Vergleich eines Deep-Learning basierten Ansatzes mit konventionellen Techniken der Merkmalsextraktion im Kontext der Klassifikation von Straßenschildern

Arnold Freitas, Stefan Kuhn, Qing Huang

***Zusammenfassung*—TO do**

***Index Terms*—Bildgestützte Automatisierung, Convolutional Neuronal Network, Data Augmentation, Merkmalsextraktion, Deep-Learning**

I. Einleitung

TO do

II. Stand der Technik

Durch den Vergleich der herkömmlichen Feature-Detection-Methoden HOG (vgl. Abbildung 1) und MLP-Klassifikator wird die Auswirkung auf die Genauigkeit durch Datenaugmentation in konvolutionellen neuronalen Netzen untersucht.

Das HOG wurde erstmals von Navneet Dalal und Bill Triggs in einem auf der CVPR 2005 vorgestellten Papier vorgeschlagen und bildet Merkmale durch Berechnung und Zählung des Histogramms von Gradientenorientierungen lokaler Regionen eines Bildes. Die Kernidee ist, dass das Aussehen und die Form eines lokalen Ziels in einem Bild durch den Gradienten oder die Dichteverteilung der Kantenorientierung gut beschrieben werden kann. Im Vergleich zu SIFT arbeitet diese Methode mit lokalen Zellen des Bildes, so dass sie sowohl gegenüber der Geometrie des Bildes als auch gegenüber optischen Verformungen invariant bleibt.

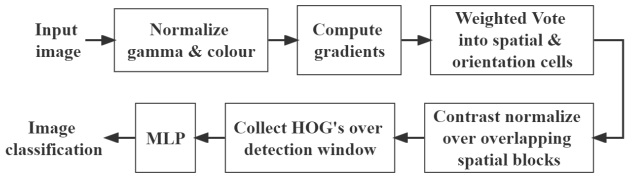

Abbildung 1. Ein Überblick über die Kette der Merkmalsextraktion (HOG).[1]

Nach der Merkmalsextraktion erfolgt die Bildklassifizierung mit einem MLP (vgl. Abbildung 2), einem vorwärtsstrukturierten künstlichen neuronalen Netz, das aus einer Eingabeschicht, einer verborgenen Schicht und einer Ausgabeschicht besteht. Im Gegensatz zu SVM, das nur eine optimale Hyperebene findet, werden bei MLP mehrere Hyperebenen durch eine Kostenfunktion gefunden. Die Eingabe

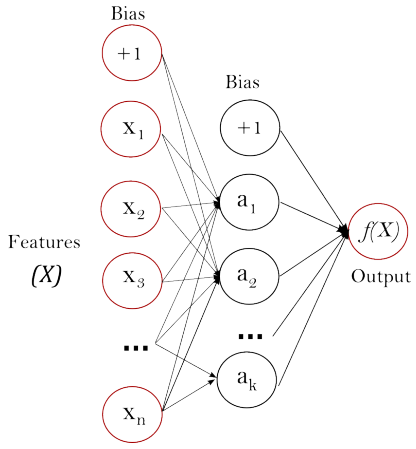

Abbildung 2. Multilayer perceptron network.[2]

ist der Merkmalsvektor der Instanz und die Ausgabe ist die Klasse der Instanz.

Darüber hinaus werden CNNs (vgl. Abbildung 3) eingesetzt, um Merkmale zu erkennen und zu klassifizieren, wobei auch Datenaugmentationen eingesetzt werden. Im Gegensatz zu den klassischen vollverknüpften Netzen kann ein CNN mehrere Convolutional-Layers, Pooling-Layers und Fully-connected-Layers. Der Convolutional-Layer verwendet mehrere lernfägige Filter zur Erkennung von Bildmerkmalen. Der Pooling-Layer komprimiert die Informationen der vorangehenden Ebene und wird für jede Feature Map einzeln durchgeführt. Der Fully-connected-Layer führt die Klassifizierung der Merkmale durch.

Aufgrund der Invarianz von CNNs in Bezug auf Verschiebung, Drehung, Größe und Beleuchtung kann die Datenaugmentation zum Einsatz kommen, um die Anzahl und Vielfalt der Trainingsmuster durch geometrische Transformationen wie Spiegelungen, Drehungen, Verschiebungen, Verzerrungen, Skalierungen und Farbverschiebungen usw. zu erhöhen und so die Robustheit des Modells zu verbessern.

Durch den Vergleich der Trainingsergebnisse der beiden Algorithmen wurde festgestellt, dass die Verwendung von Datenaugmentationen in faltigen neuronalen Netzen die Genauigkeit erheblich verbessern kann.

III. Konzept

To Do

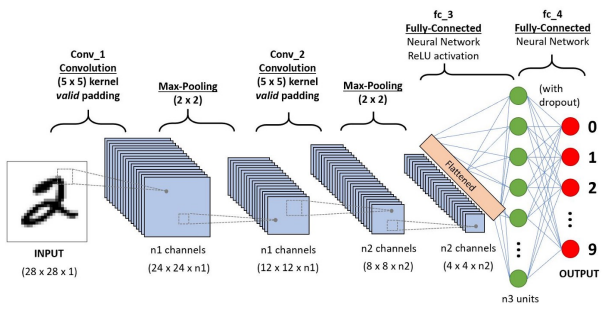

Abbildung 3. Ein CNN zur Klassifizierung handgeschriebener Ziffern.[5]

Tabelle I
PARAMETER FÜR DIE AFFINEN BILDTRANSFORMATIONEN

Parameter	Wert	Beschreibung
rotation_range	10	Gradzahlbereich für zufällige Drehungen
zoom_range	0.15	Anteiliger Bereich für zufälliges Skalieren
width_shift_range	0.1	Anteilige maximale horizontale Verschiebung gemessen an der Breite des Bildes
height_shift_range	0.1	Anteilige maximale vertikale Verschiebung gemessen an der Höhe des Bildes
shear_range	0.15	Scherwinkel in Grad

Die Angegebenen Parameter werden dem TensorFlow *ImageDataGenerator()* übergeben. Dieser generiert Batches mit der durch die Augmentation eingebrachten Datenerweiterung in Echtzeit und wird direkt in den *fit* befehl des TensorFlow Modells integriert.

A. *Datenvorverarbeitung*

Damit die Bilddateien sinnvoll für das Training eines CNN verwendet werden können sind einige Schritte der Datenvorverarbeitung notwendig.

Die Bilder des verwendeten Datensatzes verfügen nicht über einheitliche Dimensionen und werden auf eine Größe von 32 x 32 Pixeln durch bilineare Interpolation herunter skaliert (vgl. Abbildung 4). Diese Vereinheitlichung ist auf Grund der folgenden Aspekten von Vorteil: (1) Mini-Batch-Lernen mit Gradienten-basierten Verfahren erfordert die gleiche räumliche Auflösung für alle Bilder in einem Batch, (2) große Bildgrößen führen zu langsamerem Training. [6]

Um die Robustheit hinsichtlich einer Änderungen der Belichtungsverhältnisse zu erhöhen, wird der Farbraum der verwendeten Bilder Normalisiert. Das Ergebnis einer solchen Normalisierung ist in Abbildung 4 dargestellt. [3]

Abbildung 4. Visualisierung des Effektes der Skalierung auf 32 x 32 Pixel sowie der Normalisierung des RGB Farbraums

Die Bilder des GTSRB Datensatzes stammen aus realitätsnahen Verkehrssituationen, weshalb eine gleichbleibende Ausrichtung der Straßenschilder zur Kamera ist nicht gewährleistet ist. Um das zu entwickelnde Modells gegenüber Rotationen und weiteren Verzerrungen zu zusätzlich zu erhöhen werden auf Basis der vorhandenen Daten neue Bilder durch zufällige geometrische Augmentationen erzeugt (vgl. Abbildung 5). Dies führt gleichzeitig zu einer Vergrößerung des Datensatzes, der für das Training verfügbar ist. Somit zielt das Anwenden dieser Methodik auch auf eine Verbesserung der Vorhersage-Genauigkeit ab. Bei den verwendeten Augmentationen handelt es sich um die sogenannten affine Bildtransformationen. Als affine Transformationen sind definiert: Rotation, Spiegelung, Skalierung und Scherung, wobei auf horizontale und vertikale Spiegelung im Rahmen dieser Arbeit verzichtet wird.[4]

Ein weiterer Grundlegender Bestandteil der Datenvorverarbeitung ist das Aufteilen der Daten in 70% Trainings-, 15% Validierungs- und 15% Testdaten. Dabei entfallen Die Trainingsdaten werden dabei wie bereits durch den Namen impliziert für das Training der Modelle verwendet. Die Validierungsdaten werden im Rahmen des Trainings genutzt, um für jeden Trainingsschritt, einem sogenannten Epoch, den Validierungs-Loss zu ermitteln. Dieser Validierungs-Loss beschreibt den Wert eines gegebenen Fehlermaßes der bei der Prediktion der der Klassenzugehörigkeit der Validierungsdaten entsteht. Er wird genutzt um während des Trainings die Verbesserung eines Modells hinsichtlich der Fähigkeit auf ungesehene Daten zu messen. Im Anschluss an das Training der Modelle und der Auswahl bestimmter Modelle wird die Performance auf Basis der Testdaten analysiert. Die Aufsplittung des Datensatzes erfolgt durch die *train_test_split()* Funktion des *SciKit-Learn* Packages mit dem paramter $random\_state = 42$ durchgeführt.

Abbildung 5. Visualisierung des Effektes der affinen Transformationen, die im Rahmen der Daten-Augmentation auf die Bilder angewandt werden

Literatur

[1] Navneet Dalal und Bill Triggs. “Histograms of Oriented Gradients for Human Detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)* 2005 (2005).

[2] scikit-learn developers. “scikit-learn user guide (Release 0.18.2)”. In: (2017), S. 264.

[3] Hyun-Koo Kim, Ju H Park und Ho-Youl Jung. “An efficient color space for deep-learning based traffic light recognition”. In: *Journal of advanced transportation* 2018 (2018).

[4] Agnieszka Mikołajczyk und Michał Grochowski. “Data augmentation for improving deep learning in image classification problem”. In: *2018 international interdisciplinary PhD workshop (IIPhDW)*. IEEE. 2018, S. 117–122.

[5] Sumit Saha. “A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way”. In: *Towards Data Science* (2005).

[6] Hossein Talebi und Peyman Milanfar. “Learning to resize images for computer vision tasks”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, S. 497–506.