Studienarbeit ComputerVision 2

Differences and Limitations of Unet vs. Segnet

Ostbayerische Technische Hochschule Amberg-Weiden

Michael Zimmet
m.zimmet@oth-aw.de

BKI7, Berichterstattung:

Inhaltsverzeichnis

1 Einleitung **II**

2 Architektur im Überblick **II**
2.1 Unet II
2.2 SegNet IV
2.3 Unterschiede IV

3 Training **V**
3.1 ISIC 2018 (Task 1) V
3.2 ISIC 2018 (Task 1) dezimierter Datensatz VI
3.3 OxfordIIITPet VII
3.4 Cityscapes VIII
3.5 Unterschiede VIII

4 Evaluierung **IX**
4.1 Globale Metriken IX
4.2 Qualitative Bewertung X

5 Fazit **XI**

A Anhang **XI**

1 Einleitung

Die Segmentierung von Bildern spielt eine wichtige Rolle in vielen Bereichen unserer Gesellschaft wie zum Beispiel im Bereich autonomes Fahren oder in der medizinischen Bildanalyse. Vor allem in der Medizin ermöglicht dies eine Identifikation / Abgrenzung von wichtigen Bereichen, wie beispielsweise Tumorgewebe. Mithilfe der automatisierten Segmentierung mit Deep Learning Methoden kann somit den Ärzten eine Möglichkeit geboten werden, diese zu unterstützen. Einige Datensätze für solche Segmentierungsaufgaben sind beispielsweise ISIC 2018, Cityscapes oder OxfordIIITPet. In diesem Bericht wird ein Vergleich zwischen zwei bekannten Architekturtypen Unet und SegNet durchgeführt. Beide Architekturen haben sich in verschiedenen Segmentierungsaufgaben als leistungsfähig erwiesen, weisen jedoch ebenso Unterschiede auf. Anhand des Vergleiches sollen die Unterschiede und Limitationen der beiden Architekturen aufgezeigt werden. U-Net wurde ursprünglich für die biomedizinische Bildsegmentierung entwickelt und zeichnet sich durch seine Encoder-Decoder-Struktur mit Skip-Connections aus. SegNet hingegen setzt auf eine Encoder-Decoder-Architektur, bei der die Pooling-Indizes aus der Downsampling-Phase zur Wiederherstellung der räumlichen Informationen im Upsampling verwendet werden. Die im Bericht verwendeten Modelle wurden eigens auf den drei oben beschriebenen Datensätzen trainiert. Der Aufbau der Architekturen wurde aus den Vorlesungsbeispielen übernommen, zudem wurden beide Modelle mit den gleichen Hyperparametern trainiert, um einen fairen Vergleich nur auf Basis der Architektur zu gewährleisten. Der Vergleich zwischen den Modellen erfolgt sowohl quantitativ anhand verschiedener Metriken als auch qualitativ durch die Visualisierung der Vorhersagen. Darüber hinaus werden die Grenzen beider Ansätze analysiert, um ihre jeweiligen Schwächen und Stärken besser zu verstehen. Alle Checkpoints zu den Modellen können dem folgenden Github-Projekt entnommen werden.

2 Architektur im Überblick

Für die Segmentierung von Bilddaten werden spezielle Netzarchitekturen benötigt, welche sowohl räumliche Informationen erhalten als auch eine präzise Klassifizierung auf Pixelebene ermöglichen. In diesem Abschnitt werden die beiden Architekturen Unet und SegNet jeweils beschrieben.

2.1 Unet

Unet wurde ursprünglich für die biomedizinische Bildsegmentierung entwickelt und ist insbesondere für Aufgaben mit kleinen Datensätzen geeignet. Die Architektur basiert auf einer symmetrischen Encoder-Decoder-Struktur mit Skip-Connections, die eine präzise Rekonstruktion räumlicher Informationen ermöglichen.

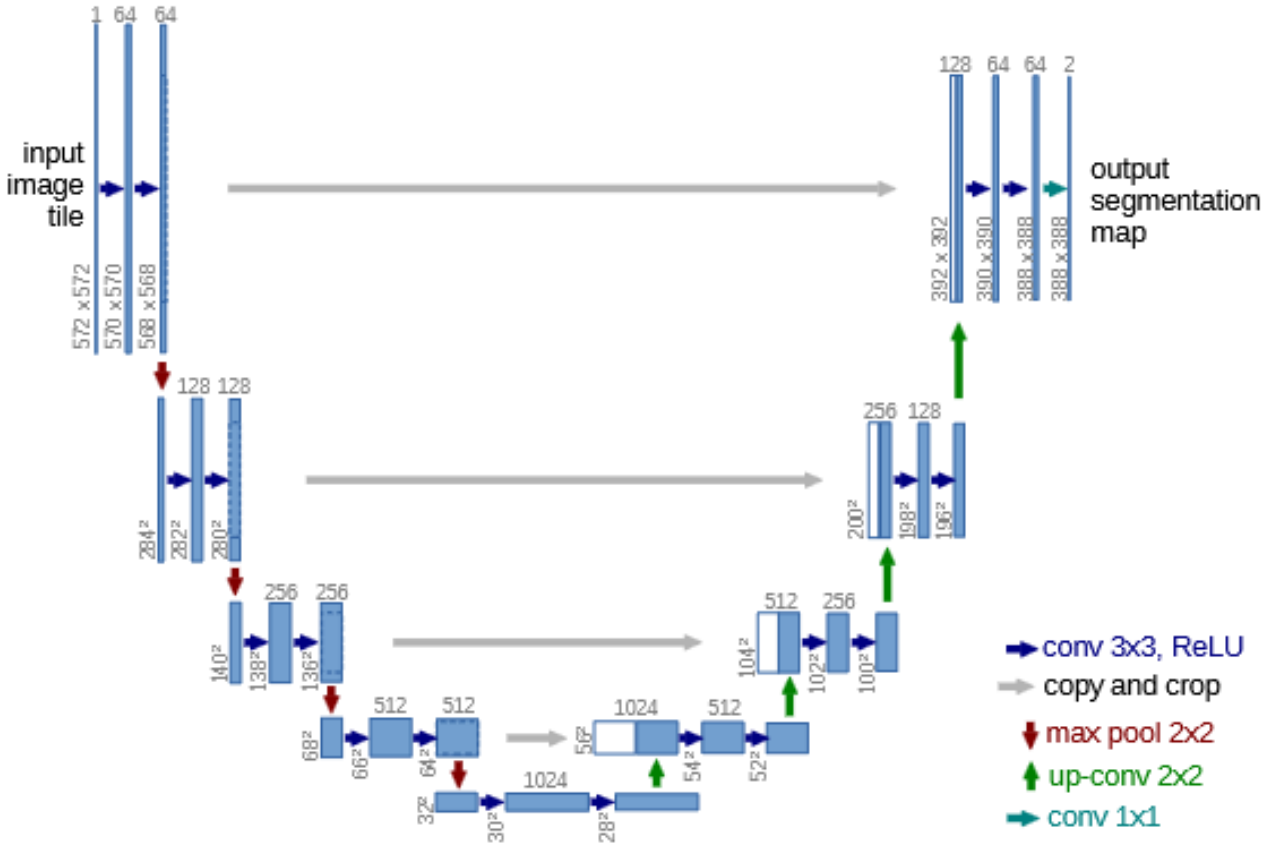

Abbildung 1: Architektur Unet

Die Architektur besteht dabei aus zwei Hauptpfaden: dem Encoder und dem Decoder. Der Encoder folgt einer typischen Architektur eines Convolutional Neural Networks (CNN), bestehend aus Convolutional Layers, ReLU-Aktivierungsfunktionen und Batch-Normalisierungsschichten. In diesem Vergleich besteht der Encoder-Pfad aus 4 Blöcken, wobei jeder Block aus einer zweifachen Anwendung einer 3x3-Convolution besteht, gefolgt von einer Batch-Normalisierung und einer ReLU-Schicht. Jeder Block wird mit einem 2x2-Max-Pooling-Layer abgeschlossen, der die Bildgröße reduziert. Die Aufgabe des Encoders ist es, wichtige Merkmale im Bild zu extrahieren. Aufgrund der Verkleinerung des Bildes können globale Informationen leichter erkannt werden. Die extrahierten Merkmale werden zwischengespeichert und dem Decoder-Pfad zur Verfügung gestellt, um das Bild im späteren Verlauf wieder zu rekonstruieren. Da durch die Verkleinerung die genauen Positionen der Merkmale verloren gehen, nutzt der Decoder zusätzliche Informationen vom Encoder, um diese Details wiederherzustellen. Der Decoder kombiniert die aus dem Encoder gewonnenen Merkmale mit den globalen Informationen aus den unteren Ebenen. Der Decoder besteht ebenfalls aus 4 Blöcken, wobei jeder Block mit einem Upsampling-Layer beginnt, um die Bildgröße schrittweise wiederherzustellen. Nach dem Upsampling folgen die gleichen Schichten wie im Encoder (zweifache Anwendung von 3x3-Convolution, Batch-Normalisierung und ReLU). Zwischen Encoder und Decoder existieren sog. Skip Connections, die es ermöglichen, die Details aus den hochauflösenden Ebenen des Encoders direkt an die entsprechenden Decoder-Ebenen weiterzugeben. Dadurch werden sowohl lokale als auch globale Informationen effizient kombiniert. Abgeschlossen wird die Architektur durch einen zusätzlichen Block, der weder Max-Pooling- oder Upsamplingoperationen enthält, sondern lediglich Encoder und Decoder miteinander verbindet. Insgesamt erlangte das Neuronale Netz dabei eine Parameteranzahl von 7.763.041 Parametern.

2.2 SegNet

SegNet wurde speziell für die semantische Segmentierung in Szenenbildern entwickelt und soll besonders effizient in der Speicher- und Rechenzeitnutzung sein. Die Architektur basiert ebenfalls auf einer Encoder-Decoder-Struktur, jedoch ohne Skip-Connections. Anders als beim Unet werden bei Segnet Pooling-Indizes verwendet, um räumliche Informationen zu bewahren.

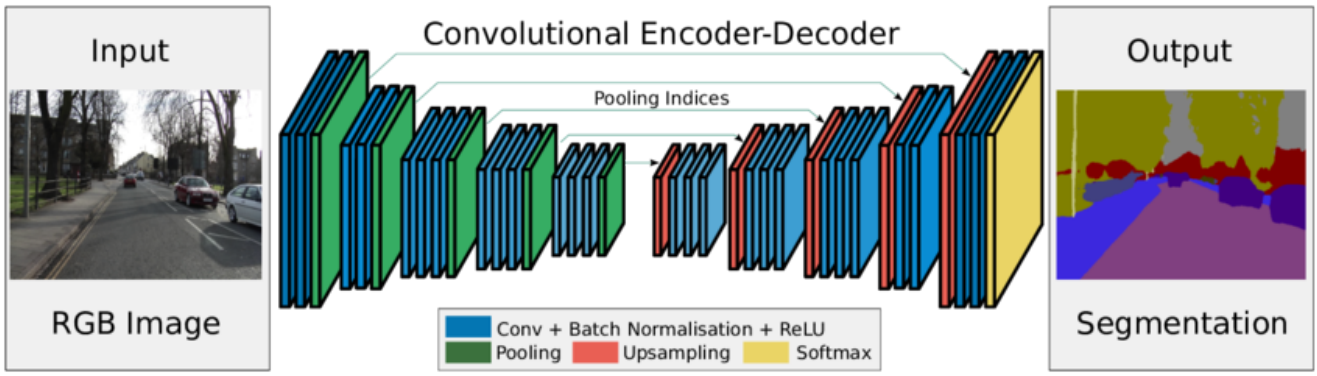

Abbildung 2: Architektur SegNet

Die Architektur besteht dabei aus zwei Hauptpfaden: dem Encoder und dem Decoder. Der Encoder folgt einer typischen Architektur eines Convolutional Neural Networks (CNN), bestehend aus Convolutional Layers, ReLu-Aktivierungsfunktionen und Batch-Normalisierungsschichten. In diesem Vergleich besteht der Encoder-Pfad aus 4 Blöcken, wobei jeder Block ähnlich aufgebaut ist wie bei Unet (zweifache Anwendung von 3x3-Convolution, Batch-Normalisierung und ReLU) und ebenfalls von einem Max-Pooling-Layer abgeschlossen wird, um die Bildgröße zu reduzieren. Im Unterschied zu Unet speichert SegNet jedoch im Encoder-Pfad nicht konkret die Merkmale im Bild ab, sondern merkt sich nur die Indizes (Positionen) der maximalen Werte in jeder Region. Diese werden dann im Decoder verwendet, um die ursprüngliche räumliche Struktur während des Upsamplings präzise wiederherzustellen. Der Decoder-Pfad ist ebenfalls in 4 Blöcke unterteilt und ähnelt dabei ebenso der Struktur des Unet-Decoders. Jeder Block beginnt dabei mit einem Max-Unpooling-Layer, der die Indizes nutzt, um die ursprüngliche Bildgröße wiederherzustellen. Im Gegensatz zu Unet, das für das Upsampling Transposed Convolutions einsetzt, verwendet SegNet diese effizientere Methode, um Speicherplatz und Rechenzeit zu sparen. Nach dem Unpooling folgen die gleichen Schichten wie im Encoder (zweifache Anwendung von 3x3-Convolution, Batch-Normalisierung und ReLU). Demzufolge verzichtet SegNet auf Skip Connections, welches die Speicher- und Rechenressourcen verringert. Insgesamt erlangte das Neuronale Netz dabei eine Parameteranzahl von 7.084.801 Parametern.

2.3 Unterschiede

Bereits im Aufbau der Architektur lassen sich Unterschiede gegenüber der beiden Architekturen verzeichnen. Während Unet im Encoder alle wichtigen Merkmale direkt speichert und per Skip-Connection an den Decoder weitergibt, speichert SegNet lediglich die Pooling-Indizes der wichtigsten Bereiche und nutzt diese später im Decoder. Ebenso werden unterschiedliche Methoden für das Upsampling verwendet. Unet verwendet ConvTranspose2d, was den Filter einer klassischen Convolution in umgekehrter Richtung anwendet, sodass das Bild erweitert wird. Dabei werden zusätzliche Werte zwischen den vorhandenen Pixeln eingefügt, deren Werte während des Trainings erlernt werden. SegNet hingegen nutzt MaxUnpooling2d zur Wiederherstellung der

ursprünglichen räumlichen Informationen. MaxUnpooling wurde speziell für das Umkehren von Max-Pooling entwickelt und nutzt dafür die während des Poolings gespeicherten Indizes, um die Maximalwerte exakt an ihre ursprüngliche Positionen zurückzusetzen. Alle restlichen Positionen mit Nullen gefüllt.

3 Training

Alle in diesem Bericht genannten Modelle wurden für die Analyse eigens trainiert. Dadurch ergaben sich während des Trainings immer wieder kleine Unterschiede, welche je nach Datensatz und Modell variierten. Diese Unterschiede werden im folgenden Abschnitt beschrieben. Um einen fairen Vergleich zu gewährleisten, wurden je Datensatz immer die gleichen Hyperparameter verwendet.

3.1 ISIC 2018 (Task 1)

Der ISIC 2018 ist ein Datensatz, welcher sich speziell mit der Binärsegmentierung von Hautläsionen beschäftigt. Dieser besteht aus einem Trainingsdatensatz mit 2594 Bildern, einem Validierungsdatensatz mit 100 Bildern und einem Testdatensatz zur Evaluierung des Datensatzes mit 1000 Bildern. Für das Training wurden die Bilder auf eine Auflösung von 256 x 256 Pixel reduziert. Aufgrund der Binärsegmentierung wurde hierbei auf den DiceLoss von Monai zurückgegriffen. Als Optimierer wurde Adam aufgrund seiner geringen Anfälligkeit bezüglich Lernraten sowie ein ExponentialLR Scheduler verwendet. Da es sich beim ISIC 2018 um den ersten Datensatz handelte, wurde Ray genutzt, um erste geeignete Hyperparameter (Lernrate und Gamma) zu finden. Das Training erstreckte sich über 50 Epochen.

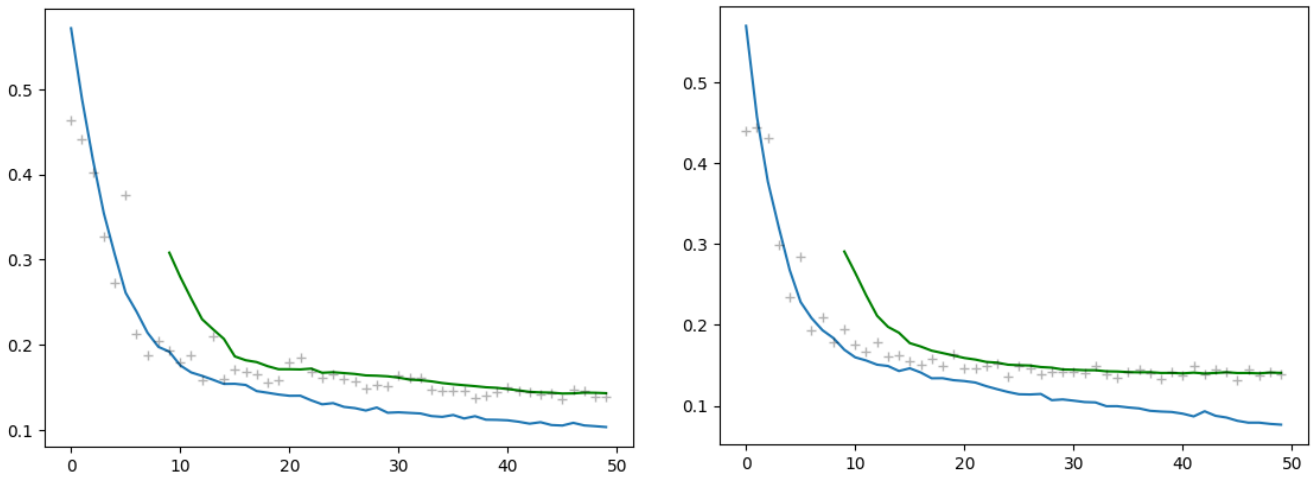

Abbildung 3: Loss für Unet(links) und SegNet(rechts)

Hinsichtlich des Loss-Verlaufs lässt sich bei beiden Modellen eine ähnliche Stagnation feststellen. Der durchschnittliche Trainings- und Validierungs-Loss beider Modelle lässt sich der Tabelle 1 entnehmen.

Loss-typ \ Modell	Unet	SegNet
Train	0.10357	0.07653
Validation	0.13932	0.13851

Tabelle 1: Durchschnittlicher Loss

Die Ergebnisse zeigen, dass sich beide Architekturen gleich gut an den Datensatz anpassen können. Allerdings zeigt SegNet eine stärkere Tendenz zum Overfitting. Während der Trainings-Loss weiter sank, blieb der Validierungs-Loss auf einem konstanten Wert. Auch hinsichtlich der Trainingszeiten pro Epoche ließen sich keine signifikanten Unterschiede ($\pm 2sec$) erkennen.

3.2 ISIC 2018 (Task 1) dezimierter Datensatz

Aufgrund des identischen Trainings und im späteren Verlauf beschriebenen Parallelen bei der Evaluierung wurde der ISIC 2018 Datensatz verkleinert, um das Verhalten bei wenig Trainingsdaten zu analysieren. Der Trainingsprozess blieb dabei unverändert, wie zuvor.

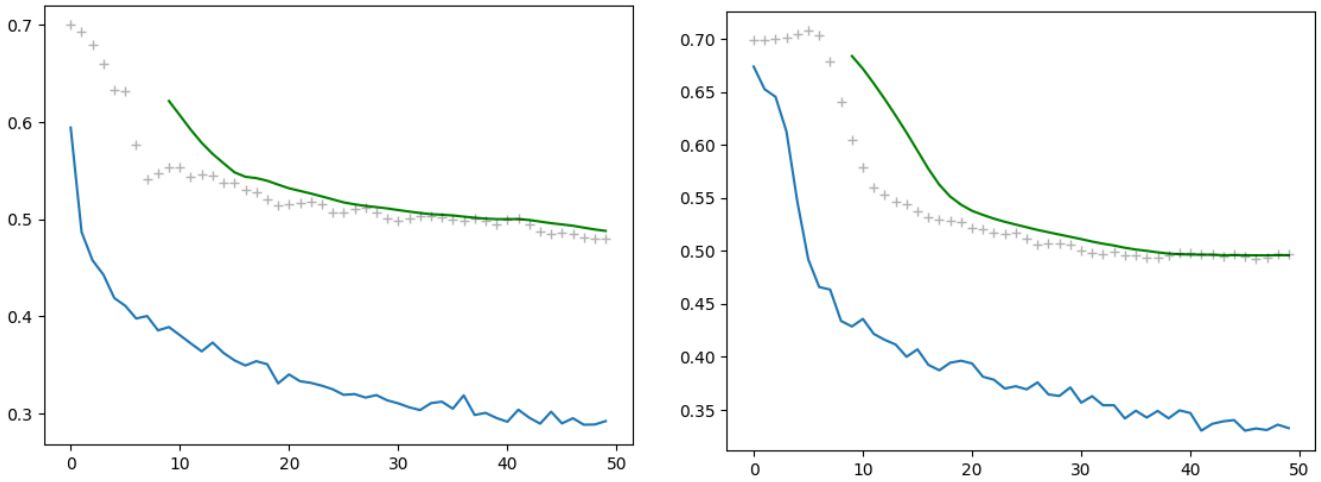

Abbildung 4: Loss für Unet(links) und SegNet(rechts)

Die Reduzierung des Datensatzes auf 100 Bilder wirkte sich zwangsweise negativ auf die Qualität der Modelle aus, was vor allem am höheren Validierungs-Loss erkennbar ist. Bei der Betrachtung der Losskurve lässt sich Folgendes sagen. Bei Unet sinkt der Validierungs-Loss von Beginn an rapide ab und zeigt ab circa Epoche 7 eine deutliche Abflachung. Trotz der Abflachung lässt sich bei längerer Trainingsdauer jedoch ein weiterer Abfall vermuten. Bei SegNet hingegen steigt der Validierungs-Loss zu Beginn kurz an, bevor er ab etwa Epoche 5 rapide abfällt und im späteren Verlauf immer weiter abflacht. Im Gegensatz zu Unet lässt sich hier keine weitere Veränderung mit längerer Trainingsdauer vermuten. Demzufolge lässt sich sagen, dass SegNet tendenziell anfälliger für Überanpassung ist.

Loss-typ \ Modell	Unet	SegNet
Train	0.29214	0.33243
Validation	0.48039	0.49699

Tabelle 2: Durchschnittlicher Loss

Bei den Trainingszeiten pro Epoche lassen sich bei dem reduzierten Datensatz ebenfalls stärkere Unterschiede erkennen. Während Unet circa 15 Sekunden pro Epoche braucht, sind es bei SegNet hingegen nur 10 Sekunden. Dies würde sich mit den Unterschieden in der Architektur decken.

3.3 OxfordIIITPet

Bei dem OxfordIIITPet Datensatz handelt es sich um einen Multi-Klassen-Datensatz zur Segmentierung der Silhouetten diverser Haustiere. Der Datensatz enthält dabei 3 Klassen (Hintergrund, Haustier und Konturen). Standardmäßig stellt der Datensatz nur einen gemeinsamen Training- und Validierungsdatensatz und einen Testdatensatz, jedoch ohne Ground Truth Masken zur Verfügung. Für das Training wurde ein 80/20 Split durchgeführt, was 2944 Trainingsbilder und 736 Validierungsbilder hervorbrachte. Aufgrund der Multiklassen wurde der CrossEntropyLoss verwendet. Als Optimierer wurde wieder Adam benutzt, diesmal jedoch mit einem OneCycleLR Scheduler, damit nicht für jeden Datensatz ein Hyperparametertuning vollzogen werden muss. Das Training erstreckte sich über 100 Epochen.

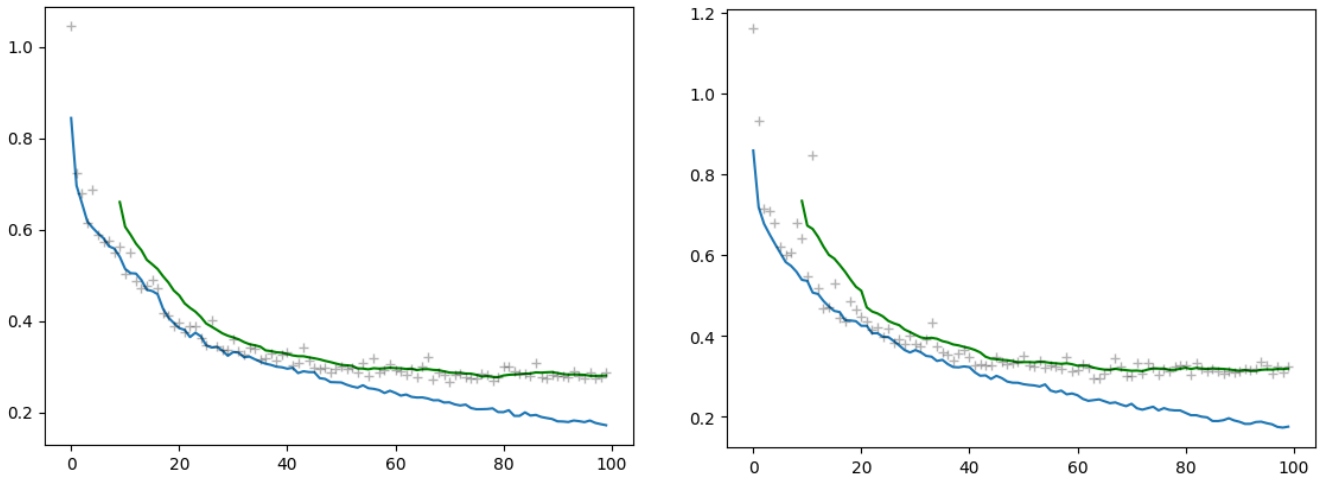

Abbildung 5: Loss für Unet(links) und SegNet(rechts)

Auf den ersten Blick lassen sich beim Training des OxfordIIITPet Datensatzes keine gravierenden Unterschiede feststellen. Anhand des Losses kann bei Segnet eine unstetigere Entwicklung zu Beginn des Trainings vermutet werden. Dies zeigt sich auch minimal am Endergebnis des Validierungs-Loss, welcher der nachfolgenden Tabelle 3 entnommen werden kann.

Loss-typ \ Modell	Unet	SegNet
Train	0.17169	0.17586
Validation	0.28700	0.32407

Tabelle 3: Durchschnittlicher Loss

Auch anhand der Trainingszeiten pro Epoche lässt sich bei Unet ebenfalls nur eine kleine Differenz von $(+2sec)$ gegenüber SegNet feststellen.

3.4 Cityscapes

Abschließend wurde ein Datensatz mit einer hohen Anzahl an Segmentierungsklassen gewählt. Bei dem Datensatz handelt es sich um Cityscapes, ein Multi-Klassen-Datensatz zur Segmentierung von Objekten im Straßenverkehr. Der Datensatz umfasst dabei 35 Klassen aus unterschiedlichen Kategorien wie (Straße, Auto, Fahrrad, Straßenschilder, etc.). Für das Training standen 2975 verschiedene Bilder zur Verfügung. Da der Testdatensatz keine Segmentierungsmasken enthält, wurde der Validierungsdatensatz mit 500 Bildern auch als Testdatensatz verwendet. Für das Training wurden die Bilder auf eine Auflösung von 256 x 512 Pixeln reduziert. Der Trainingsprozess wurde in gleichem Maße wie bei OxfordIIITPet mit CrossEntropyLoss sowie Adam Optimierer und OneCycleLR Scheduler aufgebaut. Das Training erstreckte sich über 50 Epochen.

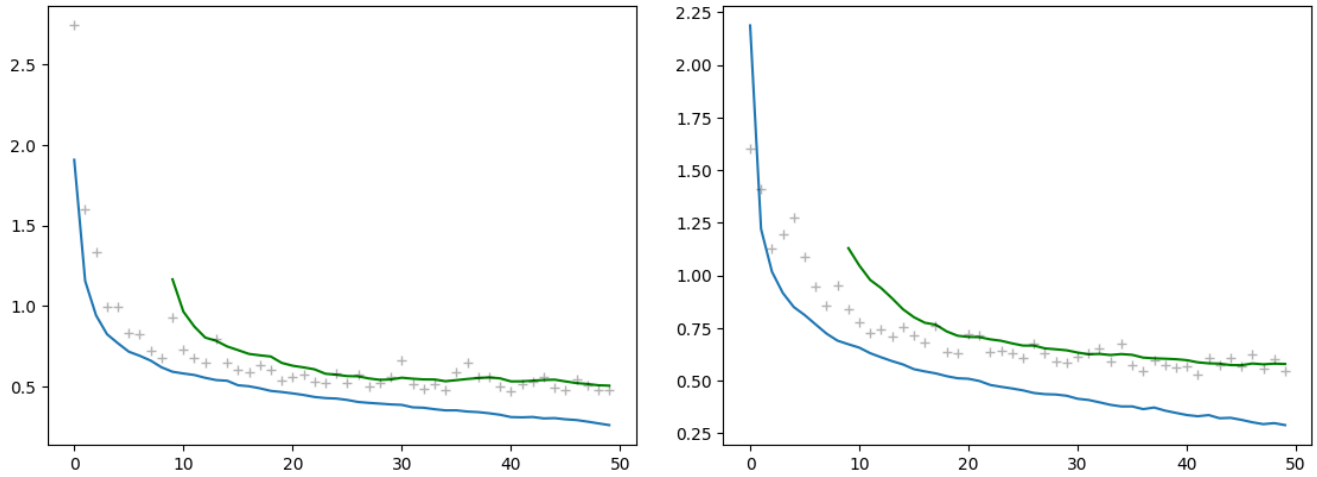

Abbildung 6: Loss für Unet(links) und SegNet(rechts)

n Die Ergebnisse des Trainings überraschten hierbei. Es wurde angenommen, dass SegNet aufgrund seines Ursprungs im Bereich Straßenbilder eine bessere Qualität liefert. Entgegen dieser Erwartung zeigte Unet jedoch während des Trainings eine bessere Performance und eine kürzere Trainingszeit von $(-2sec)$ pro Epoche auf. Dies warf einige Fragen auf, da in der Theorie SegNet schneller trainieren sollte. Der Grund hierfür liegt in der Architektur, da SegNet Max-Unpooling verwendet. Max-Unpooling besitzt im Gegensatz zu ConvTranspose keine trainierbaren Parameter.Dennoch konvergierte Unet ein Stück besser, wie auch die Tabelle 4 zeigt.

Loss-typ \ Modell	Unet	SegNet
Train	0.25839	0.27315
Validation	0.47354	0.54786

Tabelle 4: Durchschnittlicher Loss

3.5 Unterschiede

Anhand der Trainings auf diversen Datensätzen konnten sowohl Gemeinsamkeiten als auch Unterschiede zwischen Unet und SegNet beobachtet werden. So benötigt Unet beispielsweise auf den meisten Datensätzen längere Trainingszeiten pro Epoche. Besonders deutlich wurde dies beim dezimierten ISIC 2018 Datensatz, bei dem ein Unterschied von 5 Sekunden zu vermerken ist. Im Gesamten betrachtet unterscheiden sich die Trainingszeiten jedoch nicht gravierend. Im Bezug auf

die Konvergenz scheint Unet meist zu dominieren. SegNet zeigt vor allem am Anfang des Trainings immer Unstetigkeiten auf. Besonders bei geringen Daten weist Unet eine bessere Anpassbarkeit und geringeres Overfitting auf. Bei großen Datensätzen waren beide Modelle meist gleichauf, wobei SegNet scheinbar mehr zum Overfitting neigt.

4 Evaluierung

Nach der Betrachtung des Trainingsprozesses soll nun näher auf die Leistung der Modelle eingegangen werden. Dazu werden einerseits globale Metriken wie die Pixel Accuracy sowie der IoU (Intersection over Union) herangezogen, jedoch auch konkrete Beispielbilder betrachtet. Dadurch kann auch die Qualität der Segmentierung mit einbezogen werden.

4.1 Globale Metriken

Zur Überprüfung der Leistung eines Modells in der Bildsegmentierung sind globale Metriken von großer Bedeutung. Zwei der wichtigsten Metriken sind dabei die Pixel Accuracy und der Intersection over Union (IoU). In der Bildsegmentierung spielen globale Metriken eine zentrale Rolle, um die Qualität eines Modells quantitativ zu bewerten. Zwei der am häufigsten verwendeten Metriken sind die Pixel Accuracy und die Intersection over Union (IoU). Diese Metriken bieten unterschiedliche Perspektiven auf die Leistung eines Modells und sind für die Analyse von Segmentierungsergebnissen essenziell. Die Pixel Accuracy misst den Anteil der korrekt klassifizierten Pixel im gesamten Bild. Jedoch kann diese Metrik leicht eine verzerrte Sicht auf die eigentliche Modellleistung liefern, da vor allem bei unausgeglichenen Verteilungen mit viel Hintergrund und wenig Objektanteil der Hintergrund dominieren kann und das Ergebnis verfälscht. Der IoU ist diesbezüglich eine robustere Metrik, die speziell für die Bewertung von Segmentierungsmodellen geschaffen wurde. Dabei wird das Verhältnis der Überlappung zwischen Vorhersage und Ground-Truth-Maske errechnet. Berechnet wird der IoU für jede Klasse separat. Aus diesen kann abschließend ein Mittelwert gebildet werden. Die Ergebnisse der Metriken können der Tabelle 5 entnommen werden.

Modell \ Datensatz	ISIC 2018	ISIC 2018 small	OxfordIIITPet	Cityscapes
Pixel Accuracy:				
Unet	91.19%	72.99%	82.99%	44.65%
SegNet	91.31%	72.34%	80.97%	39.38%
Intersection over Union:				
Unet	76.71%	54.04%	70.59%	36.93%
SegNet	76.26%	51.21%	67.38%	32.65%

Tabelle 5: Globale Metriken

Bei der Betrachtung des Iou auf Klassenebene für den Cityscapes Datensatz, zeigen sich auch Unterschiede hinsichtlich der beiden Architekturen. Beispielsweise erzielt Unet bei kleinen Objekten immer bessere Ergebnisse. Besonders gravierend ist dies bei Fahrradfahrern und Verkehrslichtern. Für SegNet ist es kaum möglich, diese zu klassifizieren. Bei größeren Objekten sind beide Modelle meist gleichauf. Jedoch gab es auch Klassen, welche von beiden Modellen nur sehr schlecht klassifiziert werden konnten, siehe Zuggleise.

Klasse \ Modell	Unet	SegNet
rider	20.19%	7.33%
traffic light	26.65%	8.57%
bridge	20.98%	29.86%
road	91.84%	90.55%
bus	32.30%	39.90%
rail track	4.08%	2.29%

Tabelle 6: IoU für Beispielklassen

4.2 Qualitative Bewertung

Um weitere Unterschiede zwischen den Architekturen zu finden, wurde abschließend eine visuelle Auswertung durchgeführt. Die visuelle Darstellung ermöglichte es, mögliche Muster oder Abweichungen hinsichtlich der Segmentierung aufzudecken. Für den Vergleich wurde jeweils das Originalbild, die Ground-Truth-Maske sowie die vorhergesagte Segmentierungsmaske gegenübergestellt. Zur Veranschaulichung wurden prägnante Beispiele zufällig aus den Datensätzen ausgewählt, um typische Stärken und Schwächen der Modelle zu beleuchten.
Alle in diesem Abschnitt genannten Bilder sind dem Anhang zu entnehmen. Insgesamt zeigten sich über alle Datensätze hinweg immer wieder Unterschiede hinsichtlich der Segmentierungsergebnisse. Dabei brachten beide Modelle ihre Stärken und Schwächen zum Vorschein. Meist überzeugte Unet mit präzisen und detaillierten Segmentierungen. Besonders bei kleinen Objekten oder Objekten mit klaren Farbunterschieden konnte dies beobachtet werden. Jedoch gab es auch Situationen, in denen Unet inkorrekte Einschnitte in den vorhergesagten Masken erzeugte. Bei den betrachteten Bildern handelte es sich aber immer um Objekte, welche in sich starke farbliche Unterschiede aufwiesen. SegNet lieferte in diesen Fällen bessere Ergebnisse und erkannte die Objekte zumindest vollständig siehe Abbildung 7, wenn auch nicht optimal.
Kontrastarme Situationen zwischen Objekt und Hintergrund wirkten sich hin und wieder auch schlechter auf Unet aus. Dies lässt vermuten, dass Unet aufgrund der Skip-Connections stärker auf Farbunterschiede angewiesen ist, um Objekt und Hintergrund zu unterscheiden. Zu beobachten ist dieses Verhalten beispielsweise in Abbildung 11. Unet hat in dieser Situation Probleme, den Hund vom Himmel klar abzugrenzen.
Gute Ergebnisse brachten beide Modelle bei der Binärsegmentierung für kleine, klar abgetrennte Objekte auf, wie Abbildung 8 visualisiert. Nach persönlichem Empfinden kam Unet aber in vielen Fällen näher an die Form der Ground-Truth-Maske.
Eine interessante Beobachtung ergab sich unter anderem bei Abbildung 9. Diese enthielt viele Kanten und Streifen im schwarzen Bereich. Für SegNet war es nahezu unmöglich, eine saubere Segmentierung zu vollziehen, was die Annahme brachte, dass SegNet anders als Unet mehr nach Kanten im Bild sucht. Auch die Architektur von SegNet würde diese Aussage stützen, da es auf Max-Unpooling setzt und somit keine Farb- oder Texturinformationen bekommt, sondern nur die Positionen der Maximalausschläge. SegNet orientiert sich demnach mehr an geometrischen Eigenschaften und Kanten. Unet hatte in diesem Bild ebenso Probleme, konnte jedoch zumindest den Bereich des Objektes detektieren.
Der OxfordIIITPet Datensatz brachte auch ein Problem bei Unet hervor, wenn verschiedene, gleichfarbige Objekte relativ nahe beieinander liegen (siehe Abbildung 10). Dies würde abermals den Verdacht bestätigen, dass Unet stärker auf Farbunterschiede reagiert, da SegNet keine Schwierigkeiten hatte, das Objekt zu detektieren.

Gleichwertige Ergebnisse liefern beide Architekturen bei den Straßenbildern des Cityscapes Datensatzes, wobei ebenso kleinere Unterschiede zu erkennen sind. Unet zeigt wieder seine Überlegenheit bei kleinen Objekten, wie in Abbildung 15 zu sehen ist. Das Verkehrsschild wird von SegNet vollständig übersehen, während Unet es korrekt erkennt, jedoch nicht optimal segmentiert. Die Fähigkeit, feine Details zu erkennen, ist bei Straßenbildern jedoch nicht immer von Vorteil (siehe Abbildung 14). Unet klassifiziert irrtümlicherweise in diesem Beispiel ein Objekt (blau), bei dem es sich lediglich um eine farbliche Änderung der Gebäude handelt. SegNet hingegen liefert hier eine einheitliche Maskierung.
Eine abschließende Auffälligkeit ergab sich noch bei SegNet. In Abbildung 12, aber auch bei Bildern des Cityscapes-Datensatzes zeigte sich eine Anfälligkeit für Schatten bzw. stark variierenden Lichtverhältnissen. Dadurch entstehen unsaubere Segmentierungen wie zum Beispiel bei geraden Straßenkanten.

5 Fazit

Insgesamt konnten bei dieser Studienarbeit tiefgreifende Erkenntnisse über die zwei Architekturen Unet und SegNet im Bezug auf Segmentierungsaufgaben gewonnen werden. Auch wenn sich während der Implementierung immer wieder Hürden ergaben, bereitete es dennoch Freude, an dem Projekt zu arbeiten und die Unterschiede zwischen den Architekturen herauszuarbeiten. Um zum Schluss nochmal einen groben Überblick zu bekommen, welche Unterschiede zwischen den Modellen bestehen und welche Limitationen sie besitzen, werden die wichtigsten Punkte nochmals in einer Tabelle gegenübergestellt.

Aspekt	Unet	SegNet
Architektur	Skip-Connections zur Übergabe von Merkmalen aus dem Encoder	Pooling Indizes zur Rekonstruktion
Upsampling	ConvTranspose	MaxUnpooling
Trainingseffizienz	Längere Trainingszeiten pro Epoche, jedoch bessere Generalisierung	Kürzere Trainingszeiten pro Epoche, da MaxUnpooling keine traininerbaren Parameter hat
Kontrast	Schwierigkeiten bei kontrastarmen Szenen (starke Abhängigkeit von Farbunterschieden)	Robuster bei kontrastarmen Szenen; stärker auf geometrische Merkmale und Kanten fokussiert
Schwächen	Probleme bei ähnlichen Farben innerhalb eines Objekts oder zwischen Objekten	Probleme bei Schatten und variierenden Lichtverhältnissen; unsaubere Segmentierungen
Segmentierungszeit	ungeeignet für Echtzeitanwendungen	ungeeignet für Echtzeitanwendungen
Datensatzgröße	Lernt tendenziell schneller	Training schwankt zu Beginn meist stark

Tabelle 7: Unterschiede/Limitierungen Unet und SegNet

A Anhang

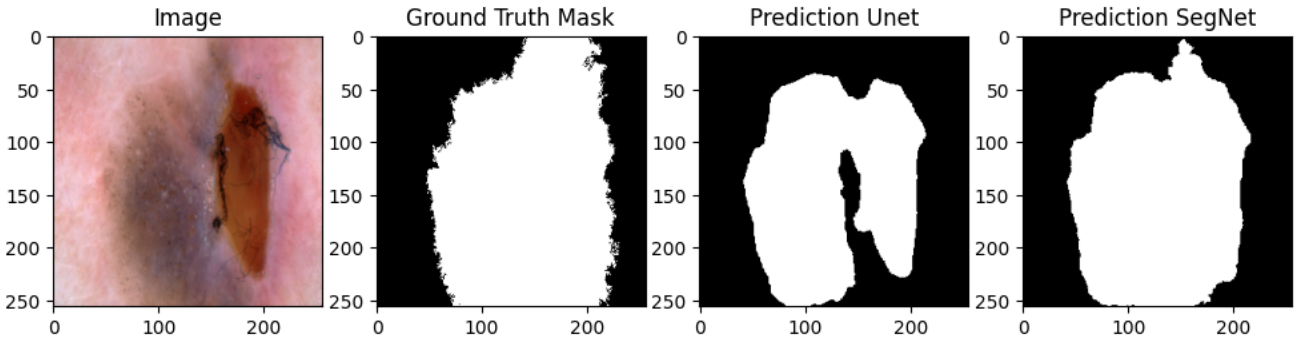

Abbildung 7: Segmentierung Isic bsp. 114

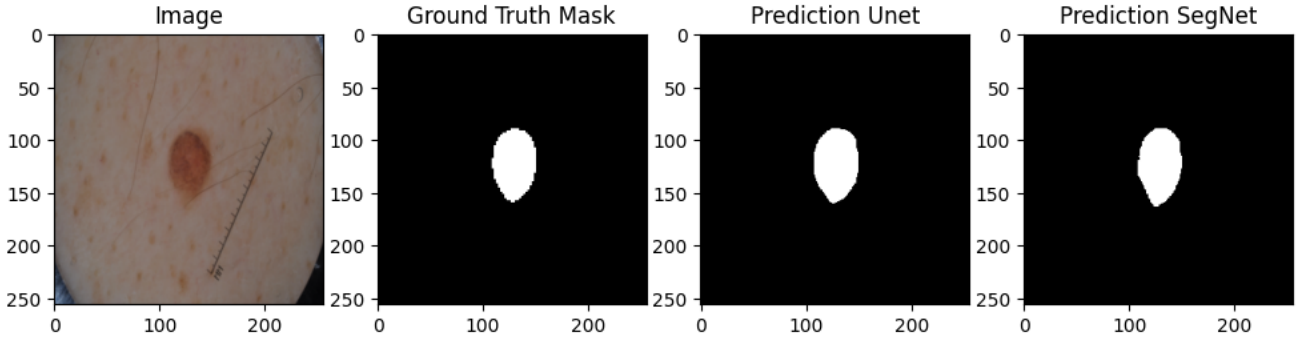

Abbildung 8: Segmentierung Isic bsp. 10

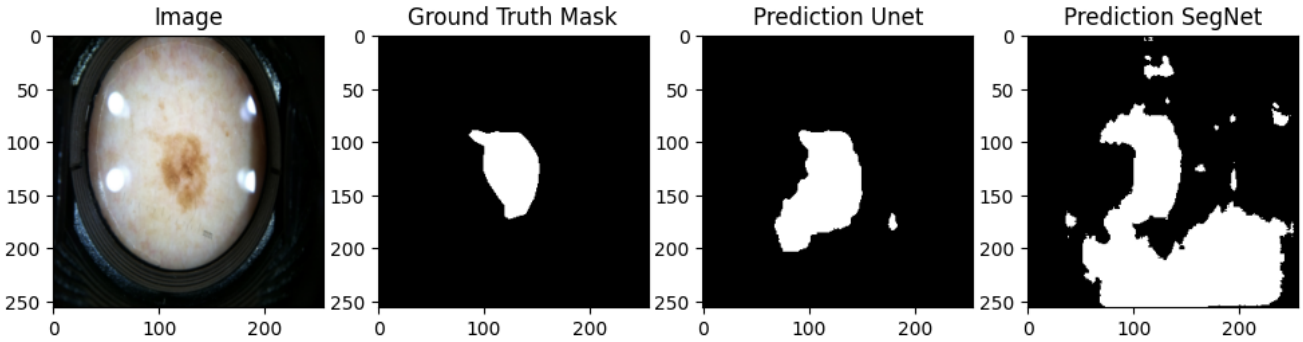

Abbildung 9: Segmentierung Isic bsp. 601

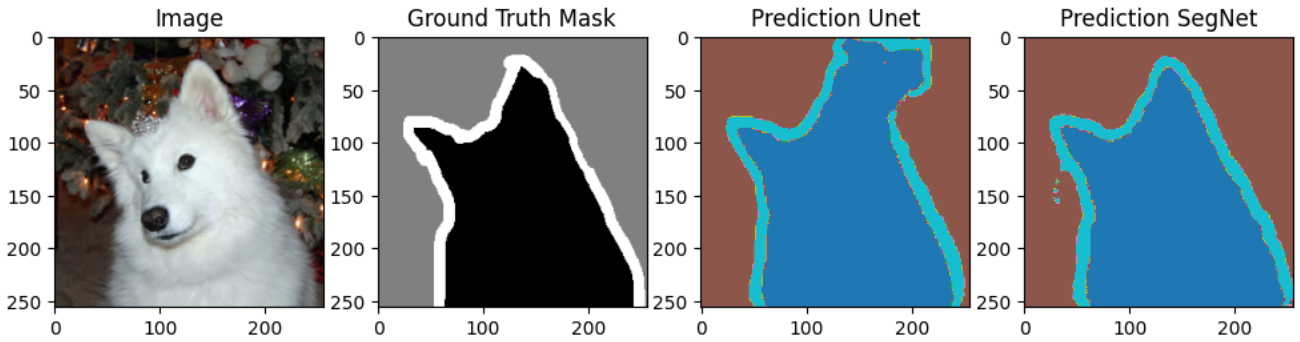

Abbildung 10: Segmentierung OxfordIIITPet bsp. 243

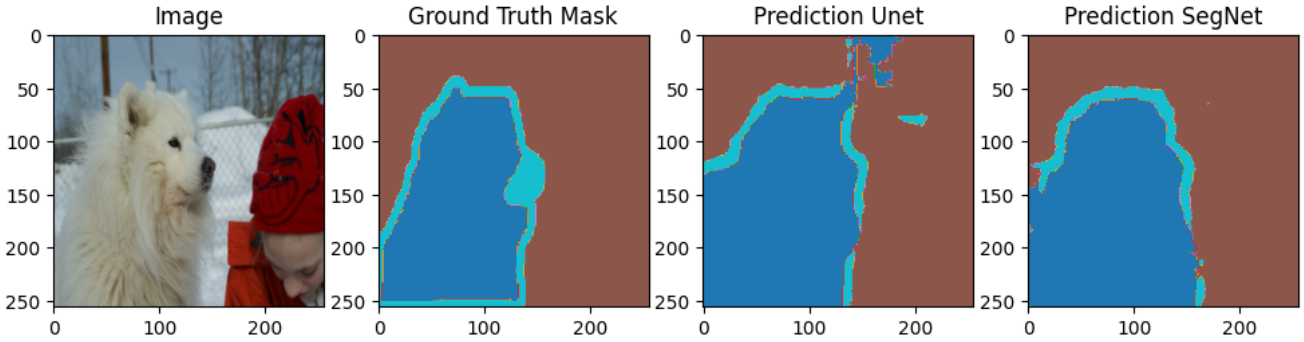

Abbildung 11: Segmentierung OxfordIIITPet bsp. 89

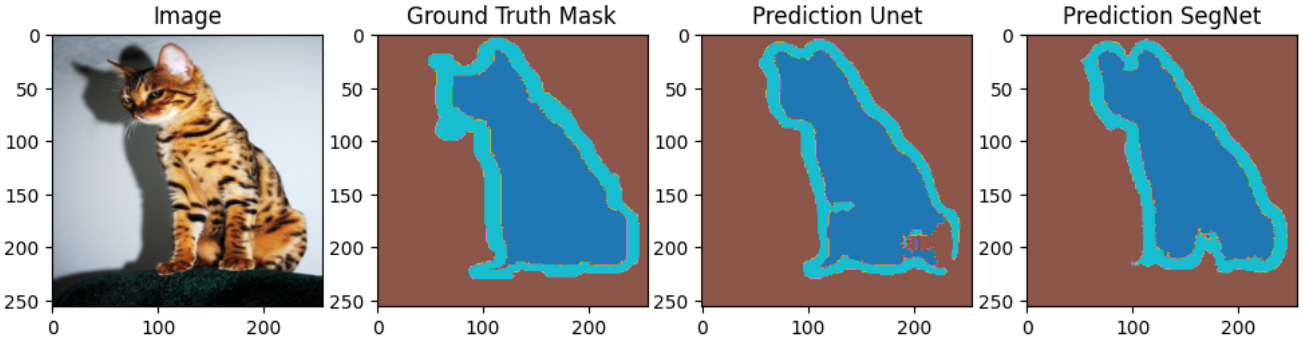

Abbildung 12: Segmentierung OxfordIIITPet bsp. 146

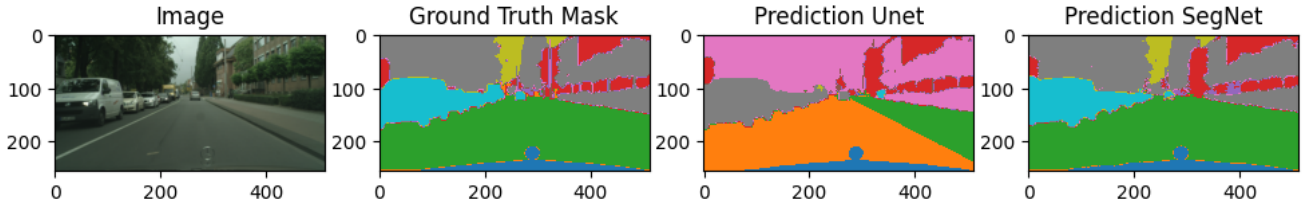

Abbildung 13: Segmentierung Cityscapes bsp. 5

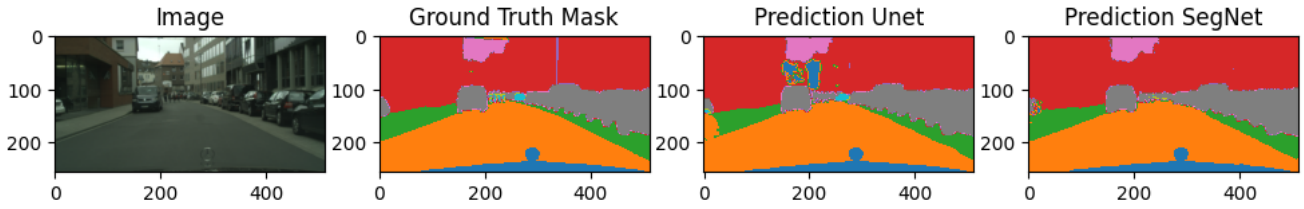

Abbildung 14: Segmentierung Cityscapes bsp. 55

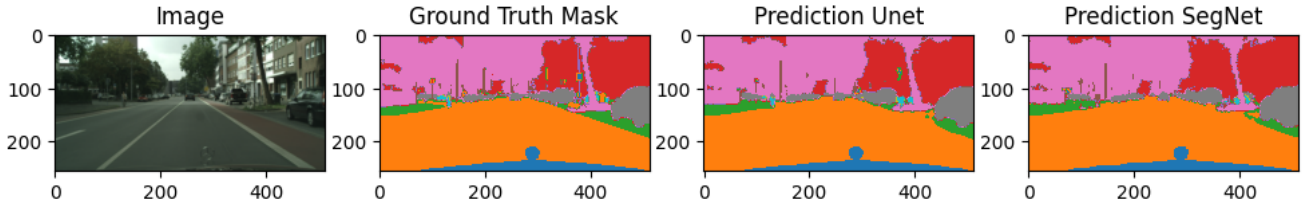

Abbildung 15: Segmentierung Cityscapes bsp. 100

Literatur

Cityscapes (2025). Cityscapes website, https://www.cityscapes-dataset.com. Accessed: Jan-2025.

Community (2024). How to calculate iou, https://discuss.pytorch.org/t/correct-way-to-calculate-iou-for-each-class/208470/5. Accessed: Jan-2025.

Community (2025a). Differences maxunpoos convtranspose, https://discuss.pytorch.org/t/difference-between-maxunpool3d-and-convtranspose3d/22024. Accessed: Jan-2025.

Community (2025b). Identical transforms, https://discuss.pytorch.org/t/torchvision-transfors-how-to-perform-identical-transform-on-both-image-and-target/10606/6. Accessed: Jan-2025.

dhruvbird (2025). pets_segnet, https://github.com/dhruvbird/ml-notebooks/blob/main/pets_segmentation/oxford-iiit-pets-segmentation-using-pytorch-segnet-and-depth-wise-separable-convs.ipynb. Accessed: Jan-2025.

Huggingface (2025). Datasets from huggingface, https://huggingface.co/datasets. Accessed: Jan-2025.

MONAI (2025). Metrics monai, https://docs.monai.io/en/stable/metrics.html. Accessed: Jan-2025.

OpenAI (2025). Chatgpt, https://chatgpt.com/. Accessed: Jan-2025.

Pytorch (2025a). Cityscapes, https://pytorch.org/vision/main/_modules/torchvision/datasets/cityscapes.html. Accessed: Jan-2025.

Pytorch (2025b). Cross entropy loss, https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html. Accessed: Jan-2025.

Pytorch (2025c). Onecyclelr scheduler, https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.OneCycleLR.html. Accessed: Jan-2025.

Pytorch (2025d). Oxfordiiitpet dataset, https://pytorch.org/vision/0.20/generated/torchvision.datasets.OxfordIIITPet.html. Accessed: Jan-2025.

Pytorch (2025e). Random flip, https://pytorch.org/vision/main/generated/torchvision.transforms.RandomHorizontalFlip.html. Accessed: Jan-2025.

Pytorch (2025f). Sgd, https://pytorch.org/docs/stable/generated/torch.optim.SGD.html. Accessed: Jan-2025.

PyTorch (2025g). Torch convtranspose, https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html. Accessed: Jan-2025.

Rashid, S. N. (2021). Unet segmentation, https://medium.com/red-buffer/semantic-segmentation-u-net-1e5c0f4516a5. Accessed: Jan-2025.

TorchMetrics (2025). Jaccardindex, https://lightning.ai/docs/torchmetrics/stable/classification/jaccard_index.html#multiclassjaccardindex. Accessed: Jan-2025.

TRU$_{P}$*rojects*(2025).*Unetsegnetsegmentation*, . Accessed: Jan-2025.