Grundlagen KI Projekt persönlicher Report

Marvin Stier

Technische Universität Clausthal, Clausthal-Zellerfeld 38678, Deutschland

1 Aufgaben

1.1 Projektstart

Zu Beginn des Projekts war ich gemeinsam mit Jannes Bikker dafür verantwortlich, erste Experimente mit dem verworfenen Datensatz durchzuführen, dabei haben wir in Pair Programming Sessions mit altem Bestandscode aus einer vorangegenen Vorlesung gearbeitet.

1.2 Recherche, Preprocessing und HOG-SVM Modell

Nach Verwurf des Datensatzes und eines neuen Problems (Lokalisation statt Klassifikation) fiel es in meine Verantwortung, mögliche Algorithmen, beziehungsweise Modelltypen zu recherchieren, die dieses Problem lösen können. Dabei stieß ich auf CNN-basierte Ansätze, sowie auch ein etwas "klassischererMMachine Learning Ansatz mit einer Histogram of Oriented Graphs und Support Vector Machine Pipeline. Für letzteren Ansatz (HOG+SVM) war ich dann schließlich auch für die Entwicklung und Training verantwortlich. Dabei wurde der Algorithmus nicht von Grund auf implementiert, sondern die Python-Bibliothek dlib [1] verwendet.

Entwickelter Code Der entwickelte Code besteht aus den folgenden Dateien, welche dazu entworfen sind, in der Reihe ausgeführt zu werden: download.py, prepare.py, train.py, validate.py aus dem localization/ Ordner. Dazu kommt hyperparams.py, welche nicht ausgeführt wird, sondern nur die Parametrisierung der anderen Skripte ermöglicht.

Dabei wurde sich gegen ein Jupyter Notebook entschieden, um das Bottleneck der Systemleistung nicht durch Notebook Overhead zu verschlimmern.

Preprocessing Den aufwändigsten Teil stellte die Vorverarbeitung der Daten dar, es mussten zum einen Bilder konvertiert werden, die Labels aus einer separaten Datei gelesen, nach Gültigkeit gefiltert (es existiert eine Mindestgröße für Bounding Boxes), gruppiert und in ein für dlib [1] verständliches XML Format gebracht werden. Dieser Schritt erforderte viel Experimentieren und ich habe dies mit nicht ins Repository commiteten temporären Jupyter Notebooks gelöst, der fertige Code landete dann in prepare.py.

Hinweise Für weniger spannende Tasks wie Code für das Herunterladen von Dateien wurde ChatGPT für einen initialen Ansatz verwendet und der Code dann an die indivuellen Anforderungen des Projekts angepasst.

In graphics.ipynb im docs/ Ordner befindet sich noch Code der während des Schreibens der Dokumentation entstanden ist [2].

2 Challenges

Codemanagement Die ersten Iterationen der Modelle und Code wurden per WhatsApp geteilt, was mit der Zeit zu aufwändig und unübersichtlich wurde, weshalb wir fortan mit einem GitHub Repository gearbeitet haben. Als positiven Nebeneffekt ist auch die Version Control, über die Iterationen des Codes verglichen werden konnten.

Ein bis zum Ende des Projekts nicht vollständig gelöstes Problem ist die Verteilung der fertig trainierten Modelle, teilweise sind sie ins Repository mithilfe von git-lfs (Large File System) hochgeladen, da sie teils bis zu 160MB groß wurden, teilweise wurden sie im WhatsApp-Gruppenchat verteilt. Eine mögliche Lösung wären hier vermutlich Releases, idealerweise mit einer Pipeline zum Training auf leistungsfähiger Hardware kombiniert.

Systemleistung war ein limitierender Faktor, da dlib [1] sehr RAM-intensiv implementiert ist. Dies wurde umgangen, indem eine Möglichkeit, die Menge genutzter Daten anzupassen (insbesondere mit der HOG+SVM Pipeline sind sowieso nicht viele Daten für ein befriedigendes Ergebnis im Vergleich zu anderen Methoden benötigt).

3 Learnings

Datensätze sollten vor Verwendung besser überprüft und erforscht werden, um unnötige Arbeit oder Trainingszeit zu vermeiden. Ganz unnötig war es jedoch auch nicht, weil man schließlich auch aus Misserfolgen lernt.

Auch wenn die verwendete KI-Methode (HOG+SVM) keine Idealen Ergebnisse liefert, beziehungsweise nicht mehr der neuste Stand der Technik ist, war es interessant, damit zu experimentieren. Zudem vermute ich, dass ein Histogram of Oriented Graphs ein interessanter Input für ein CNN (zum Beispiel für schnellere Lokalisation gegenüber dem Sliding Window Detector mittels SVM) sein könnte, mit dem das Training gegenüber dem rohen Bild beschleunigt werden könnte.

Literatur

1. dlib Python Dokumentation, https://dlib.net/python/index.html, letzter Zugriff 16.02.2025
2. Histogram of Oriented Gradients explained using OpenCV, https://learnopencv.com/histogram-of-oriented-gradients/, letzter Zugriff 16.02.2025