Programmdokumentation:

Objekterkennung

Deep Learning Traffic Sign

Zeitraum: WS 2021/2022

Abgabedatum 10.01.2022

Projektteam
Nico Isheim
Fabian Beckdorf
Jacob Prütz
Ali Reza Teimoury
Julian Müller
Michael Sievers

Inhaltsverzeichnis

1. Liste der Gruppenteilnehmer

Name	Matrikelnummer	Semester
Nico Isheim	690222	AI3
Fabian Beckdorf	690047	AI3
Jacob Prütz	690043	AI3
Ali Reza Teimoury	690065	AI3
Julian Müller	690018	AI3
Michael Sievers	690593	AI3

2. Verteilung der Aufgaben

Mitglied	Aufgabe
Nico Isheim	Step2TrainRegionDetection, Step2TestRegionDetection, Kommentare geschrieben, Optimierung der Parametrisierung
Fabian Beckdorf	Step1TrainRegionDetection, Step2TestRegionDetection Step2TrainRegionDetection, Step2TestSystem
Jacob Prütz	Programmdokumentation, Anwenderdokumentation, Step1TrainClassification,, Step1TestClassification
Ali Reza Teimoury	Step2TrainClassification
Julian Müller	Step2TrainClassification, Step2TestClassification, optimieren der Netze, Dokumentation
Michael Sievers	Step2TrainClassification, Augmentierung Classification, optimieren der Netze, Programmdokumentation

Dazu ist zu sagen, dass jedes Gruppenmitglied grundsätzlich in allen Bereichen mitgewirkt hat, da vieles in gemeinsamer Arbeit entstanden ist und sich gegenseitig ausgetauscht und geholfen wurde.

3. Programmstruktur

Wie vorgegeben, haben wir die Skripte in “Step1*” und “Step2*” unterteilt. In den Step1-Skripten wurden die einzelnen Elemente der grundlegende Funktionalität bereitgestellt. Dabei unterteilen wir jeweils nach Training und Testen der Region Detection, also das finden der Straßenschilder auf den Bildern und nach Training und Testen der Classification, also der Einteilung des gefundenen Straßenschildes in die einzelnen Klassen der Geschwindigkeitsbegrenzungsschilder. Für die Geschwindigkeitsbegrenzungsschilder gibt es die Klassen “30GBS”, “50GBS”, “60GBS” und “AndereGBS”. Als Unterstützung für diese Skripte haben wir noch die Funktionen “augmentData”, “preprocessData” und “LoadAndRandomizeData” erstellt: Mit dem Skript “augmentData” wird unsere Datenbasis erhöht, indem die originalBilder verändert (spiegeln, drehen, zoomen) und dann dem Datensatz angefügt werden (Augmentierung). Das Skript “preprocessData” formatiert die Bilder aus dem übergebenen DataStore in die übergebene Bildgröße. Das Skript “LoadAndRandomizeData” liest die Bilder in einen DataStore ein, mischt diese und teilt den DataStore zu festgelegten Teilen in Trainings-, Validierungs- und Testdaten auf.
in den Step2-Skripten wurden Anpassungen an den Parametern der Netze vorgenommen, die die Performance der Netze verbessern.
Um die trainierten Netzen auf die Bilder anzuwenden kann das Skript Step1TestSystem / Step2TestSystem ausgeführt werden. In diesen Skripten sind jeweils die Region Detection und die Classification des jeweiligen Steps hintereinander geschaltet. Das Trainieren der Netze wird dabei bewusst nicht ausgeführt.

Diese Aufteilung ist sinnvoll, da so auch Teile des Systems verwendet werden können. Zum Beispiel kann man das bereits trainierte Netz verwenden, ohne die Zeit und Rechenpower aufzuwenden, die nötig ist, um es nochmal neu zu trainieren.

4. Verwendete Algorithmen

Bevor die Bilddaten den Netzen zum Training übergeben werden, werden sie augmentiert. Das bedeutet, dass die Bilder um bestimmte Eigenschaften verändert werden, um die Trainingsmenge weiter zu variieren und somit aus einem kleineren Datensatz mehr Möglichkeiten zum Trainieren zu generieren.
Bei der Augmentierung der Daten für die Regionserkennung verändern wir die Bilder und deren bounding Boxes, indem wir sie zufällig drehen, spiegeln, in ihrer Größe verändern und verzerren. Die Bilder zur Klassifikation werden gedreht und verschoben, um nah genug an den Variationen zu bleiben, die zwischen verschiedenen Bildern aus der echten Welt existieren können.

Da komplett unvorgelernte Netze mit begrenzten Ressourcen nicht in einem verträglichen Zeitrahmen zu einer vergleichbaren Akkuratesse zu trainieren wären, werden vortrainierte Netze verwendet und nur die letzten Schichten neu trainiert (transfer learning).

Zur Regionserkennung, also um zu Erkennen, wo sich in einem gegebenen Foto überhaupt ein Straßenschild befindet, wird ein Resnet 50, welchem ein faster RCNN zugrunde liegt, verwendet. Dieses Resnet 50 ist, wie der Name bereits vermuten lässt, 50 Schichten tief. Dieses Netz ist bereits mit tausenden Bildern vortrainiert worden, um eine möglichst breite

Variation an Objekten erkennen zu können. Diesem Netz haben wir unsere Daten nun gegeben und es somit auf unseren speziellen Anwendungsfall trainiert.

Um die Schilder zu kategorisieren verwenden wir das Alexnet, welches 8 Schichten tief und ebenfalls bereits vortrainiert ist, um die Erkennungsraten zu verbessern. Es verfügt somit über sehr gute Fähigkeiten Objekte auf Bildern zu klassifizieren. Diesem convolutional neural Network übergeben wir nun die Ausschnitte, auf denen nur die Schilder zu erkennen sind und trainieren somit das Netz darauf, die korrekte Geschwindigkeitsbegrenzung den Schildern zuzuordnen.

5. Genauigkeit der Netze

5.1 Step 1

In unserem ersten Ansatz haben wir für die Netze folgende Genauigkeiten erreichen können

Netz	**Funktion**	**Genauigkeit**
Region detection net (netDetectorResNet50)	Erkennt Geschwindigkeitsbegrenzungsschilder in Bildern	73%
Classification net (netClassification)	Ordnet die erkannten Schilder der Kategorien (30, 50 60, AndereGBS)	91,51%

5.2 Step 2

In unserem zweiten Ansatz haben wir für die Netze folgende Genauigkeiten erreichen können.

Netz	**Funktion**	**Genauigkeit**
Region detection net (netDetectorResNet50)	Erkennt Geschwindigkeitsbegrenzungsschilder in Bildern	73%
Classification net (netClassification)	Ordnet die erkannten Schilder der Kategorien (30, 50 60, AndereGBS)	98,87%

6. Diskussion der Ergebnisse

Um unsere Accuracy noch weiter zu erhöhen, müssten wir unser Neuronales Netzwerk noch weiter optimieren. Dies könnte man mit noch mehr Trainingsdaten umsetzen, bzw. noch weiter augmentieren. Dadurch hätte das Netz mehr Datensätze zum lernen. Zudem könnte man versuchen noch weiter an den Trainingsparameter rum zu schrauben. Zum Beispiel die Epochen verändern oder die Lernrate. Des Weiteren könnten wir uns noch weiter mit den vortrainierten Netzen befassen, um vielleicht eins zu nehmen was besser auf unsere Schilderkennung anspricht. Zusätzlich wurden wir in unserem Training leider durch unsere Ressourcen begrenzt, da unsere Grafikkarten “nur” 8 GB dedizierten Speicher haben. Da das Trainieren des Region Detection Netzes sehr viel Speicher benötigt konnten wir nur eine input Size von 448*448 Pixeln nutzen. Mit einer größeren input Size wären die Bilder schärfer und das Netzwerk hätte eine bessere Basis um die Bilder und Schilder zu differenzieren.

7. Sourcen

Die verwendete Ressourcen wurden ausschließlich über die Vorlesungsinhalte, die bereitgestellten Inhalte und über die Matlab eigene Dokumentation auf https://de.mathworks.com/help/ bezogen.