



ulm university universität
ulm

Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Institut für Mess-, Regel- und Mikrotechnik

Segmentierung von Punktwolken mit neuronalen Netzen

Masterarbeit

von

Tarik Enderes

31.12.2001

Betreuer: Dr.-Ing. Vasileios Belagiannis
1. Prüfer: Prof. Dr.-Ing. Brian F. Smith
2. Prüfer: Prof. Rivera

Hiermit versichere ich, dass ich die vorliegende Arbeit mit dem Titel

Segmentierung von Punktwolken mit neuronalen Netzen

bis auf die offizielle Betreuung selbstständig und ohne fremde Hilfe angefertigt habe und die benutzten Quellen und Hilfsmittel vollständig angegeben sind. Aus fremden Quellen direkt oder indirekt übernommene Gedanken sind jeweils unter Angabe der Quelle als solche kenntlich gemacht.

Ich erkläre außerdem, dass die vorliegende Arbeit entsprechend den Grundsätzen guten wissenschaftlichen Arbeitens gemäß der „Satzung der Universität Ulm zur Sicherung guter wissenschaftlicher Praxis“ erstellt wurde.

Ulm, den 31.12.2001

Tarik Enderes

Inhaltsverzeichnis

1 Literatur	1
1.1 Verwandte Arbeiten	1
1.1.1 PointNet	1
1.1.2 UPSNet	1
2 Theorie	3
2.1 Segmentierung	3
2.1.1 Semantische Segmentierung	3
2.1.2 Instanz-Segmentierung	3
2.1.3 Panoptische Segmentierung	4
2.2 Technologien in DeepLab	4
2.2.1 Deep Convolutional Neural Networks für Semantische Segmentierung	5
Convolutional Neural Networks	5
Anpassungen für Semantische Segmentierung	5
2.2.2 Atrous Convolution	5
2.2.3 Atrous Spatial Pyramid Pooling	6
2.2.4 Fully-Connected Conditional Random Fields	8
2.2.5 Residual Networks	8
2.3 Kamerakalibrierung	8
3 Arbeitsmethodik und Entwicklung	9
3.1 Integration von DeepLab	9
3.2 Backbones	9
3.2.1 Xception	9
3.2.2 MobileNetV2	9
4 Datensätze	11
4.1 Cityscapes	11
4.2 KITTI	11
5 Experimente	13
5.1 Technische Daten des für die Experimente verwendeten Rechners . . .	13

5.2	Backbones	13
5.2.1	MobileNetV2	13
5.2.2	Xeption65	18
5.3	Experimente mit Verfeinerung	19
5.3.1	Verfeinerung mit KITTI	19
5.4	Aufgetretene Probleme und Lösungen	21
5.4.1	False Positives	21
5.4.2	Overfitting	21
6	Zusammenfassung	25
	Literaturverzeichnis	27

1 Literatur

1.1 Verwandte Arbeiten

1.1.1 PointNet

1.1.2 UPSNet

2 Theorie

2.1 Segmentierung

Segmentierung bezeichnet einen Vorgang, bei dem ein Bild nach bestimmten Homogenitätskriterien in inhaltlich zusammenhängende Regionen einzuteilen. Von den verschiedenen Ansätzen, die das erreichen sollen, befasst sich diese Arbeit mit pixelbasierten Verfahren, bei denen jedem Pixel in einem Bild eine Klasse zugeordnet wird. Man unterscheidet, wie in [ups] beschrieben, semantische Segmentierung, Instanz-Segmentierung und panoptische Segmentierung.

2.1.1 Semantische Segmentierung

Bei der Semantischen Segmentierung soll jeder Pixel eine valide Klasse erhalten. Es wird dabei nicht zwischen unterschiedlichen Instanzen einer Objektklasse unterschieden. Wenn beispielsweise auf einem Bild zwei Fahrzeuge zu sehen sind und bei der Segmentierung die Klasse "Fahrzeug" aufgeteilt werden soll, erhalten die Pixel beider Fahrzeuge das Label "Fahrzeug". Die Anzahl valider Klassen bleibt somit bei jeden prozessierten Bild gleich.

2.1.2 Instanz-Segmentierung

Im Gegensatz zur semantischen Segmentierung werden bei der Instanz-Segmentierung nurzählbare Objekte betrachtet und deren Instanzen berücksichtigt. Übertragen auf vorheriges Beispiel würden die Pixel des einen Fahrzeug ein Label wie "Fahrzeug1" und die des anderen analog "Fahrzeug2" erhalten.

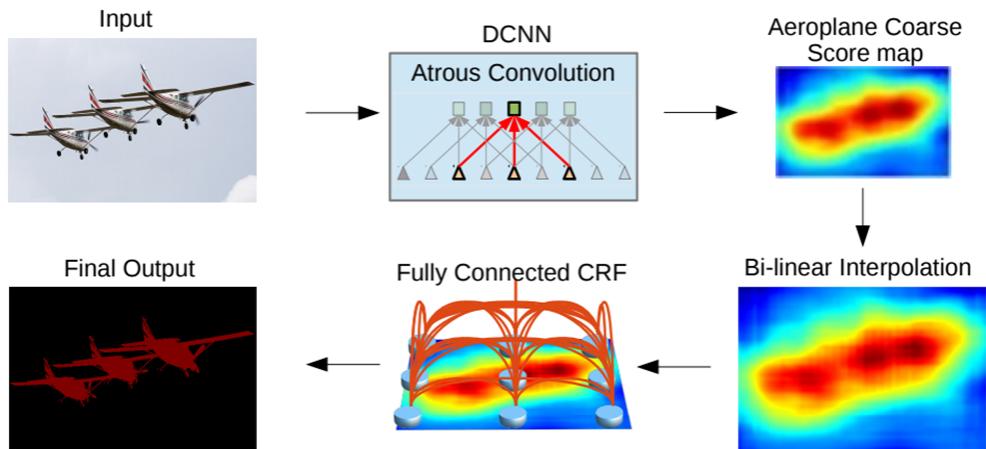


Abbildung 2.1: Grundsätzliche Funktionsweise von DeepLab

2.1.3 Panoptische Segmentierung

Die panoptische Segmentierung stellt eine Kombination der vorherigen Segmentationsarten dar. Zählbare Objekte werden demnach nach dem Prinzip der Instanz-Segmentierung und amorphe nach dem der semantischen Segmentierung segmentiert. Die Ergebnisse beider Verfahren werden anschließend kombiniert.

2.2 Technologien in DeepLab

DeepLab ist ein von Google entwickeltes, 2015 in [dl1] vorgestelltes Modell für semantische Segmentierung. Bei der in [dl2] vorgestellten Methode wird ein Deep Convolutional Neural Network (DCNN) zum Erzeugen einer Score Map benutzt, die anschließend mit einem Conditional Random Field (CRF) zur endgültigen Ausgabe weiterverarbeitet wird. Das Verfahren wird in Abbildung 2.1 grob dargestellt.

2.2.1 Deep Convolutional Neural Networks für Semantische Segmentierung

Convolutional Neural Networks

Anpassungen für Semantische Segmentierung

Klassische DCNNs haben Eigenschaften, die sie für die Verwendung zur Bildsegmentierung nicht ideal machen.

- Der Einsatz von Downsampling führt zu verringriger Auflösung, die bei Klassifizierungsaufgaben nicht ins Gewicht fällt, für die Segmentierung aber essentiell ist.
- Neuronale Netze sind in der Regel gut geeignet, um Objekte unterschiedlicher Größe zu erkennen, wenn solche in der Lernphase präsentiert werden. Die Eigenschaften der Faltung, insbesondere dem begrenzten Sichtbereich beim Berechnen eines einzelnen Pixels ist allerdings für diese Problematik ungünstig.
- Der wiederholte Einsatz von Convolutional Layers führen zu einem Verlust an Ortsinformation. Infolgedessen produzieren DCNNs bei Segmentierungsaufgaben verschwommene, oft verrauschte Ergebnisse ohne klare Kanten.

Um diese Probleme zu lösen erhält das von DeepLab verwendete DCNN einige Anpassungen. Zunächst werden alle Fully Connected Layers durch Convolutional Layers ersetzt, um ein Fully Convolutional Network zu bilden. Noch dazu wird anstatt von Pooling Layers in den unteren Schichten Atrous Convolution eingesetzt, womit die Auflösung der Ausgabe erhöht wird. In den höheren Schichten werden auch hier Pooling Layers eingesetzt, um Speicherbedarf und Rechenzeit zu verbessern. Um die Größeninvarianz zu verbessern wird bei den unteren Schichten Atrous Spatial Pyramid Pooling verwendet.

2.2.2 Atrous Convolution

Atrous Convolution, auch Dilated Convolution genannt, beschreibt eine Technik bei der eine Matrix mit einem spärlich bestückten Kernel gefaltet wird, wie in Abbildung 2.2 illustriert.

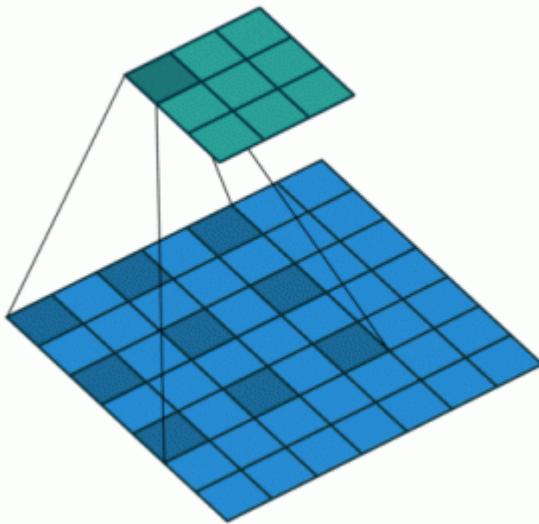


Abbildung 2.2: Prinzip von Atrous Convolution

Die Abstände der zu berücksichtigenden Werte in der Matrix wird dabei durch die s.g. Dilation Rate bzw. Erweiterungsrate (kurz Rate) festgelegt. Das Tatsächliche Sichtfeld des Filters wird also festgelegt durch die Größe des Kernels und die Rate bestimmt.

ein Filter mit einem Kernel der Größe 3x3 und einer Rate von 2, was dem Einfügen einer leeren Zeilen und Spalte zwischen den Werten entspricht, hat demnach ein Sichtfeld der Größe 5x5. Dadurch wird das effektive Sichtfeld des Filters erhöht und es kann eine höhere Auflösung bei gleichen Rechenaufwand erreicht werden. Die Vorteile der Verwendung von Atrous Convolution für Bildsegmentierung sind in Abbildung 2.3 dargestellt.

2.2.3 Atrous Spatial Pyramid Pooling

Beim Atrous Spatial Pyramid Pooling werden mehrere parallele Convolutional Layers, die Atrous Convolutional Layers mit unterschiedlicher Rate verwenden, in das DCNN eingebaut. Das Prinzip ist in Abbildung 2.4 dargestellt.

Durch dieses Vorgehen soll Größeninvarianz erreicht werden.

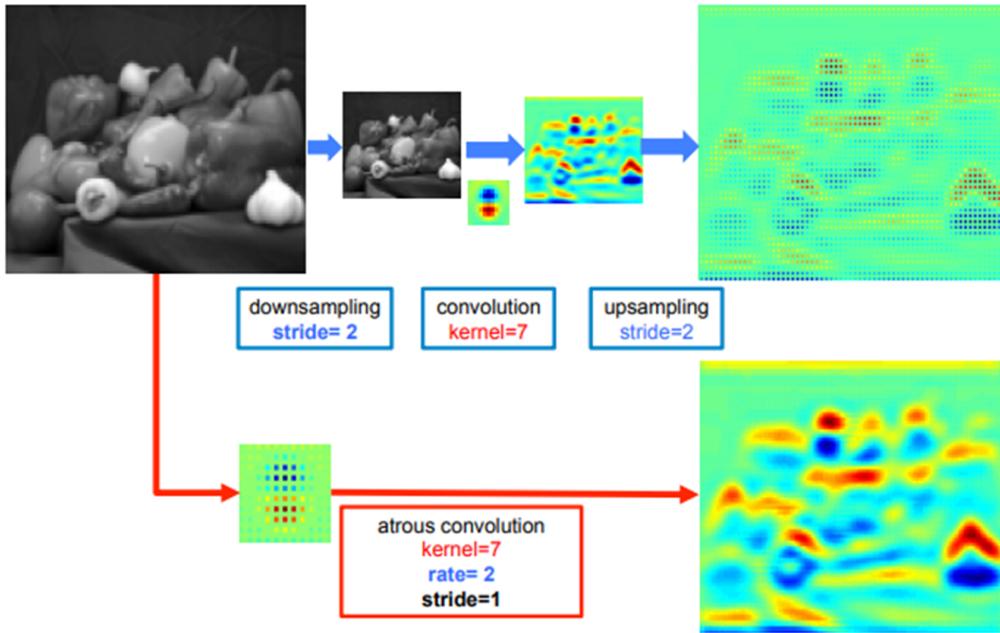


Abbildung 2.3: Beispielhaft dargestellte Vorteile von Atrous Convolution

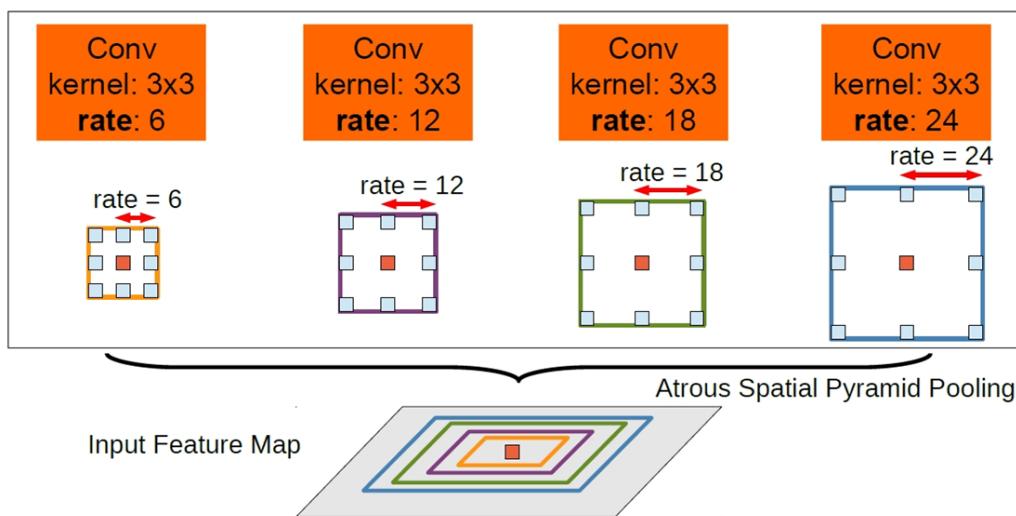


Abbildung 2.4: Beispielhaft dargestellte Vorteile von Atrous Convolution

2.2.4 Fully-Connected Conditional Random Fields**2.2.5 Residual Networks****2.3 Kamerakalibrierung**

3 Arbeitsmethodik und Entwicklung

3.1 Integration von DeepLab

3.2 Backbones

3.2.1 Xception

3.2.2 MobileNetV2

4 Datensätze

4.1 Cityscapes

4.2 KITTI

5 Experimente

5.1 Technische Daten des für die Experimente verwendeten Rechners

- Prozessor: Intel(R) Core(TM) i7-7700HQ CPU @ 2.860GHz
- Grafikkarte: NVIDIA GeForce GTX 1070
- Arbeitsspeicher: 16GB RAM

5.2 Backbones

Für die Durchführung folgender Experimente wird der fein annotierte Cityscapes-Datensatz verwendet. Die Netze werden auf dem aus 2975 Bildern bestehenden Trainingssatz trainiert und auf dem 500 Bilder fassenden Validierungssatz ausgewertet. Für jede Trainingsepoke werden aus dem Trainingssatz 1487 Bilder zufällig ausgewählt. Als Grundlernrate wird 0.007 gewählt. Der Trainingsfehler wird über die in [lov] beschriebene Lovasz-Funktion berechnet. Weitere Hyperparameter sind eine Dropout-Rate von 0.1, eine L2 Regularisierungsrate von $4 * 10^{-5}$ und ein Momentum-Faktor von 0.9.

5.2.1 MobileNetV2

Da Echtzeitfähigkeit eine wichtige Rolle spielt, konzentrieren sich die Experimente auf das leichtgewichtige MobileNetV2, statt dem leistungsfähigeren Xception65.

Wie in Tabelle 5.1 und Abbildung 5.1 zu erkennen ist, verbessert sich die Bewertung

Epochen trainiert	1	5	10	15	25	30
IoU Straße	0.6260	0.6498	0.6727	0.6729	0.6774	0.6675
IoU Gehsteig	0.1835	0.3788	0.4330	0.4616	0.5193	0.5078
IoU Gebäude	0.5563	0.6402	0.6573	0.6771	0.6931	0.7068
IoU Mauer	0.0	0.0	0.0	0.0	0.0	0.0
IoU Zaun	0.0	0.0	0.0	0.0	0.0	0.0
IoU Pfahl	0.0791	0.1931	0.2159	0.2561	0.2835	0.2728
IoU Ampel	0.0	0.0	0.0	0.0	0.0	0.0
IoU Verkehrszeichen	0.0	0.1056	0.1785	0.1859	0.2233	0.2291
IoU Vegetation	0.5581	0.7153	0.7273	0.7518	0.7728	0.7663
IoU Gelände	0.0	0.0915	0.1274	0.1355	0.1593	0.1445
IoU Himmel	0.5353	0.6606	0.6867	0.6977	0.7153	0.7081
IoU Person	0.0	0.1160	0.1582	0.1458	0.1680	0.1863
IoU Radfahrer	0.0	0.0	0.0	0.0	0.0	0.0
IoU Auto	0.3537	0.5438	0.6014	0.5863	0.6419	0.6103
IoU Lastwagen	0.0	0.0	0.0	0.0	0.0	0.0
IoU Bus	0.0	0.0	0.0	0.0	0.0	0.0
IoU Zug	0.0	0.0	0.0	0.0	0.0	0.0
IoU Motorrad	0.0	0.0	0.0	0.0	0.0	0.0
IoU Fahrrad	0.0	0.0	0.0	0.0750	0.1340	0.1542
Durchschnitt IoU	0.1522	0.2155	0.2346	0.2444	0.2625	0.2607
Durchschnitt IoU ≠ 0	0.4131	0.4094	0.4458	0.4222	0.4534	0.4503

Tabelle 5.1: Bewertung von Models in Intersection-over-union (IoU) Metrik in Abhängigkeit der Trainingsdauer. Berechnung des IoU durch:
 $\text{IoU} = \text{True Positives} / (\text{True Positives} + \text{False Positives} + \text{False Negatives})$

des Models bis zu einem Punkt, der in etwa bei Epoche 25 liegt und verschlechtert sich bei Verlängerung der Trainingsdauer wieder. Es tritt also trotz der L2 Regularisierung der Netzparameter und der Nutzung des Dropout-Verfahrens wahrscheinlich Overfitting auf. Im Folgenden wird das für 25 Epochen trainierte Netz verwendet. Die Bewertung dieses ist in Abbildung 5.2 dargestellt.

Die Models weisen durchwegs ihre höchsten Ergebnisse beim Erkennen der amorphen Klassen Straße, Gebäude, Vegetatio und Himmel auf, wie bei einem semantischen Segmentierungsverfahren zu erwarten. Das niedrige Ergebnisse bei der Klasse Gelände lässt sich dadurch erklären, dass der Cityscapes-Datensatz in städtischen Umgebungen aufgenommen wird, wo diese Klasse selten auftritt. Vergleichsweise hoch ist auch der IoU-Wert in der Klasse Auto, die in dem Datensatz besonders häufig ist.

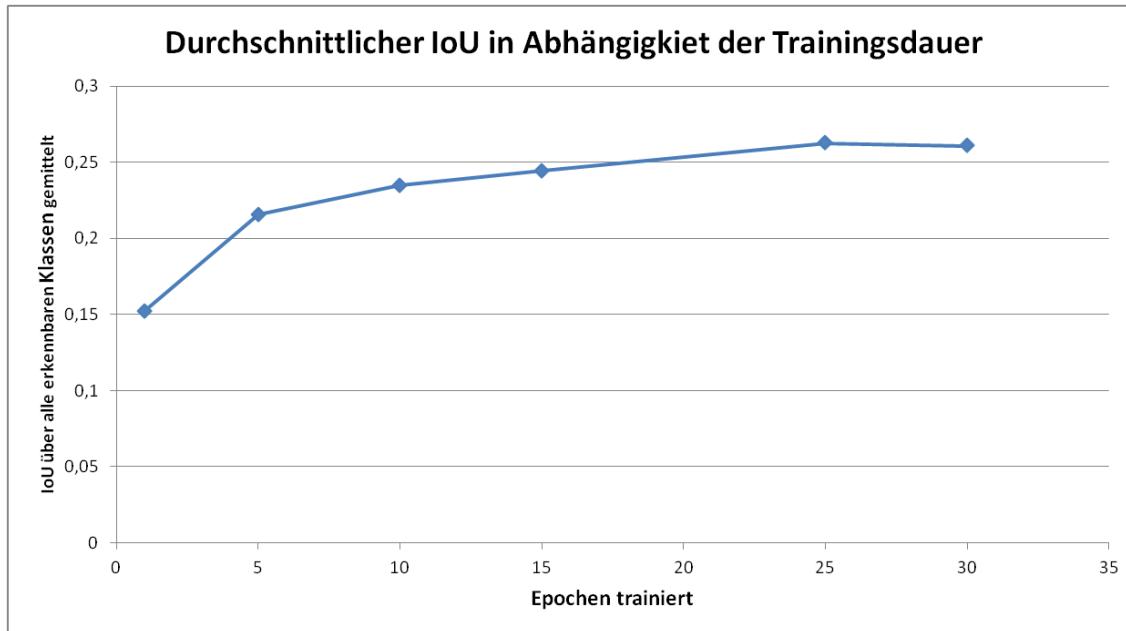


Abbildung 5.1: Durchschnittlicher IoU in Abhängigkeit der Anzahl trainierter Epochen

Die niedrigsten positiven IoU-Werte weisen die Ergebnisse bei kleineren,zählbaren Objekten wie Personen, Pfähle, Fahrräder und Verkehrszeichen auf.

Die in Abbildung 5.2 dargestellten Ergebnisse lassen außerdem erkennen, dass das Netz bestimmte Klassen praktisch nicht erkennt. Dies lässt vermuten, dass es nur eingeschränkt fähig ist, Bildsegmente anhand ihres Kontextes zu bewerten und beispielsweise zwischen einem Fußgänger und einem Fahrradfahrer oder zwischen einer Mauer und einem Gebäude zu unterscheiden und die häufiger auftretende Variante auswählt. Die Ergebnisse der Klasse Fahrrad lassen vermuten, dass ein längeres Training dieses Verhalts verbessern könnte. Da ein zu langes Training sich, wie vorher erwähnt, negativ auf den durchschnittlichen IoU auswirkt, wird in diesem Experiment aber davon abgesehen. Eine weitere Möglichkeit wäre, dem Trainingssatz mehr Daten hinzuzufügen, die vermehrt die entsprechenden Objekte enthalten.

5.3 zeigt ausgewählte Ausgaben des für 25 Epochen lang trainierten Netzes, das die besten Ergebnisse in der IoU-Metrik liefert. Sie spiegeln die Bewertung in 5.1 mit hoher Genauigkeit bei großen und amorphen Objekten und niedriger bei kleineren,zählbaren. Besonders auffällig sind False Positives der Klasse Person, die das Modell oft an Fahrräder oder Bäume in der Nähe von Personen vergibt.

Die Beispiele lassen erkennen, dass das Netz teilweise die Trainingsdaten memorisiert. So wird dem oberen Teil eines Fahrrades häufig die Klasse Person zugewiesen und

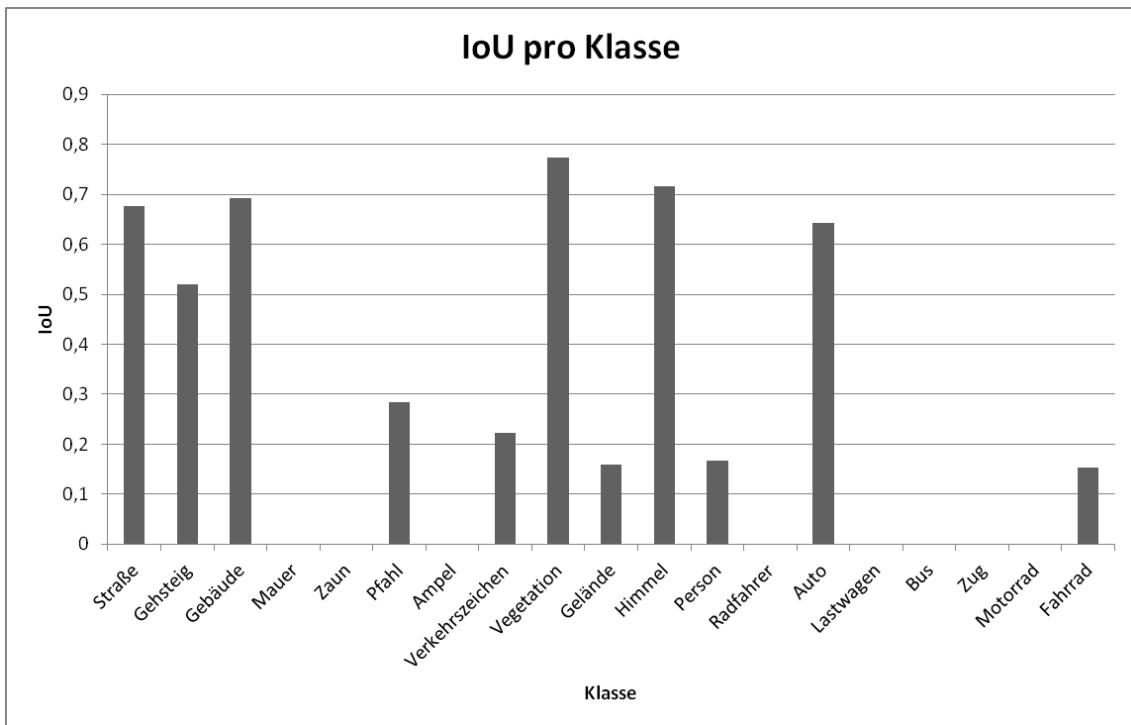


Abbildung 5.2: IoU für jede Klasse vom Test des für 25 Epochen trainierten Netzes

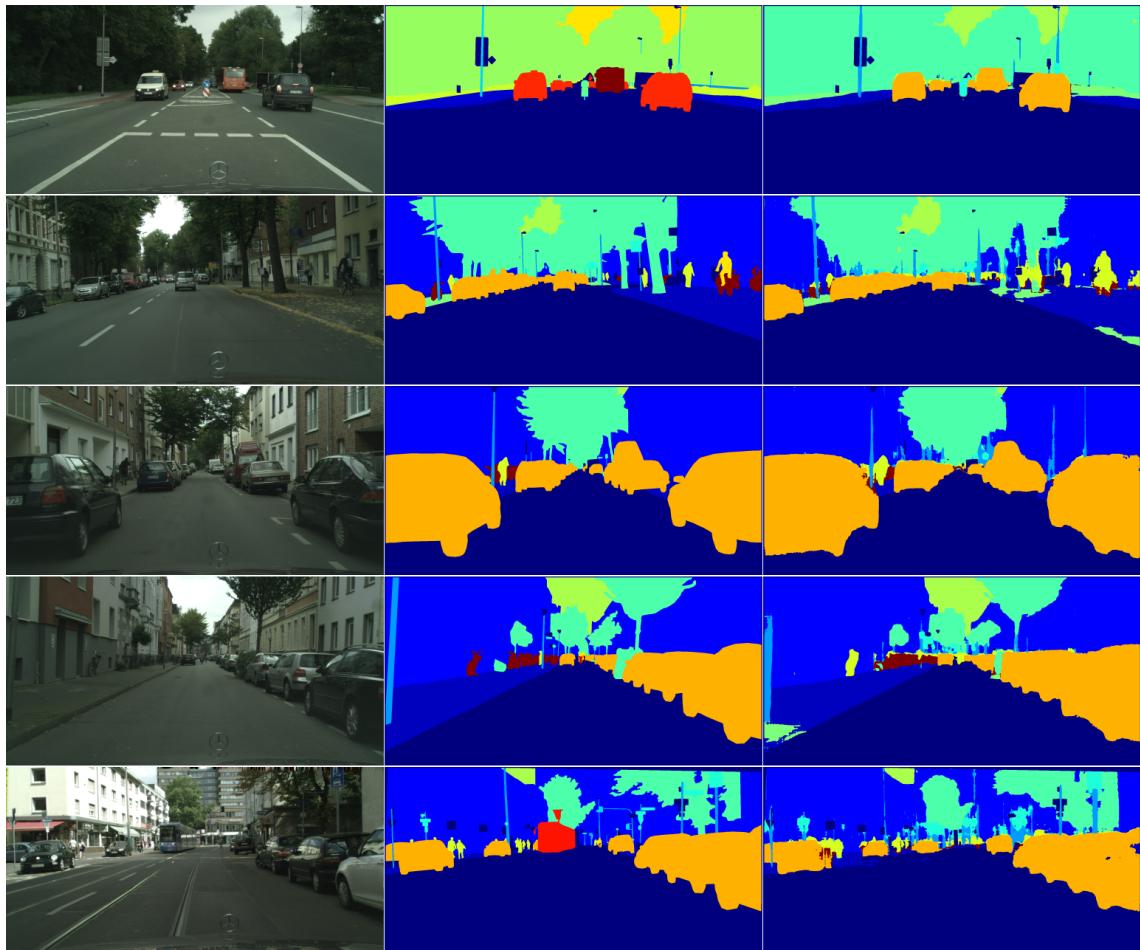


Abbildung 5.3: Beispiele für Ausgaben des Models (25 Epochen trainiert)
Von links nach rechts: Eingabebild, Ground Truth, Ausgabe von DeepLab

umgekehrt der untere Teil einer Person als Fahrrad erkannt. Genauso werden Bereiche, die sich über einem als Straße erkannten Segments befinden tendenziell eher als Auto klassifiziert.

5.2.2 Xception65

Wir betrachten den Xception65-Backbone im Vergleich zu MobileNetV2. Auf ein Experiment mit unterschiedlichen Trainingsepochen wird an dieser Stelle aufgrund der langen Trainingsdauer der Netze verzichtet. Das verwendete Model ist 60 Epochen lang trainiert mit denselben Hyperparametern wie die Models mit MobileNetV2.

	Xception65	MobileNetV2	Differenz
IoU Straße	0.6951	0.6774	0.0177
IoU Gehsteig	0.7238	0.5193	0.2045
IoU Gebäude	0.8533	0.6931	0.1602
IoU Mauer	0.1461	0.0	0.1461
IoU Zaun	0.1641	0.0	0.1641
IoU Pfahl	0.5843	0.2835	0.3008
IoU Ampel	0.3049	0.0	0.3049
IoU Verkehrszeichen	0.6592	0.2233	0.4359
IoU Vegetation	0.8558	0.7728	0.0830
IoU Gelände	0.2068	0.1593	0.0475
IoU Himmel	0.7707	0.7153	0.0554
IoU Person	0.5234	0.1680	0.3554
IoU Radfahrer	0.2386	0.0	0.2386
IoU Auto	0.8369	0.6419	0.1950
IoU Lastwagen	0.0691	0.0	0.0691
IoU Bus	0.0893	0.0	0.0893
IoU Zug	0.0185	0.0	0.0185
IoU Motorrad	0.0685	0.0	0.0685
IoU Fahrrad	0.4080	0.1340	0.4080
Durchschnitt IoU	0.4324	0.2625	0.2346
Durchschnittliche Verarbeitungszeit [ms]	764	387	377

Tabelle 5.2: Vergleich zwischen Xception65 und MobileNetV2 in IoU-Metrik und Verarbeitungszeit

Wie in Tabelle 5.2 und Abbildung 5.4 zu sehen ist, erzeugt das Model mit Xception65 durchwegs bessere Ergebnisse als das mit MobileNetV2, benötigt aber im Durchschnitt

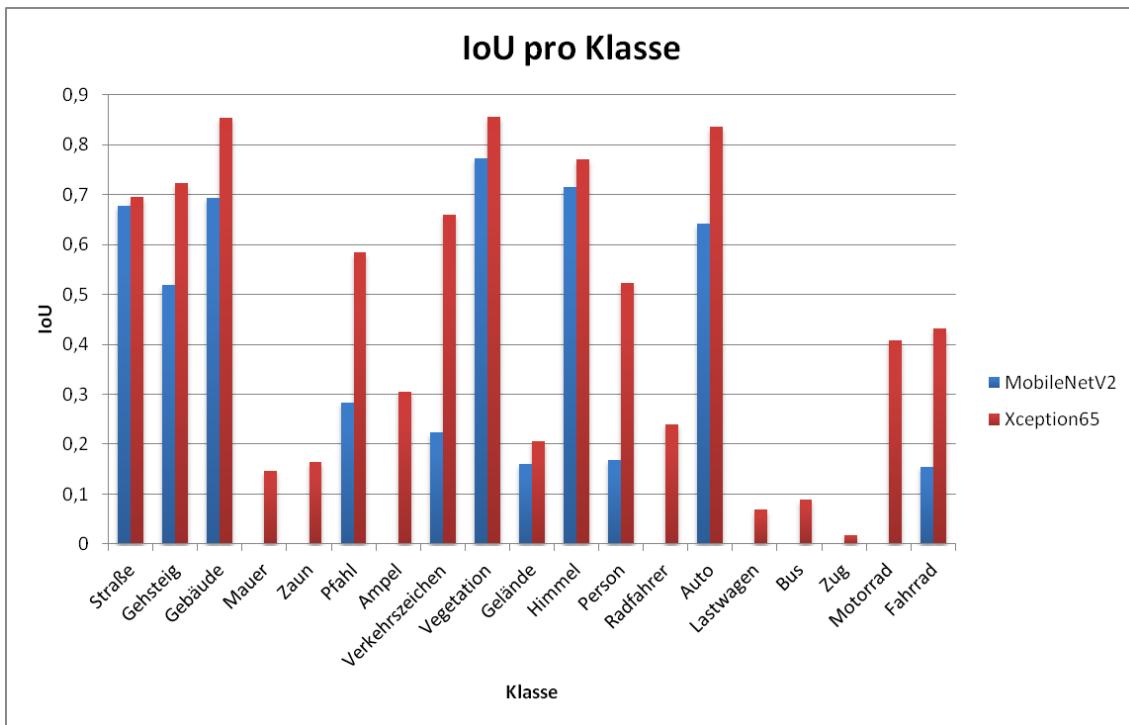


Abbildung 5.4: Vergleich Xception65 und MobileNetV2 in IoU-Metrik

97% mehr Zeit für die Verarbeitung eines Eingabebildes.

Durch die Verwendung von Xception65 ist das Model außerdem in der Lage, alle Klassen des Datensatzes zu erkennen, auch wenn die IoU-Werte für die von MobileNetV2 nicht erkannten Klassen vergleichsweise niedrig sind. Eine besonders große Steigerung der Bewertung im Vergleich mit MobileNetV2 weist das Netz bei kleineren, zählbaren Objekten auf wie Personen (311%) und Verkehrszeichen (295%).

Abbildung 5.5 zeigt beispielhafte Ergebnisse des Models aus dem Validierungs-Datensatz von Cityscapes.

5.3 Verfeinerung mit KITTI

In diesem Experiment wird das am besten bewertete, mit Cityscapes trainierte Model mit MobileNetV2 als Backbone mehrere Epochen mit den 200 Bildern umfassenden Trainingsdaten für semantische Segmentierung von KITTI nachtrainiert. Die Resultierenden Models werden anschließend mit anderen Bildern aus dem KITTI-Datensatz getestet. Da KITTI nur für diese 200 Bilder eine Ground Truth zur Verfügung stellt und ein Test auf den Trainingsdaten wenig aussagekräftig ist, ist es nicht möglich, die

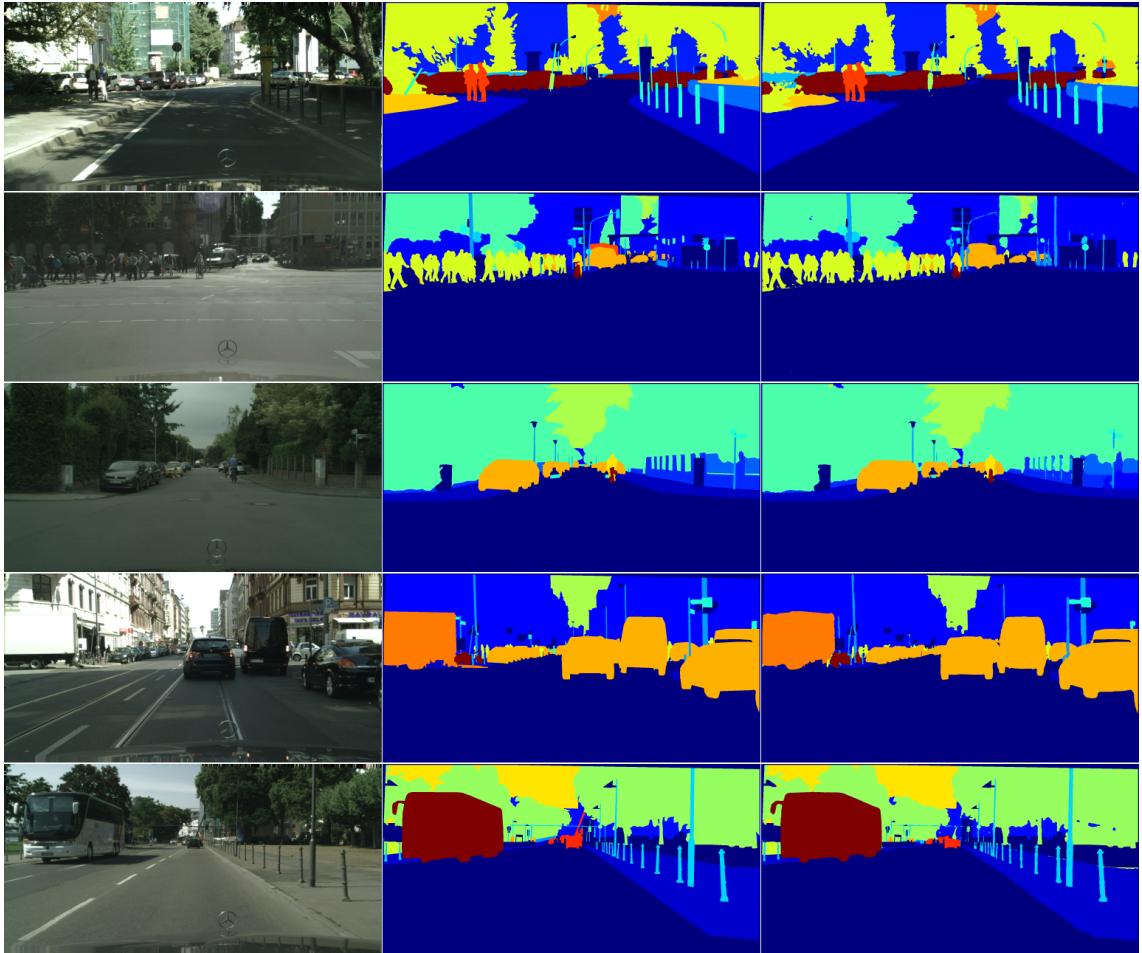


Abbildung 5.5: Beispiele für Ausgaben des Models

Von links nach rechts: Eingabebild, Ground Truth, Ausgabe von DeepLab

Ergebnisse mit der IoU-Metrik zu bewerten. Das Experiment begnügt sich deshalb mit einer empirischen Auswertung.

Abbildung 5.6 zeigt beispielhaft je ein Bild aus beiden Datensätzen und die Ausgaben von DeepLab für verschiedene lang nachtrainierte Modelle. Wie man sieht, tritt bei den Ergebnissen auf dem KITTI-Datensatz bereits nach einer Epoche Verfeinerung mit KITTI-Daten eine deutliche Verbesserung auf. Weitere Trainingsepochen verbessern die Ergebnisse weiter, aber weniger erheblich. Gleichzeitig verschlechtert sich die Ausgabe für ein Bild aus dem Cityscapes-Datensatz in ähnlichem Maße.

Das unterste Bild von Abbildung 5.6 zeigt die Ausgabe eines Modells, das für 25 Epochen auf Cityscapes-Daten trainiert, für 5 Epochen mit KITTI-Daten verfeinert und anschließend für eine Epoche mit Cityscapes-Daten nachtrainiert ist. Wie man sieht, führt die Verfeinerung mit dem ursprünglichen Datensatz wieder zu einer Verschlechte-

rung der Ergebnisse auf KITTI-Bildern und einer Verbesserung auf Cityscapes-Bildern.

5.4 Aufgetretene Probleme und Lösungen

5.4.1 False Positives

Models, die MobileNetV2 benutzen neigen dazu, einige Klassen zum Großteil mit einer anderen zu klassifizieren. Beispiele dafür sind die bereits angesprochene Erkennung von Fahrrädern als Personen und die Klassifizierung großer Fahrzeuge als Gebäude. Bei Verwendung des Xception65-Backbones kommt es reproduzierbar bei bestimmten Bildern zu einem Phänomen, bei dem eine große Anzahl Pixel nahe einer der Ecken als Straße klassifiziert wird. Es besteht kein erkennbarer Zusammenhang zwischen den Bildern, bei denen dieses Verhalten auftritt. Abbildung 5.7 zeigt Beispiele dafür.

5.4.2 Overfitting

Overfitting hat sich als eines der hauptsächlichen Probleme bei der Verwendung von MobileNetV2 herausgestellt. Schon kleine Änderungen in der Perspektive, wie sie beispielsweise im KITTI-Datensatz auftritt, verursachen eine spürbare Verschlechterung der Ergebnisse.

Um bessere Ergebnisse auf dem KITTI-Datensatz zu erzeugen, hat es sich als vorteilhaft herausgestellt, den Trainingssatz für semantische Segmentierung der KITTI-Daten beim Trainieren des Netzes mitzuberücksichtigen. Das Netz mit jenen Daten nachzutrainieren verbessert ebenfalls die Ergebnisse, kann aber wiederum zu Overfitting auf diesen Datensatz führen. Es kann hilfreich sein, nach dem Nachtraining noch eine Epoche mit allen verfügbaren Daten nachzutrainieren.

Weitere Möglichkeiten, gegen Overfitting vorzugehen, mit denen in dieser Arbeit aber nicht experimentiert wird, sind eine Erhöhung der Dropout-Rate und dem L2 Regularisierungsfaktor.

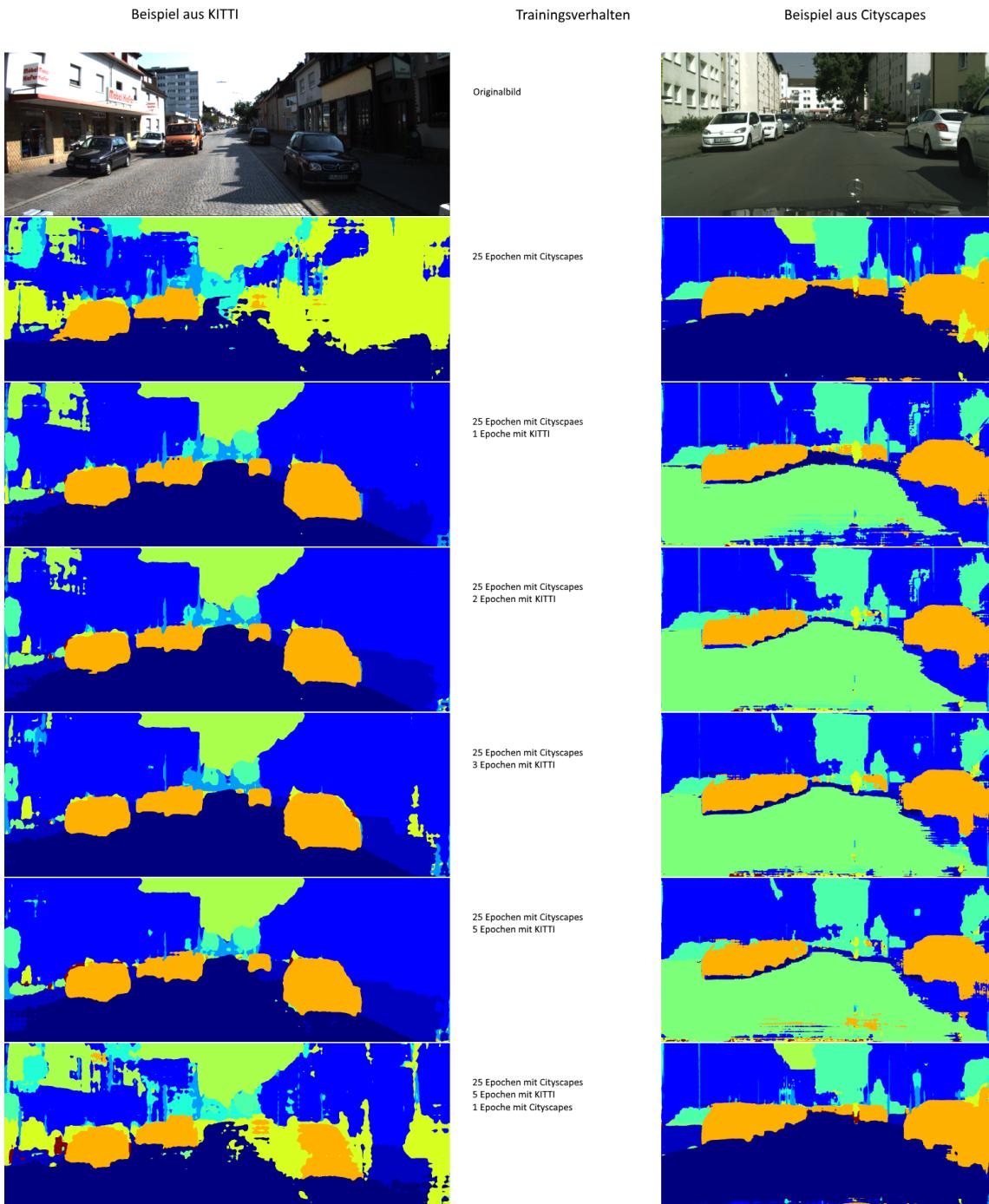


Abbildung 5.6: Beispiel für Ausgabe der verfeinerten Models

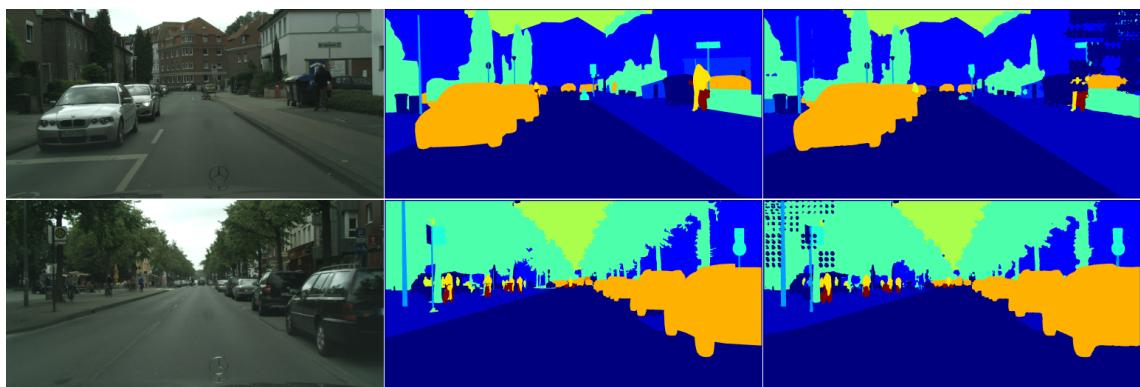


Abbildung 5.7: Beispiele für fehlerhafte Ausgaben des Models mit Xception65
Von links nach rechts: Eingabebild, Ground Truth, Ausgabe von DeepLab

6 Zusammenfassung

Literaturverzeichnis

- [dl1] Chen, Liang-Chieh; Papandreou, George; Kokkinos, Iasonas; Murphy, Kevin und Yuille, Alan L.: *Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs*. URL: <https://arxiv.org/pdf/1412.7062.pdf> (zuletzt besucht am 24.07.2019).
- [dl2] Chen, Liang-Chieh; Papandreou, George; Kokkinos, Iasonas; Murphy, Kevin und Yuille, Alan L.: *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*. URL: <https://arxiv.org/pdf/1606.00915.pdf> (zuletzt besucht am 24.07.2019).
- [lov] Berman, Maxim; Triki, Amal Rannen und Blaschko, Matthew B.: *The Lovasz-Softmax loss: A tractable surrogate for the optimization of the 'intersection-over-union' measure in neural networks*. URL: https://zpzascal.net/cvpr2018/Berman_The_LovaSz-Softmax_Loss_CVPR_2018_paper.pdf (zuletzt besucht am 05.08.2019).
- [ups] Xiong, Yuwen; Liao, Renjie; Zhao, Hengshuang u. a.: *UPSNet: A Unified Panoptic Segmentation Network*. URL: <https://arxiv.org/pdf/1901.03784.pdf> (zuletzt besucht am 24.07.2019).