UE 4 – Aufgabenstellung

Mehrklassen-Klassifikation mit Neuronalen Netzen

Überblick

Das Ziel der Vorbereitungsaufgabe war das Kennenlernen der Modellentwicklung mit Keras Framework. Eine optionale praktische Aufgabe war die Implementierung eines Neuronalen-Netz-Modells mit den HOG-Features anhand des Beispiels (first end-to-end example). Habt ihr das Modell nachimplementieren können? Ihr könnt gerne eure Erfahrungen / Ergebnisse im Forum zur Diskussion von Übungsaufgaben teilen.

Wie ihr sicherlich erkannt habt, handelt es sich bei dem oben genannten Beispiel-Modell um ein mit Dense-Layers implementiertes *Fully Connected Network,* dessen Schichten aus eindimensionalen Vektoren bestehen.

Aufgabe

In dieser Aufgabe werdet ihr eine andere Netzarchitektur – *Convolutional Neural Networks* (ConvNets) – kennenlernen. Diese Netzarchitektur wurde speziell für die Verarbeitung von Bildern konzipiert. Zusätzlich zu den Dense-Layers, die für die Klassifikationsaufgabe zuständig sind, kommen zwei weitere ConvNets Layer-Typen, die die Merkmalsextraktionsaufgabe übernehmen. Das sind Convolutional Layers und Pooling Layers.

Eure Aufgabe besteht darin, ein simples ConvNet-Modell zur Klassifikation von Verkehrszeichen zu implementieren, zu trainieren und mit euren Beispielbildern zu testen. Diese Aufgabe könnt ihr angelehnt an die Schritte der Modellentwicklung mit Keras in folgende Teilaufgaben unterteilen:

1. Datenaufbereitung
2. Aufbau des Modells
3. Kompilieren des Modells
4. Training des Modells
5. Evaluation des trainierten Modells
6. Speichern des Modells
7. Verwendung des gespeicherten Modells zur Klassifikation eigener Beispielbilder

Das angelegte Jupyter-Notebook enthält Hinweise zu den einzelnen Schritten.

Ihr könnt gerne das Forum nutzen, um die einzelnen Schritte zu besprechen.

Vergleicht die Performance eures ConvNet-Modells mit der Performance des Neuronalen-Netz-Modells mit den HOG-Features. Welches Modell schneidet besser ab? Warum?