Layer mit Softmax-Aktivierung hinzu. Welche Größe (Anzahl Neuronen) muss Letzterer haben?
4. Kompiliert das Modell unter Verwendung der aus der letzten Woche bekannten Parametern. Trainiert nun das Modell auf den Trainingsdaten mit einer Batch Size von 32 für 20 Epochen und benutzt 20% der Trainingsdaten zur Validierung. Wie verändern sich Training Loss und Validation Loss über die Zeit?

Testet abschließend euer Modell mit der `evaluate`-Methode auf den Testdaten.

5. Kürzt nun das Training ab, indem ihr den `EarlyStopping`-Callback nutzt und nach 3 aufeinander folgenden Epochen ohne verbesserten Validation Loss das Training beendet. Fügt zudem noch einen `ModelCheckpoint`-Callback hinzu und speichert das beste Modell ab. Ladet die Gewichte dieses Modells dann wieder vor der Evaluation. Wie verändern sich die Ergebnisse?
6. Fügt einen zweiten Conv-Layer in euer Modell direkt hinter den ersten ein. Der neue Conv-Layer soll wiederum über 32 3×3 Filter und ReLU-Aktivierung verfügen. Wie verändern sich die Ergebnisse?
7. Ergänzt in euer Modell einen Max-Pooling-Layer mit der Größe 2×2. Dieser soll direkt nach dem zweiten Conv-Layer positioniert sein. Wie verändern sich die Ergebnisse und wie verändert sich die Laufzeit pro Epoche? Müssen nun mehr oder weniger Gewichte trainiert werden?
8. Ihr habt nun einen Block bestehend aus zwei Conv- und einem Max-Pooling-Layer. Dupliziert diesen Block und setzt eine Kopie hinter den ersten Block. Dieser zweite Block soll jedoch je Conv-Layer 64 Filter haben. Welche Auswirkungen hat dies auf die Ergebnisse? Wie viele trainierbare Gewichte sind nun in eurem Modell vorhanden und wie sind sie auf die Layer verteilt?
9. **Zusatzaufgabe:** Welchen Einfluss hat die Batch Size auf das Training? Testet sinnvolle 2er-Potenzen. Was ist die größte mögliche Batch Size?

Aufgabe 2 — Themen für die Projektaufgabe

Überlegt euch in eurer 3er-Gruppe zwei Themen für die Bildklassifikation, die ihr im Rahmen des Projekts bearbeiten wollen würdet. Wir wählen dann in der nächsten Woche gemeinsam ein Thema aus. Achtet bei der Auswahl der Themen darauf, dass ihr euch lediglich mit der Bildklassifikation beschäftigen sollt und nicht mit einer Klassifizierung/Lokalisierung von mehreren Objekten in einem Bild. Zudem sollt ihr für die beiden Themen einen groben Plan entwickeln, welche Variante (Deep Learning oder klassischer Ansatz) ihr wie umsetzen wollt. **Hinweis:** Zu dieser Aufgabe ist keine Abgabe nötig, wir diskutieren eure Vorschläge nächste Woche in der Präsenzzeit.

Aufgabe 3 — Zusatzaufgabe: Klassifikation mit vortrainierten CNNs

Vergleicht in dieser Aufgabe den Nächster-Nachbar-Klassifikator mit dem Merkmal 1D-Farb-Historgamme (3 Farbhistorgamme je Bild als Merkmalsvektor) mit der CNN-Architektur aus Aufgabe 1 auf den drei Klassen Auto (Label 1), Hirsch (Label 4) und Schiff (Label 8) des CIFAR-10 Datensatzes. Nutzt jedoch im Unterschied zu den Aufgaben in Woche 3 diesmal alle Bild der entsprechenden Klassen im CIFAR-10 Datensatz.

1. Wie gut klassifiziert der Nächster-Nachbar-Klassifikator die Bilder von Autos, Hirschen und Schiffen mit den 1D-Farb-Historgammen? Welchen Einfluss haben die Anzahl der Behälter je Histogramm und die Anzahl der Nachbarn im Klassifikator?

2. Wie müsst ihr die Architektur und das Training des CNNs anpassen, um mit den drei Klassen zu trainieren? Kommt ihr auf eine Trefferquote von über 90%?