HAW HAMBURG

SEMINARARBEIT
Sebastian Berghoff

Erkennung und Klassifizierung von Verkehrsschildern unter schwierigen Verhältnissen mit Convolutional Neural Networks (CNN)

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

Betreuung durch: Prof. Stephan Pareigis
Eingereicht am: 12. Juli 2020

HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Abstrakt

Zur Klassifizierung von Verkehrsschildern mit neuronalen Netzen werden entsprechende Datensätze benötigt. Dabei ist die korrekte Klassifizierung auch unter schwierigen Verhältnissen wie Umweltstörungen durch Schnee und Regen oder teilweiser Verdeckung zu gewährleisten. Es werden Data Augmentation Methoden vorgeschlagen, um diese Einflüsse mit ImageMagick und Photoshop zu simulieren. Als Grundlage zur Simulation wird hierzu der GTSRB [1] Datensatz verwendet. Der erweiterte Datensatz wird als Trainings- und Validationsdatensatz für ein CNN verwendet. Die Experimente zeigen eine verbesserte Erkennung von Verkehrsschildern unter schwierigen Bedingungen.

Keywords: Verkehrsschilder Erkennung, CNN, Regen, Schnee

1 Einleitung

Zwei verschiedene Ansätze haben sich hierbei in der Vergangenheit als besonders vielversprechend herausgestellt. Zum einen ist dies die Sliding-Window Technik, zum anderen die Anwendung von CNN [4] [5]. Üblicherweise wird als Datensatz die "German Traffic Sign Recognition Benchmark" (GTSRB) verwendet. Dieser Datensatz beinhaltet allerdings kaum bis keine Bilder die unter schwierigen Verhältnissen, wie Regen oder Schnee gemacht wurden. Zu dem Prinzip des Sliding-Window wurden bereits Untersuchungen gemacht inwiefern solche Witterungen Einfluss auf die Klassifizierung haben und konnten dabei eine Genauigkeit von 98% erreichen [3]. Bei diesem Prinzip werden transformierte Teilausschnitte der zu untersuchenden Bilder auf Übereinstimmungen mit Mustern verglichen. Dieses Paper beschäftigt sich damit, welchen Einfluss schlechte Verhältnisse auf die Klassifizierung von CNNs hat. Insbesondere werden hierbei der Einfluss von Schnee, Regen, und teilweiser Abdeckung untersucht und miteinander verglichen.

Abbildung 1: Auszüge aus der German Traffic Sign Recognition Benchmark (GTSRB)

2 Die Datensätze

2.1 Der Verkehrsschilder Datensatz (GTSRB)

Der genutzte Datensatz, um die hier behandelten Verkehrsschilder Klassifikatoren zu trainieren, ist die "German Traffic Sign Recognition Benchmark" (GTSRB). Der Satz beinhaltet 51839 Bilder von Schildern aus 43 unterschiedliche Klassen. Die Bilder sind dabei bereits in geeignete Trainings- (39209) und Testdaten (12630) unterteilt.

In der realen Anwendung wie beim autonomen Fahren ist die Auswertung der gemachten Bilder in 2 Schritte unterteilt. Zuerst werden die Verkehrsschilder im Gesamtbild detektiert und lokalisiert. Danach können sie erkannt und klassifiziert werden. In der GTSRB sind die einzelnen Bilder bereits auf die Verkehrsschilder zugeschnitten, sodass keine Detektierung mehr stattfinden muss, sondern die Bilder direkt klassifiziert werden können.

Der Datensatz beherbergt allerdings auch einige realistische Probleme, die betrachtet werden müssen.
Zum einem ist die Qualität der Bilder teilweise mangelhaft. Sie haben eine geringe Auflösung, einen schlechten Kontrast, oder wurden unter einem extremen Winkel aufgenommen, sodass sie stark verzerrt sind. Dadurch sind einzelne Bilder sogar für das menschliche Auge schwer und teilweise nicht mehr zu Klassifizieren. Zum anderen sind die Daten nicht gleichmäßig über die Klassen verteilt. Die Verteilung für die GTSRB ist in Abb.2 dargestellt. Die am meisten vertretene Klasse ist "Geschwindigkeitsbegrenzung 50 km/h" mit 3000 Bildern. Die am wenigsten Vertretene ist "Geschwindigkeitsbegrenzung 20 km/h"

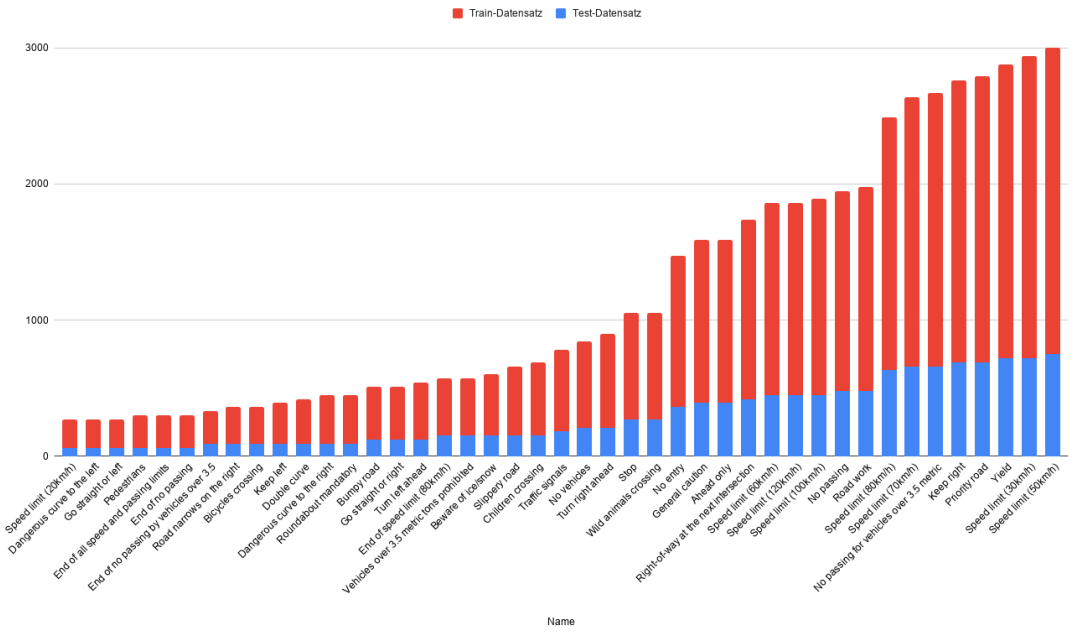

Abbildung 2: Verteilung der Trainings- und Testdaten über die Klassen der GTSRB

mit nur 270 vorkommen. Um ein Netzwerk erfolgreich trainieren zu können, müssen diese Probleme adressiert werden.

2.2 Die generierten Datensätze

In diesem Paper soll vor allem der Einfluss von schwierigen Verhältnissen auf die Klassifizierung von Verkehrsschildern untersucht werden. Dabei werden 3 Klassen gesondert untersucht. Dies sind der Einfluss von Schneefall, Starkregen und teilweiser Abdeckung von Verkehrsschildern.
Entsprechende Datensätze stehen nicht öffentlich in einer ausreichenden Vielfalt und Menge zur Verfügung, dass sie einen repräsentativen Vergleich zur GTSRB zulassen würden und eine Erstellung solcher Datensätze wäre unverhältnismäßig aufwändig. Daher wurde beschlossen diese Einflüsse stattdessen zu simulieren und die fehlenden Datensätze somit aus dem GTSRB zu generieren.

Abbildung 3: Durch Schnee teilweise bedecktes Verkehrsschild (Quelle:https://www.celebsnet.com/breaking/Snow-covered-traffic-signs-are-actually-still-valid-h2440.html)

2.3 Die Simulation von teilweise abdeckenden Einflüssen

In diesem Datensatz soll die regionale, teilweise Abdeckung von Verkehrsschildern simuliert werden. Im realen Einsatz können diese zum Beispiel durch an Schildern haftenden Schnee, nicht ausreichend gestutzte Äste mit Blättern oder Vandalismus, wie das Besprühen oder Bekleben von Verkehrsschildern, vorkommen.

Dieser Datensatz wurde mit Hilfe des Programms ImageMagick generiert. Dazu wurde je ein schwarzes Rechteck zufällig auf die einzelnen Bilder des GTSRB Datensatzes gezeichnet. Die Rechtecke sind so gewählt, dass sie die Form und Größe von Stickern repräsentieren, mit denen die Schilder beklebt wurden. Diese Rechtecke unterliegen den folgenden Bedingungen:

- Der äußere Rand von 3px kann nicht Teil des Rechteckes sein
- Die Breite darf maximal ein Viertel der verbleibenden Breite betragen
- Die Höhe darf maximal ein Achtel der verbleibenden Breite betragen

Die Resultate lassen sich der Abb.4 entnehmen. Es kann vorkommen, dass wichtige Informationen wie in Zeile 4 vollständig bedeckt sind, sodass eine Klassifizierung unmöglich wird. Es gibt aber auch Situationen, wo das eigentliche Verkehrsschild überhaupt gar

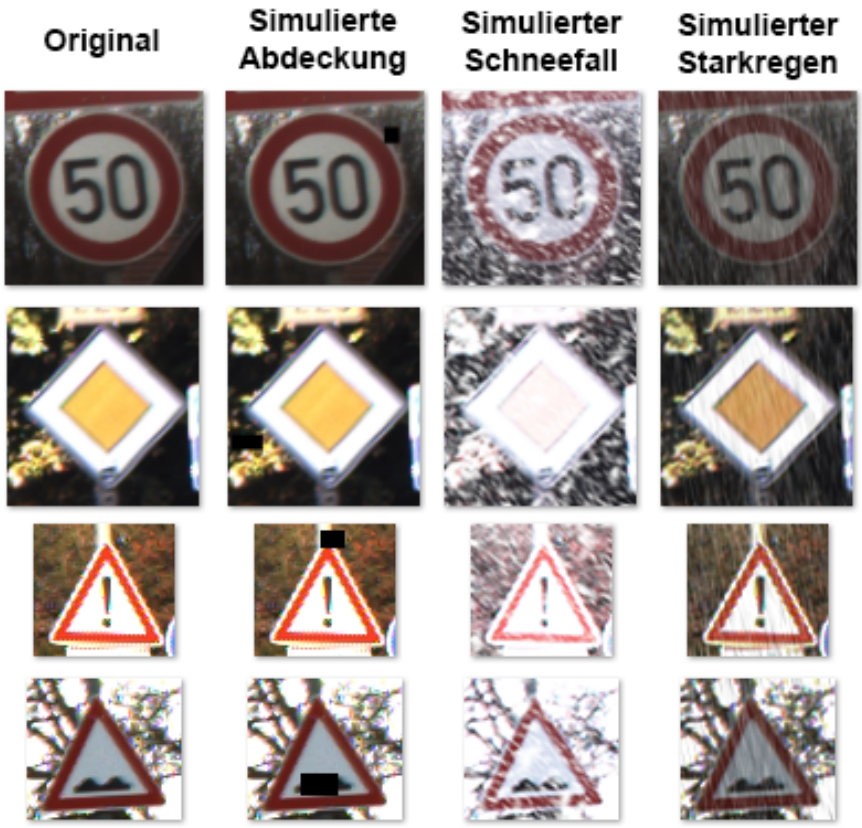

Abbildung 4: Originale und bearbeitete Datensätze im Vergleich

nicht bedeckt wird, wie in Zeile 2 zu sehen. Die Abdeckungen sind somit repräsentativ für reale Bedingungen.

2.4 Die Simulation von Schneefall

Schneefall beeinträchtigt die Sicht von Kameras, erhöht die Helligkeit durch die Reflexionen von Schnee und reduziert die Sättigung von Videoaufnahmen. Eine Simulation von Schnee (und Starkregen) wurde in anderen Arbeiten oft nur stark vereinfacht dargestellt. Dieses Paper legt einen großen Wert darauf, eine realistische Simulation von Schnee und Regen zu erstellen. Dazu wurden die Bilder in Photoshop CS6 mit einem Action Macro als Batch bearbeitet. Die Einzelschritte sehen dabei wie folgt aus:

1. Skalierung mit “Bicubic Smoother” auf 200%

2. Erstellung und Anwendung eines “Adjustment Layers” vom Typ “Channel Mixer” mit Änderung im
3. Output Channel “Blue” auf Green +80% und Blue +40%
4. Die Sättigung wird um 25% reduziert
5. Die Helligkeit wird um 60% erhöht
6. Ein neuer Layer im Modus “Screen” wird erstellt und mit der neutralen Farbe schwarz gefüllt
7. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 200%
8. Anwendung von “Gaussian Blur” mit dem Radius 2px
9. Anwendung eines “Adjustment Layers” des Typ “Levels” mit einer unteren Grenze von 118 und einer oberen von 138
10. Anwendung von “Motion Blur” mit einer Distanz von 10px unter einem Winkel von -25°
11. Ein neuer Layer im Modus “Screen” wird erstellt und mit der neutralen Farbe schwarz gefüllt
12. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 200%
13. Anwendung von “Gaussian Blur” mit dem Radius 2px
14. Anwendung eines “Adjustment Layers” des Typ “Levels” mit einer unteren Grenze von 118 und einer oberen von 138
15. Skalierung mit “Bicubic Smoother” auf 200%
16. Doppelte Anwendung von “Motion Blur” mit einer Distanz von 5px unter einem Winkel von +25°
17. Ein neuer Layer im Modus “Screen” wird erstellt und mit der neutralen Farbe schwarz gefüllt
18. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 200%
19. Anwendung von “Gaussian Blur” mit dem Radius 1px

20. Anwendung eines “Adjustment Layers” des Typ “Levels” mit einer unteren Grenze von 108 und einer oberen von 177
21. Doppelte Anwendung von “Motion Blur” mit einer Distanz von 5px unter einem Winkel von -60° Merge aller Layer
22. Skalierung mit “Bicubic Sharper” auf 50%

Die Aktionen 1, 21 und 22 lassen eine präzisere Bearbeitung des Bildes bei einer höheren Auflösung zu. Die Schritte 2 bis 4 verändert die Lichtverhältnisse in einer Form, wie sie unter Schneefall realistisch aussehen. Die Layer die in 5 bis 9, 10 bis 15 und 16 bis 20 erstellt werden generieren die Schneeflocken in 3 unterschiedlichen Größen und Einfallswinkeln. Größere und somit nähere Flocken, fallen dabei unter einem Winkel ein, der durch die Aerodynamik des Autos bedingt seitlicher ist. Durch die Verwendung des Noise werden stets unterschiedliche Flocken erzeugt und das neuronale Netz kann nicht die Position der Schneeflocken lernen. Das generierte Ergebnis ist qualitativ hochwertig und ist eine gute Simulation für realistischen Schneefall (Abb.4).

2.5 Die Simulation von Starkregen

Starkregen beeinträchtigt ebenfalls die Sicht von Kameras, verringert aber auch darüber hinaus die generelle Helligkeit und damit den maximalen Kontrast von Video-Aufnahmen. Auch hier wurden die Bilder in Photoshop CS6 mit einem Action Macro als Batch bearbeitet. Die Einzelschritte sehen dabei wie folgt aus:

1. Skalierung mit “Bicubic Smoother” auf 200%
2. Die Sättigung wird um 15% reduziert
3. Die Helligkeit wird um 50% reduziert
4. Der Kontrast wird um 30% reduziert
5. Ein neuer Layer im Modus “Hard Light” wird erstellt und mit der neutralen Farbe grau gefüllt
6. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 50%
7. Doppelte Anwendung von “Motion Blur” mit einer Distanz von 15px unter einem Winkel von -78°

8. Ein neuer Layer im Modus “Screen” wird mit 80% Opacity erstellt und mit der neutralen Farbe schwarz gefüllt
9. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 50%
10. Anwendung eines “Adjustment Layers” des Typ “Levels”mit einer unteren Grenze von 253 und einer oberen von 255
11. Doppelte Anwendung von “Motion Blur” mit einer Distanz von 10px unter einem Winkel von -82°
12. Ein neuer Layer im Modus “Screen”wird mit 80% Opacity erstellt und mit der neutralen Farbe schwarz gefüllt
13. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 50
14. Anwendung eines “Adjustment Layers” des Typ “Levels” mit einer unteren Grenze von 220 und einer oberen von 255
15. Dreifache Anwendung von “Motion Blur” mit einer Distanz von 10px unter einem Winkel von -90°
16. Ein neuer Layer im Modus “Hard Light” wird erstellt und mit der neutralen Farbe grau gefüllt
17. Anwendung eines “Monochromatic Gaussian” Noise der Stärke 50%
18. Skalierung mit “Bicubic Smoother” auf 400%
19. Doppelte Anwendung von “Motion Blur” mit einer Distanz von 15px unter einem Winkel von -71°
20. Anwendung von “Motion Blur” mit einer Distanz von 22px unter einem Winkel von -71°
21. Merge aller Layer
22. Skalierung mit “Bicubic Sharper” auf 50%

Die Aktionen 1, 21 und 22 lassen erneut eine präzisere Bearbeitung des Bildes bei einer höheren Auflösung zu. Die Schritte 2 bis 4 verändern die Lichtverhältnisse in einer Form, wie sie unter Starkregen realistisch aussehen. Die Layer die in 5 bis 7 und 16 bis 20 erzeugt werden generieren gräuliche Regentropfen in 2 unterschiedlichen Größen wie sie durch eine Reflexion der Umwelt entstehen. Weitere zwei Layer werden in 8 bis 15 erzeugt. Sie generieren hellere Tropfen die eine Reflexion von Scheinwerferlicht darstellen. Auch hier sind die Regentropfen durch das zufällige Rauschen bei jeder Generierung anders, sodass sich das neuronale Netz nicht auf die Positionen der Regentropfen gewöhnen kann. Das Ergebnis kann der Abb.4 entnommen werden.

3 Umgebung

Für das Projekt werden folgende Packages benutzt:

- OpenCV
- NumPy
- scikit-learn
- scikit-image
- imutils
- matplotlib
- TensorFlow 2.0

4 Das Verkehrsschild CNN

4.1 Der Aufbau

Als CNN wurde das Netzwerk in Abb.5 gewählt. Das CNN kann zur Erklärung in 6 Abschnitte unterteilt werden. Der erste Abschnitt nutzt einen 5x5 Kernel, um gröbere Strukturen zu lernen, die helfen Form und Farbe der Verkehrsschilder zu lernen. Der zweite und dritte Abschnitt vertiefen das Netzwerk. Abgeschlossen wird es mit zwei fully-connected Layern und dem Softmax-Classifier.

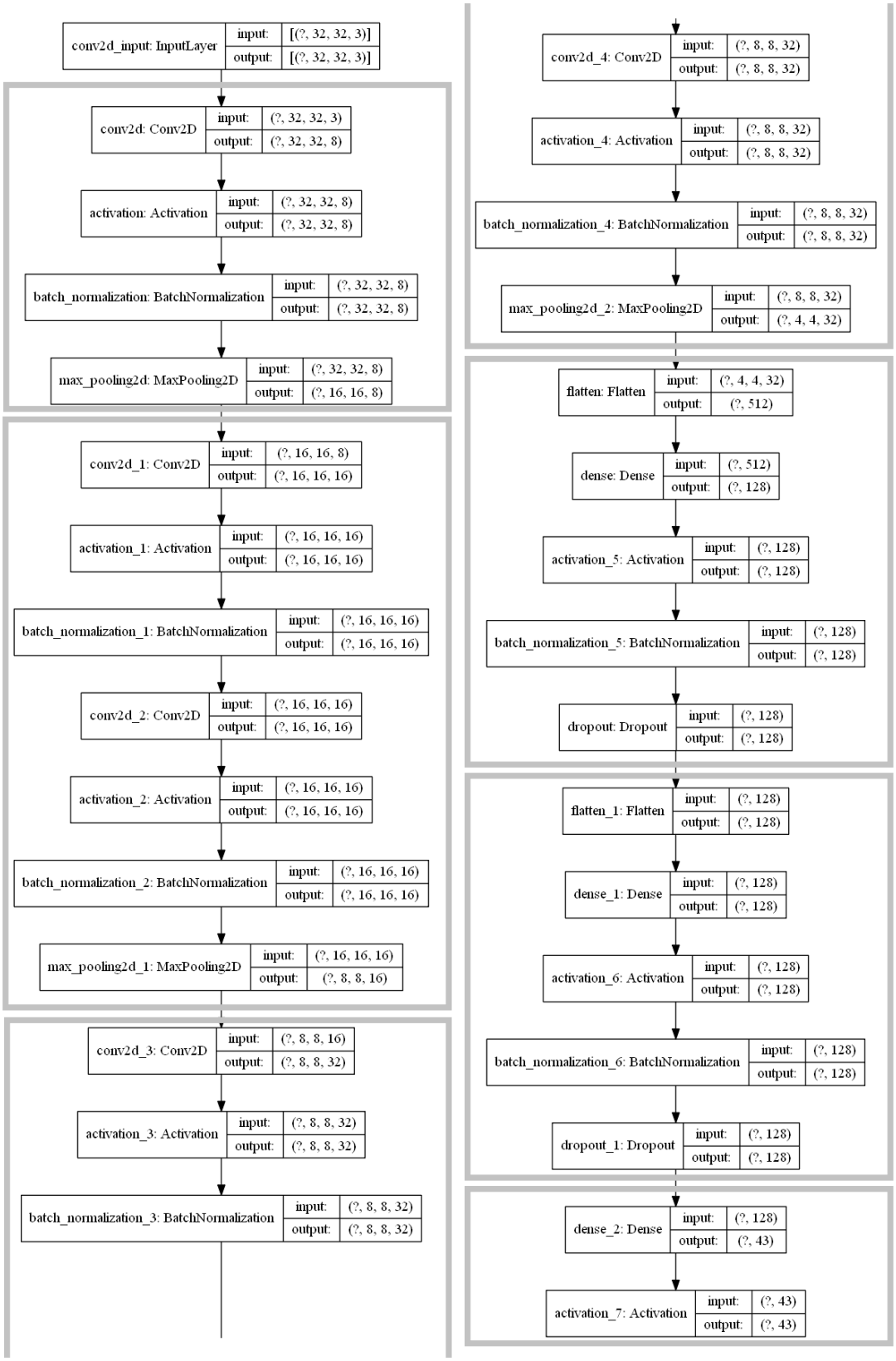

Abbildung 5: Der Aufbau des gewählten CNN

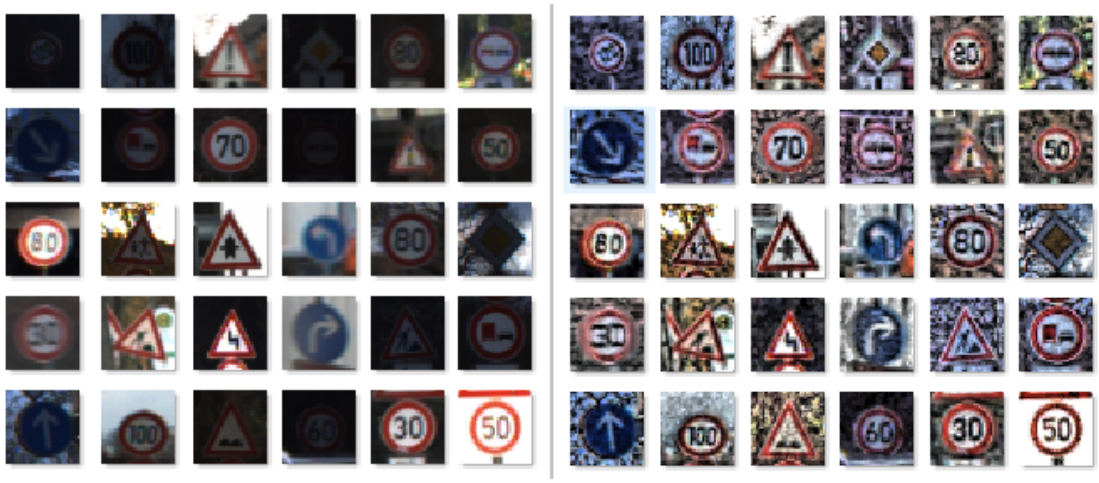

Abbildung 6: Originaldaten auf 32x32Pixel skaliert ohne und mit adaptivem Histogrammausgleich

Die Verwendung von Dropout hilft dabei Overfitting zu vermeiden [6]. Dieses Netzwerk kombiniert eine gute Klassifizierungsqualität mit verhältnismäßig schnellem Training, was für die spätere Vergleichsanalyse hilfreich ist. Auf eine komplexere Lösung wie sie zum Beispiel in [8] vorgestellt wurde, wurde zur Reduzierung des Aufwandes bewusst verzichtet. Sie bietet zwar eine bessere Genauigkeit, allerdings konzentriert sich dieses Paper auf den Vergleich von unterschiedlichen Datensätzen. Um Aussagen hierüber zu treffen, reicht das hier gewählte Netz aus.

4.2 Ablauf des Trainings-Script für das CNN

Der GTSRB Datensatz ist bereits in Trainings- und Testdaten unterteilt. Die Pfade zu den Bildern der einzelnen Datensätze sind je in einer .csv-Datei gespeichert, die zu Beginn extrahiert werden müssen.
Anschließend werden die einzelnen Datensätze geladen und die Bilder auf 32x32 Pixel skaliert, um sie für das CNN auf ein einheitliches Format zu bringen. Danach wird ein adaptiver Histogrammausgleich auf ihnen ausgeführt, um den zuvor genannten teilweise mangelhaften Kontrast zu verbessern [7]. Das Ergebnis kann in Abb.6 betrachtet werden. Das Ergebnis entspricht nicht der Realität, verbessert allerdings die Erkennbarkeit für das menschliche Auge und hilft dem Netzwerk bei der Klassifizierung. Die geladenen Datensätze werden in NumPy Arrays konvertiert und die Skalierung der Bilder auf der Raum [0..1] gebracht.

Nun werden die Trainings und Testlabel one-hot encodiert. Es wird also der Wert einer jeden einzelnen Binärziffer einer der 43 Klassen zugeordnet. Den einzelnen Klassen werden anschließend Gewichte zugeordnet, um einer ungleichmäßigen Klassenverteilung entgegenzuwirken. Diese werden während des Trainings verwendet.
Nun wird die Data-Augmentation initialisiert. Während des Trainings werden dadurch zufällige Transformationen auf die Bilder ausgeführt. Diese ist notwendig, um mehr Varianz in den Datensatz zu bringen und damit das Overfitting zu reduzieren. Es wurden hierfür die folgenden Parameter benutzt:

- rotation_range=10,
- zoom_range=0.15,
- width_shift_range=0.1,
- height_shift_range=0.1,
- shear_range=0.15,
- horizontal_flip=False,
- vertical_flip=False,
- fill_mode="nearest"

Hierbei ist besonders hervorzuheben, dass Straßenschilder nicht gespiegelt werden können. Das CNN wird anschließend mit dem Adam-Optimierer kompiliert und mit der Funktion “fit_generator()” trainiert. Als Batch-Size wurde dabei stets 64 gewählt. Die Anzahl der Epochen betrug jeweils ungefähr 10-30, unterscheidet sich aber für jedes Netzwerk, da sich der Zeitpunkt, zu dem Overfitting einsetzt, jeweils unterscheidet. Um dies zu vermeiden wird vorher ein Early-Dropout durchgeführt. Abschließend wird das Model auf der Festplatte gespeichert.

4.3 Ablauf der Embedded Prediction

Der Klassifikator wurde auf einem Raspberry Pi mit Pi-Cam getestet, um das System auf eine zukünftige Anwendung in einem Embedded System vorzubereiten, die Ergebnisse sind aber von der Hardware unabhängig. Zunächst wird das Modell geladen. Nun können mit der Taste T einzelne Frames der Kamera analysiert werden. Diese Frames

werden dabei genauso wie beim Training zunächst auf 32x32 Pixel skaliert und mit einem adaptiven Histogrammausgleich verbessert. Nun wird mit dem bearbeiteten Frame und dem geladenen Netzwerk die Prognose erstellt und in der Konsole ausgegeben. Die Performance auf dem Embedded System wurde hierbei nicht weiter beachtet, verbessert oder analysiert, da diese Arbeit einen anderen Schwerpunkt hat und sich eine verwandte Arbeit, die ebenfalls im Rahmen des Seminars entstanden ist, bereits damit beschäftigt. Zudem wurde bereits gezeigt, das auch große ConvNets auf FPGA basierten Embedded Systems in Echtzeit laufen können [2].

5 Methoden

Es wurden insgesamt 7 unterschiedliche Netze mit unterschiedlichen Datensätzen trainiert:

1. Unbearbeitete Trainings- und Testdaten
2. Unbearbeitete Trainingsdaten und Testdaten mit simulierter Abdeckung
3. Trainingsdaten und Testdaten mit simulierter Abdeckung
4. Unbearbeitete Trainingsdaten und Testdaten mit simuliertem Schneefall
5. Trainingsdaten und Testdaten mit simuliertem Schneefall
6. Unbearbeitete Trainingsdaten und Testdaten mit simuliertem Starkregen
7. Trainingsdaten und Testdaten mit simuliertem Starkregen

Es wurden jeweils bis zu maximal 30 Epochen trainiert, die in der Praxis jedoch nicht erreicht werden, da zuvor stets ein Early-Dropout veranlasst wird, sobald sich die Val-Accuracy zunehmend verschlechtert.

Danach werden die Netze mit verschiedenen Testsätzen evaluiert:

- Netzwerk 1 mit unbearbeiteten Testdaten
- Netzwerk 1 mit Testdaten mit simulierter Abdeckung
- Netzwerk 1 mit Testdaten vom simulierten Schneefall
- Netzwerk 1 mit Testdaten vom simulierten Starkregen

- Netzwerk 2 mit Testdaten mit simulierter Abdeckung
- Netzwerk 3 mit Testdaten mit simulierter Abdeckung
- Netzwerk 4 mit Testdaten vom simulierten Schneefall
- Netzwerk 5 mit Testdaten vom simulierten Schneefall
- Netzwerk 6 mit Testdaten vom simulierten Starkregen
- Netzwerk 7 mit Testdaten vom simulierten Starkregen

Folgende Hardware wurde für das Training benutzt:

- Intel i9 9900k
- GSkill Trident Z 16GB, 3600MHz, CL15
- Gigabyte Z390 Aorus Pro
- Radeon RX Vega 56 GPU

6 Ergebnisse

Zunächst wird der Trainingsverlauf des mit dem originalen Datensatz trainierten Netzwerk 1 betrachtet. In Abb.7 ist der Verlauf von train_loss, train_acc, val_loss und val_acc über die Epochen abgebildet. Die beste Genauigkeit wurde nach 26 Epochen erreicht, woraufhin ein Early-Dropout erfolgte, da danach Overfitting einzusetzen begann.

In Abb.8 ist die Genauigkeit des unter normalen Bedingungen trainierte Netzwerk 1 dargestellt. Den höchsten f1-score erreichen dabei die Klassen “No passing for vehicles over 3,5 metric tons”, “Vehicles over 3,5 metric tons prohibited” und “Turn left ahead” mit annähernd 100%. Am schlechtesten schneidet die Klasse “Double curve” ab. Sie erreicht nur einen f1-score von 75%. Die Genauigkeit über alle Klassen hinweg (weighted avg. f1-score) beträgt 96%.
Besonders hervorzuheben ist auch die Klasse “Dangerous curve to the right”, welche eine sehr hohe Disparität zwischen ihrem Recall von 100% und der Precision von nur 81% aufweist.

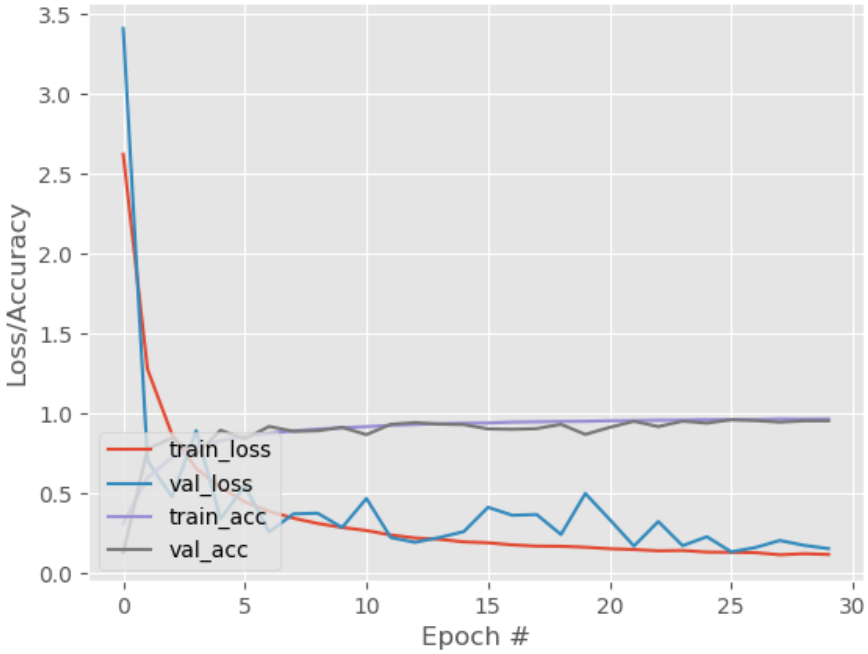

Abbildung 7: Trainingsverlauf von Network 1 über die einzelnen Epochen

Die Trainingsverläufe und Evaluierungen der anderen Netzwerke sind analog hierzu und werden im Folgenden nicht einzeln im Detail betrachtet. Allgemein sinkt die Anzahl der Trainings-Epochen, die bis zu einem Early-Dropout verstreichen mit der Stärke der hinzugefügten Bildstörungen. Die finalen Genauigkeiten der untersuchten Netzwerke sind in Abb. 9 aufgelistet. Bei der Erkennung von teilweise abgedeckten Verkehrsschildern schneidet das Netzwerk 2 am besten ab, welches mit dem originalen Datensatz trainiert und mit dem teilweise abgedeckten Datensatz getestet wurde. Bei Schnee und Regen verhält es sich anders. Hier schneiden die Netzwerke 5 und 7 am besten ab, welche mit den simulierten Daten trainiert und getestet wurden.

7 Diskussion

Während sich die hohe Genauigkeit der Klasse “No passing for vehicles over 3,5 metric tons” in Netzwerk 1 noch durch eine der größten Trainings-Mengen von 2010 Bildern erklären ließe, gelingt dies bei den Klassen “Vehicles over 3,5 metric tons prohibited” und “Turn left ahead” nicht mehr. Mit einer Menge von je nur 420 Bildern gehören sie zu den am wenigsten vertretenen Klassen. Jedoch ist die Größe an schwarzer Fläche bei

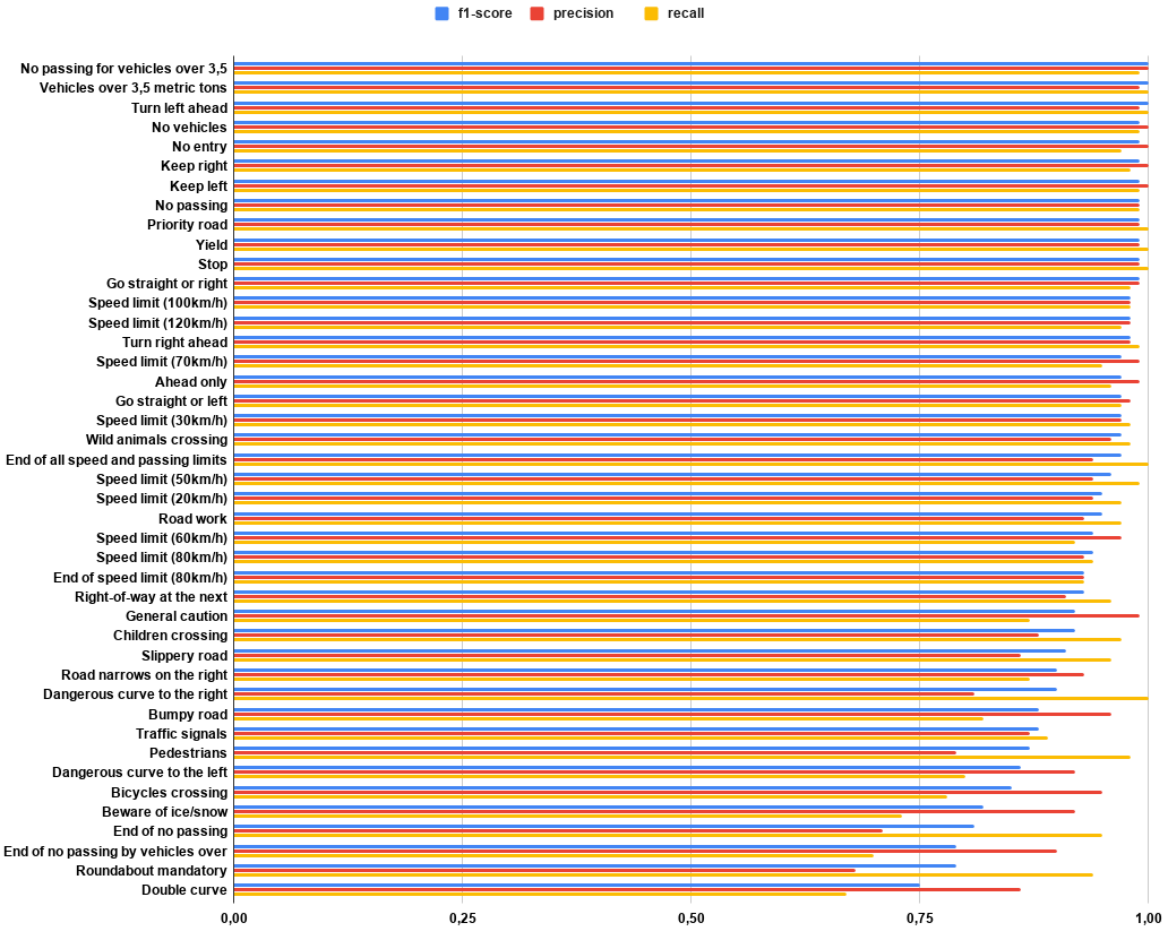

Abbildung 8: Das mit unbearbeiteten Trainings- und Testdaten trainierte und mit unbearbeiteten Testdaten evaluierte Netzwerk in der Genauigkeit absteigend nach f1-score sortiert

		Netzwerk 1	Netzwerk 2	Netzwerk 3	Netzwerk 4	Netzwerk 5	Netzwerk 6	Netzwerk 7
	Trainingsdaten	Original	Original	Abdeckend	Original	Schneefall	Original	Starkregen
	Testdaten	Original	Abdeckend	Abdeckend	Schneefall	Schneefall	Starkregen	Starkregen
Evaluations-daten	Original	**96%**						
	Abdeckend	2%	**86%**	84%				
	Schneefall	1%			45%	**72%**		
	Starkregen	5%					77%	**88%**

Abbildung 9: Genauigkeit (weighted avg. f1-score) der trainierten CNNs

Abbildung 10: Auszüge aus der Metadaten der GTSRB zu den diskutierten Klassen

“Vehicles over 3,5 metric tons prohibited” in der Mitte des Bildes übermäßig groß und die blaue Fläche der Klassen wie “Turn left ahead” sehr einzigartig, sodass diese gut im Abschnitt mit dem 5x5 Kernel des neuronales Netzes erkannt werden können.

Das schlechte Abschneiden von “Achtung”-Schildern lässt sich mit ihren Ähnlichkeiten zueinander und einer geringeren Vertretung im Datensatz erklären.
Die hohen Disparitäten einiger Klassen zwischen ihrem Recall- und dem Precisionwert korrelieren vor allem mit einer geringen Anzahl im Datensatz.

Der frühere Early-Dropout bei den mit den simulierten Testdaten trainierten Netzen erklärt sich dadurch, dass Störungen im Bild ein Netzwerk mit einer höheren Abstraktion benötigen. Overfitting tritt also verhältnismäßig früher als bei konsistenteren Trainingsdaten ein. Dies lässt sich vor allem an der schlechten Erkennungsrate des Netzwerk 1 bei der Klassifizierung der simulierten Datensätze in Tabelle? erkennen. Das Netzwerk ist bereits stark spezialisiert und geht auf feine Strukturen ein. Diese werden durch Störeinflüsse so stark beeinflusst, dass keine korrekte Klassifizierung mehr möglich ist.
Anders sieht es bei den Netzwerken 2,4 und 6 aus. Obwohl sie mit den gleichen Trainingsdaten trainiert wurden, schneiden sie deutlich besser ab. Dem liegt zugrunde, dass ein für die jeweilen Testdaten optimaler Early-Dropout gewählt wurde. Hierdurch befindet sich das Netzwerk noch in einer Phase, in der feine Strukturen nicht optimal antrainiert wurden und Störungen dadurch weniger ins Gewicht fallen. Im Fall des Datensatzes mit den teilweise verdeckten Bildern schneidet das Netzwerk 2 mit einer Erkennungsrate von 86% dabei sogar am besten ab. Dies könnte daran liegen, das durch ein Training mit den originalen Datensätzen am meisten Informationen enthalten sind und das Lernen dadurch optimiert wird.
Die Netzwerke 3, 5 und 7 wurden je mit ihren simulierten Datensätzen trainiert und auch getestet. Die Netzwerke 5 und 7 erzielen dabei auch die besten Ergebnisse in ihrer

Kategorie mit jeweils 72% und 88%. Diese Netzwerke haben also gelernt die Störeinflüsse in Form von Schnee und Regen herauszufiltern und sich damit auf die wesentlichen Information zur Klassifizierung zu konzentrieren.

8 Zusammenfassung und zukünftige Arbeiten

Diese Arbeit hat gezeigt, dass es sinnvoll ist spezielle Netzwerke für bestimmte Arten von schwierigen Verhältnissen zu trainieren. Dabei ist es nicht immer der optimale Weg, die Datensätze dieser schwierigen Verhältnisse auch als Trainingsdatensatz zu verwenden. Zusätzlich ist deutlich geworden, dass ein verfrühter Early-Dropout sinnvoll sein kann, auch wenn sich ein Netzwerk noch verbessert. Falls in der Realität noch unentdeckte und im Testdatensatz nicht enthaltene Situationen anzutreffen sind, hilft dies, um eine höhere Abstraktionsfähigkeit zu bewahren.

Im Vergleich zu der Sliding-Window Methode scheint ein CNN anfälliger für solche Störeinflüsse zu sein. Es ist allerdings noch unbekannt, wie ein Sliding-Window Ansatz bei diesen speziell generierten Datensätzen abschneiden würde. Eine zukünftige Arbeit könnte sich daher mit diesem Vergleich konkret beschäftigen und für beide Methoden denselben Datensatz verwenden.
Eine weitere Arbeit könnte sich damit beschäftigen einen Klassifikator für die Art der Störung in den generierten Datensätzen zu entwerfen. In einem gemischten Testdatensatz, in welchem Bilder mit Schnee, Regen und anderen Störungen den Bildern mit optimalen Bedingungen untergemischt sind, kann es sinnvoll sein zuerst die Art der Störung zu klassifizieren, um das Verkehrsschild dann mit dem jeweils besten CNN zu klassifizieren.

Literatur

[1] *https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign*

[2] C. Farabet, P. Akselrod S. Talay Y. L. ; Culurciello, E.: *Hardware accelerated convolutional neural networks for synthetic vision systems.* 2010

[3] Fleyeh, H.: *Traffic sign recognition without color information.* 2015

[4] J. Torresen, J. W. B. ; Sekanina, L.: *Efficient recognition of speed limit signs.* 2004

[5] Nguwi, Y.-Y ; Kouzani, A.: *Detection and classification of road signs in natural environments. Neural Computing and Applications.* 2008

[6] Nitish Srivastava, Alex Krizhevsky Ilya Sutskever Ruslan S.: *Dropout: A Simple Way to Prevent Neural Networks from Overfitting.* 2014

[7] R. S. Vaddi, H. D. V. ; Anne, K. R.: *Comparative analysis of contrast enhancement techniques between histogram equalization and CNN.* 2011

[8] Sermanet, P. ; LeCun, Y.: *Traffic sign recognition with multi-scale Convolutional Networks.* 2011