ALS Klassifizierung mit Submanifold Sparse Convolutional Networks

STEFAN SCHMOHL[1] & UWE SÖRGEL[1]

Zusammenfassung: Die Klassifizierung von Punktwolken aus Airborne Laser Scanning (ALS) ist eine der Hauptkomponenten in deren Verarbeitung. Etablierte Methoden beruhen auf der aufwendigen Berechnung von punktweisen Merkmalen. Convolutional Neural Networks (CNNs) haben sich als mächtige Klassifikatoren etabliert, die zugleich auch die optimalen Merkmale selbstständig erlernen. Ihre Anwendung auf ALS-Daten ist jedoch nicht trivial und basiert bisher meist auf der Projektion der Punktwolke in zweidimensionale Abbildungen. Reine 3D-CNNs benötigen sehr viel Speicher und Rechenzeit. Wir verwenden daher Sparse Submanifold Convolutional Networks (SSCNs) für eine effiziente semantische Segmentierung von Voxelwolken in einer Ende-zu-Ende Encoder-Decoder-Architektur. Wir demonstrieren das Verfahren auf dem ISPRS Vaihingen 3D Semantic Labeling Benchmark und erreichen bis zu 85,0% Gesamtgenauigkeit bei zugleich geringen Inferenzzeiten von nur wenigen Sekunden.

1 Einleitung

Flugzeuglaserscanning (ALS) liefert Massendaten in Form von 3D-Punktwolken. Um hieraus semantische Information über Objekte zu gewinnen, wird als Zwischenschritt oftmals jedem 3D-Punkt eine Klasse aus einem gegebenen Katalog von Objektkategorien zugewiesen. Eine solche Klassifikation kann jedoch nicht isoliert für Einzelpunkte erfolgen. Vielmehr ist die Einbeziehung des räumlichen Kontextes erforderlich, der sich aus der Verteilung der Punkte in einer lokalen Nachbarschaft ergibt. Üblicherweise werden dazu aus der Umgebung jedes Punktes geometrische Merkmale abgeleitet. Im klassischen Ansatz erfolgt die Definition der betrachteten Nachbarschaften a priori durch den Menschen. Die Klassifikation der Punkte im Merkmalsraum erfolgt anschließend mit Standardverfahren wie Random Forests.

Convolutional Neural Networks (CNNs) haben sich in den letzten Jahren als State of the Art in der Bildanalyse etabliert. Um auch 3D-Daten mit diesen in erster Linie für 2D-Bilder entwickelten Methoden zu verarbeiten, erfolgt häufig eine Abbildung der 3D-Daten in eine Menge von 2D-Projektionen. Dies ist jedoch mit einem Verlust an Informationen verbunden und lässt sich nicht auf Daten anwenden, deren Dreidimensionalität erhalten bleiben soll.

Da Faltungsoperationen auf Rasterdaten mathematisch keiner Beschränkung bezüglich der Dimension des Raumes unterliegen, können CNNs theoretisch Rasterdaten mit beliebig vielen Dimensionen und beliebiger Größe verarbeiten. In der Praxis begrenzt jedoch der hohe Speicher- und Rechenbedarf die Datenmenge und damit die Auflösung von 3D-Inputs.

3D-Daten sind oftmals durch eine stark inhomogene räumliche Verteilungsdichte gekennzeichnet. Große Teile des (Voxel-) Raumes sind in der Regel nicht belegt. Um dies auszunutzen, adaptieren

[1] Universität Stuttgart, Institut für Photogrammetrie, Geschwister-Scholl-Str. 24D, D-70174 Stuttgart, E-Mail: [stefan.schmohl, uwe.soergel]@ifp.uni-stuttgart.de

wir in dieser Arbeit Submanifold Sparse Convolutional Networks (SSCN) zur semantischen Segmentierung von ALS-Punktwolken.

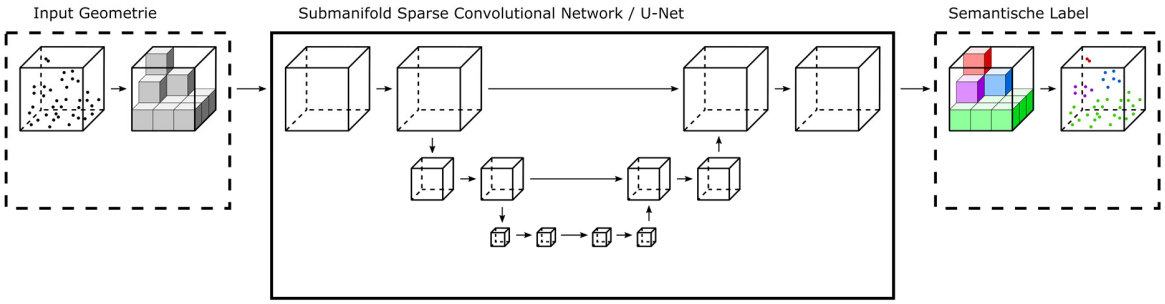


Abb. 1: Schema der Prozesskette. Ein Punktwolken-Sample wird einem Voxelfilter unterzogen und anschließend durch ein SSCN in der Form eines U-Nets semantisch segmentiert. Anschließend werden die Voxellabel zurück auf die Punktwolke übertragen. Angedeutet ist die räumliche Auflösung des Samples innerhalb des Netzes: umso tiefer die Stufe, desto geringer die Auflösung.

2 Forschungsstand

2.1 Überwachte Klassifizierung von ALS Daten

Das übliche Vorgehen bei der semantischen Segmentierung einer Punktwolke, oft auch Punktwolkenklassifizierung genannt, besteht aus einem zweistufigen Verfahren. Als Erstes werden auf Basis von Expertenwissen entworfene, punktweise Merkmale berechnet. Neben echobasierten Merkmalen und normalisierten Höhen können eine Reihe von nachbarschafts-bezogenen Merkmalen verwendet werden, beispielsweise berechnet aus den Eigenwerten des Strukturtensors.
Im zweiten Schritt folgt die Klassifizierung der Punkte anhand dieser Merkmale. Typische Klassifikatoren sind Support Vector Machines (SVM) oder Random Forests (RF), z.B. in CHEHATA et al. (2009), BLOMLEY & WEINMANN (2017) und HACKEL et al. (2016). Solche Verfahren behandeln jeden Punkt einzeln, ohne Berücksichtigung semantischer Interaktionen zwischen den Klassen benachbarter Punkte, was zu feinkörnig verrauschten Resultaten führen kann. Zur Einbeziehung von räumlichem Kontext klassifizieren NIEMEYER et al. (2014; 2016) alle Punkte simultan in einem Conditional Random Field (CRF).
Die Berechnung der für diesen Ansatz notwendigen Merkmale beruht zu einem Großteil auf zeitaufwendigen Nachbarschaftsanfragen. Zudem muss für jede Anwendung manuell ein möglichst optimaler Satz dieser Merkmale gefunden werden.

2.2 Convolutional Neural Networks zur ALS Klassifizierung

Convolutional Neural Networks (CNNs) sind State of the Art in vielen Disziplinen wie der Computer Vision, insbesondere in der Bild-Klassifizierung. Sie lernen neben der Merkmalsklassifikation zusätzlich auch deren Extraktion in einer Ende-zu-Ende-Manier. Gewöhnliche CNNs benötigen rasterisierte, zweidimensionale Eingangsdaten. 3D Punktwolken sind jedoch unstrukturiert und weisen stark inhomogene Punktdichten auf. Die Anwendung auf ALS-Daten ist aus diesen Gründen nicht trivial.

Die meisten vergleichbaren Arbeiten konzentrieren sich darauf, ALS-Punktwolken möglichst sinnvoll für die Verarbeitung mit CNNs in 2D bzw. 2,5D Raster umzuwandeln. So klassifizieren beispielsweise HU & YUAN (2016) ALS-Punkte, indem sie jeden Punkt durch eine vertikale Projektion ihrer Umgebung beschreiben. Jedes Pixel erhält dabei drei Werte: Z_{min}, Z_{mittel} und Z_{max}. Die vom neuronalen Netzwerk prädizierte Objektkategorie für ein solches Bild wird dann übertragen auf den ursprünglichen, in der Mitte liegenden 3D Punkt. Nachteilig bei diesem Ansatz sind die vielen redundanten Berechnungen, da für nahe beieinanderliegende Punkte mehrfach dieselben Merkmale berechnet und innerhalb des Netzes verarbeitet werden müssen. Zusätzlich ist das Ergebnis anfällig für verrauschte Ergebnisse, da die Punkte einzeln, ohne Berücksichtigung der semantischen Beziehungen benachbarter Punkte, klassifiziert werden.
Im Gegensatz dazu erlauben Encoder-Decoder-Architekturen die simultane Zuordnung aller Elemente (Pixel) der Eingabe (POLITZ & SESTER 2018; RIZALDY et al. 2018). Szenen können damit am Stück prozessiert werden. Das Problem des Informationsverlustes durch die Projektion in 2,5D bleibt jedoch bestehen, insbesondere bei Verdeckungen, Fassaden und Multi-Echo-Signalen. Zudem stellt die Berechnung der Abbildungen sowie die Rückprojektion einen zusätzlichen Aufwand dar.
Grundsätzlich können die Operationen eines CNN über beliebig viele Dimensionen definiert werden. Ebenso ist die Rasterisierung von Punktwolken auch im dreidimensional Raum möglich. Die daraus resultierenden Voxelgitter erfordern allerdings sehr viel Speicher und Rechenzeit bei der Verarbeitung in einem 3D-CNN. Dies ist besonders deshalb unverhältnismäßig aufwendig, da der Großteil des Raumes leere Voxel beinhaltet, also sehr *sparse* ist.
Um die geringe Dichte von 3D Daten zu nutzen, wurden verschiedene Ansätze entwickelt, um CNNs auch auf anderen 3D Datenstrukturen als Voxelgittern anzuwenden. GRAHAM et al. (2018) nutzen in ihren *Submanifold Sparse Convolutional Networks* (SSCNs) eine Implementierung der Faltungsschichten in Form von Matrix-Multiplikationen, um ausschließlich „belegte“ Voxel zu betrachten. Dieses Verfahren erzielte die besten Resultate in der Segmentierung von Objektteilen (YI et al., 2017).
Die in der Computer Vision entwickelten 3D-CNNs wurden bisher meist nur für kleine, synthetische Datensätze oder räumlich stark begrenzte, terrestrische Aufnahmen und Innenraum-Scans verwendet. Die Anwendung auf großräumige, mittels ALS von realen Objekten aufgenommene, topographische Punktwolken ist nach unserem Wissen bisher nicht untersucht worden. In dieser Arbeit zeigen wir die Eignung von SSCNs für die Klassifizierung von ALS Punktwolken.

3 Methodik

3.1 Submanifold Sparse Convolutional Network

Wesentlicher und namensgebender Bestandteil von Convolutional Neural Networks sind die Faltungs-Schichten. In ihnen werden mehrere Faltungskerne mit gelernten Gewichten auf die Ergebnisse der jeweils vorherigen Schicht (*activation maps*) angewandt. Im 2D-Fall sind diese activation maps und Faltungskerne dreidimensional, wobei die dritte Dimension die Anzahl an Input-Kanälen bzw. an Filtern der vorherigen Schicht repräsentiert. Die Faltung hat die Form

$$Y_f^l = X^l * W_f^l \tag{1}$$

Dabei bezeichnet W_f^l einen 3D-Faltungskern f der aktuellen Schicht l und $X^l = h(Y^{l-1})$ das Ergebnis der vorherigen Schicht nach der Aktivierungsfunktion $h(\cdot)$.
Um diese Faltungen effizient auf GPUs zu berechnen, kann die Faltungsoperation als Matrixmultiplikation umgeschrieben werden (CHELLAPILLA et al. 2006; CHETLUR et al. 2014):

$$\boldsymbol{Y}^l = \boldsymbol{X}^l \cdot \boldsymbol{W}^l \tag{2}$$

Die Matrix $\boldsymbol{W}^l \in \mathbb{R}^{k^2 c \times |f|}$ enthält alle $|f|$ Faltungskerne der Schicht, jeder mit der Größe $k \times k \times c$. Im Input $\boldsymbol{X}^l \in \mathbb{R}^{|n| \times k^2 c}$ bzw. Output $\boldsymbol{Y}^l \in \mathbb{R}^{|n| \times |f|}$ steht $|n|$ für die Anzahl an Positionen des Faltungskerns. Bei Bildern entspricht dies, ohne Stride und mit entsprechendem Padding, der Bild-Breite multipliziert mit der Bild-Höhe. Das Grundprinzip der Submanifold Sparse Convolution (SSC) ist es, nur diejenigen Zeilen n zu behalten, deren korrespondierende Stellen im ursprünglichen Input nicht leer sind. Dafür genügt es, die nicht leeren Stellen in Listenform zu speichern (*Voxelwolke*), ein voll aufgespanntes Voxelgitter wird nicht benötigt. Für weitere Details siehe GRAHAM (2015) und GRAHAM et al. (2018).

3.2 Netzwerk-Architektur

Wir adaptieren die U-Net Architektur (RONNEBERGER et al. 2015) aus GRAHAM et al. (2018) zur semantischen Segmentierung von voxelierten ALS Punktwolken (Abb. 1). Die Encoder-Decoder-Architektur erlaubt die Ende-zu-Ende Verarbeitung einer Voxelwolke. Pro Stufe des Encoders wird die Auflösung durch Unterabtastung halbiert. Das Netz setzt sich aus genug Schichten zusammen, um die räumliche Ausdehnung in der tiefsten Schicht auf ein Voxel zu reduzieren. Der Decoder ist symmetrisch zum Encoder und verwendet „Deconvolution"-Schichten, um die Auflösung stückweise wieder zu erhöhen. Diese Werte werden jeweils mit denen aus der korrespondieren Encoder-Stufe verbunden. Am Ende des Netzes besitzt jedes Voxel prädizierte Klassen-wahrscheinlichkeiten und wird der Klasse mit der höchsten Wahrscheinlichkeit zugewiesen. Die Label jedes Voxels werden schließlich, außerhalb des Netzes, auf die darin liegenden Punkte übertragen.

3.3 Zielfunktion

Ein Problem bei der semantischen Segmentierung mit CNNs sind Datensätze mit stark inhomogenen Klassenverteilungen. Neuronale Netze neigen generell dazu, beim Training häufig auftretende Klassen bei der Inferenz zu bevorzugen. Im Gegensatz zur üblichen Klassifizierung ist Unter- oder Übersampling hier aber nicht praktikabel, da die Klasseninstanzen nicht einzeln auftreten, sondern nur als Teil der Stichproben, z.B. als Pixel in einem Bild oder Voxel in einem (dünn besetzten) 3D-Gitter.
Alternativ zum Sampling kann auch die Zielfunktion angepasst werden. Wir verwenden in dieser Arbeit einen gewichteten, elementweisen Cross-Entropy-Loss, der mittels Stochastic Gradient Descent optimiert wird (LONG et al. 2015; RONNEBERGER et al. 2015; EIGEN & FERGUS 2015):

$$E = -\frac{1}{\sum_{n=1}^{N} \sum_{\boldsymbol{x} \in \Omega^{(n)}} w(\boldsymbol{x})} \sum_{n=1}^{N} \sum_{\boldsymbol{x} \in \Omega^{(n)}} \sum_{c=1}^{C} y_c(\boldsymbol{x}) \log\left(\hat{y}_c(\boldsymbol{x})\right)\ w(\boldsymbol{x}) \tag{3}$$

wobei N die Anzahl an Samples n im Mini-Batch bezeichnet, $\boldsymbol{x} \in \Omega^{(n)}$ die Voxelpositionen pro Sample, $\hat{y}_c(\boldsymbol{x})$ die prädizierte Wahrscheinlichkeit der Zugehörigkeit von $\boldsymbol{x}$ zur Klasse c, y_c das

gegebene, binär kodierte, Klassenlabel und C die Klassenanzahl. Es hat sich gezeigt, dass die Wurzel der inversen Klassenhäufigkeit hier bei diesem Datensatz ein guter Kompromiss zwischen Recall und Precision der seltenen Klassen ist. Wir verwenden daher

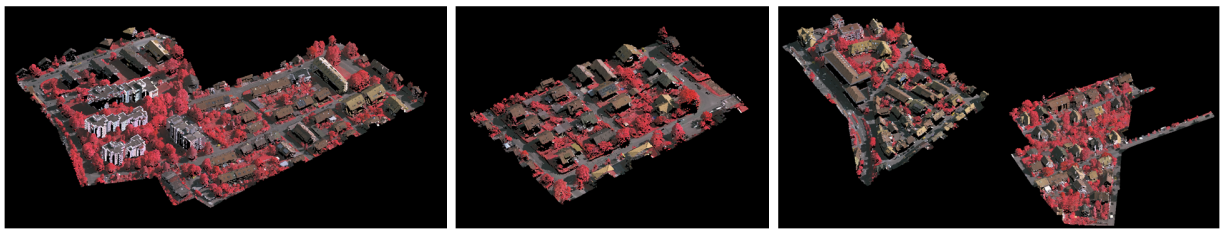
$$w(\boldsymbol{x}) = w\big(y(\boldsymbol{x})\big) = w_c = \frac{1}{\sqrt{f_c}} \tag{4}$$

mit f_c als relative Häufigkeit des wahren Labels $y(\boldsymbol{x})$ bzw. der Klasse c im Trainingsdatensatz. Dieses *class balancing* führt dazu, dass seltene Klassen einen stärkeren Einfluss auf den Gradienten und damit den Lernfortschritt haben als häufigere Klassen. Bei feineren Voxelgrößen, die zu größerem Ungleichgewicht führen, sollte auf die Wurzel verzichtet werden.

4 Experimente

4.1 Datengrundlage

Wir demonstrieren die Eignung des Verfahrens auf dem Datensatz des ISPRS 3D Semantic Labeling Contest[2] (NIEMEYER et al. 2014). Er besteht aus zwei ALS Punktwolken, je eine für Training und Test, aufgenommen über Vaihingen an der Enz, Deutschland. Jedes Echo eines LiDAR-Sendepulses wurde darin als eigener Punkt aufgezeichnet mit den Merkmalen Intensität, Echonummer und -Anzahl. Zusätzlich liegt für jeden Punkt ein Label aus den Klassen *Powerline*, *Low vegetation*, *Impervious surfaces*, *Car*, *Fence/Hedge*, *Roof*, *Facade*, *Shrub* und *Tree* vor. Die nominelle Punktdichte pro Streifen beträgt 4 pts/m². Durch die 30 % Streifenüberlappung beträgt die effektive Punktdichte global etwa 8 pts/m². Zum Zeitpunkt dieser Arbeit ist der Contest bereits geschlossen, daher konnten wir unsere Ergebnisse nicht zur Evaluierung einsenden. Da nun aber auch die Label der Test-Punktwolke verfügbar sind, war trotzdem eine Auswertung zum Vergleich mit anderen Verfahren möglich.
Zusätzlich zu der Punktwolke steht ein True Orthophoto (TOP) desselben Gebiets aus dem entsprechenden 2D Contest zur Verfügung (CRAMER 2010). Dieses hat eine GSD von 9 cm und enthält die spektralen Kanäle Rot, Grün und nahes Infrarot (CIR). Bei einem Teil der Experimente wurde die Punktwolke zuvor anhand dieses TOP eingefärbt (Abb. 2).



(a) Trainings-Split (b) Validierungs-Split (c) Test-Split

Abb. 2: ISPRS Vaihingen 3D Semantic Labeling Datensatz. Angezeigt werden die anhand des TOP kolorierten Punktwolken.

[2] http://www2.isprs.org/commissions/comm3/wg4/3d-semantic-labeling.html

Zur laufenden Überwachung des Lernfortschritts separierten wir die Trainings-Punktwolke manuell in einen festen Trainings- bzw. Validierungs-Split (Abb. 2). Der Training-Split enthält 659.428 Punkte, der Validation-Split 94.448 und zum Testen stehen 411.722 Punkte zur Verfügung.

4.2 Vorverarbeitung

Im Gegensatz zu klassischen Verfahren ist hier keine separate Berechnung von Merkmalen erforderlich. Der einzige notwendige Vorverarbeitungsschritt ist das Voxelieren der Punktwolke. Dies dient zusätzlich der Homogenisierung der Punktdichte. Statt eines voll aufgespannten Voxelgitters bestimmen wir eine Liste nicht-leerer Voxel. Jedem Element dieser Voxelwolke werden die aus den innenliegenden Punkten gemittelten Echomerkmale zugewiesen. Die Klassen-Label werden durch Mehrheitsvotum bestimmt. Als Nebenprodukt des Voxelfilters fällt eine Indexliste an, mit der die prognostizierten Klassenwahrscheinlichkeiten von den Voxeln auf die ursprünglichen Punkte übertragen werden können. Wir untersuchen Voxelgrößen von 2 m, 1 m, 0,5 m, 0,25 m und 0,125 m.

Abb. 3: Voxeliertes Trainingsample mit Kantenlänge 32 × 32 × 64 m und Voxelgröße 1 m. Die Klassen sind farbcodiert gemäß in Abb. 5.

Zur künstlichen Erweiterung der Trainingsdaten werden Trainings- und Validierung-Split vor der Voxelierung zwölfmal um je 30° um die Z-Achse rotiert. Um dem Netzwerk viele Lernbeispiele zeigen zu können und Überanpassung zu vermeiden, teilen wir die Punkt- bzw. Voxel-Wolken in kleinere Samples entlang eines horizontalen Gitternetzes auf. Die Samples müssen jedoch groß genug sein, um einen Aussagekräftigen räumlichen Kontext zu beherbergen. Wir experimentieren mit Samples von 16 × 16 × 64 m, 32 × 32 × 64 m und 64 × 64 × 64 m Ausdehnung. Jedes Sample deckt so die volle vertikale Ausdehnung des Datensatzes ab. Der Überlapp der Training-Samples beträgt 30 %.

4.3 Training

Die Mini-Batch-Größe beim Training beträgt 128 für 16 × 16 × 64 m große Samples. Da bei größerer Ausdehnung und gleicher Überlappung weniger Samples extrahiert werden können, die Anzahl an Parameterupdates pro Epoche jedoch vergleichbar bleiben soll, besteht ein Mini-Batch bei 32 m bzw. 64 m Kantenlänge aus 32 bzw. 8 Samples. Zur Optimierung verwenden wir Stochastic Gradient Descent mit Momentum und Weight Decay. Für jede Konfiguration werden 10 gleiche Netze unabhängig voneinander trainiert. Sie unterscheiden sich nur durch die zufällige Initialisierung ihrer Faltungskerne und die Reihenfolge der Training-Samples.

4.4 Inferenz

Standardmäßig werden die Validierungs- und Test-Sets auf die gleiche Weise in Samples aufgeteilt wie die Trainingsdaten, jedoch ohne Überlappung. Da bei der Inferenz von kleineren Samples an den Rändern möglicherweise wertvolle Nachbarschaftsinformation fehlen, nutzen wir die *fully convolutional* (LONG et al. 2015) Eigenschaft der Architektur, um die Genauigkeit von am Stück

klassifizierten Punktwolken zu untersuchen. Für bessere und stabilere Ergebnisse verwenden wir zudem Ensembles aus zehn Netzen, deren prädizierte Klassenwahrscheinlichkeiten gemittelt werden. Zugewiesen wird einem Voxel die Klasse mit der höchsten Wahrscheinlichkeit.

4.5 Implementierungsdetails

Implementiert wurden die Experimente in Python 3.5 mit PyTorch[3] 0.4 als Deep Learning Bibliothek. Das Framework für Submanifold Sparse Convolutional Networks von GRAHAM et al. (2018) ist öffentlich verfügbar[4]. Die Punktwolke wurde eingefärbt mittels OPALS (PFEIFER et al. 2014).

5 Ergebnisse und Diskussion

Tab. 1: Klassifizierungsgenauigkeit (Overall Accuracy, OA) für den Test-Set. Unter den Voxelgrößen ist jeweils die Anzahl an Voxeln im Test-Set sowie die reine Inferenz-Zeit für ein einzelnes Netz angegeben. Pro Konfiguration ausgewertet aus 10 Netzen.

Voxel OA [%] Punkte OA [%] (Mittel ± Std.-Abw. Ensemble)		Voxel-Größe [m]				
		2,0 (26k in 0,2 s)	1,0 (85k in 0,3 s)	0,5 (210k in 0,6 s)	0,25 (320k in 0,8 s)	0,125 (374k in 1,2 s)
Sample-Größe [m]	16 × 16 × 64	71,6 ± 0,7 74,1 76,3 ± 0,4 78,3	76,0 ± 1,3 78,0 80,2 ± 1,0 81,2	78,4 ± 1,8 80,6 80,3 ± 1,5 82,3	80,4 ± 1,3 82,9 80,7 ± 1,1 82,9	79,2 ± 1,3 82,0 79,2 ± 1,3 82,0
	32 × 32 × 64	72,5 ± 0,6 75,9 76,7 ± 1,0 79,8	77,4 ± 0,6 79,5 81,4 ± 0,5 83,1	79,9 ± 0,7 81,9 81,6 ± 0,6 **83,5**	80,7 ± 0,7 83,4 81,0 ± 0,7 83,2	78,8 ± 2,0 82,5 78,8 ± 2,0 82,4
	64 × 64 × 64	72,4 ± 1,2 75,0 77,0 ± 0,8 79,1	77,5 ± 0,8 79,7 81,4 ± 0,7 83,2	79,6 ± 0,7 81,7 81,4 ± 0,7 83,4	81,1 ± 1,0 83,4 81,5 ± 0,9 83,4	80,6 ± 1,4 83,6 80,5 ± 1,4 83,7

In Tabelle 1 werden die Klassifizierungsgenauigkeiten bei verschiedenen Auflösungen und Sample-Größen gezeigt. Es ist klar zu erkennen, dass die Ensembles erwartungsgemäß die besten Ergebnisse liefern. Außerdem fällt die Auswertung auf den ursprünglichen Punkten bei gröberen Auflösungen besser aus als auf den Voxeln. Überschreitet die Auflösung die mittlere Punktdichte der ursprünglichen Punktwolke, ist kein signifikanter Unterschied mehr vorhanden. Das beste Ergebnis von **83,5 %** liefert eine Sample-Größe von 32 × 32 × 64 m mit einer Voxelgröße von 0,5 m. Bei allen weiteren Untersuchungen bleibt dies die beste Konfiguration.

5.1 Inferenz via Samples vs. als Ganzes

Prädizierte man die Klassenlabel für den gesamten Test-Split am Stück, d.h. ohne zu samplen, verringerte sich die Klassifizierungsgenauigkeit bei Netzen, die auf 16 × 16 × 64 m großen Sam-

[3] https://pytorch.org/

[4] https://github.com/facebookresearch/SparseConvNet

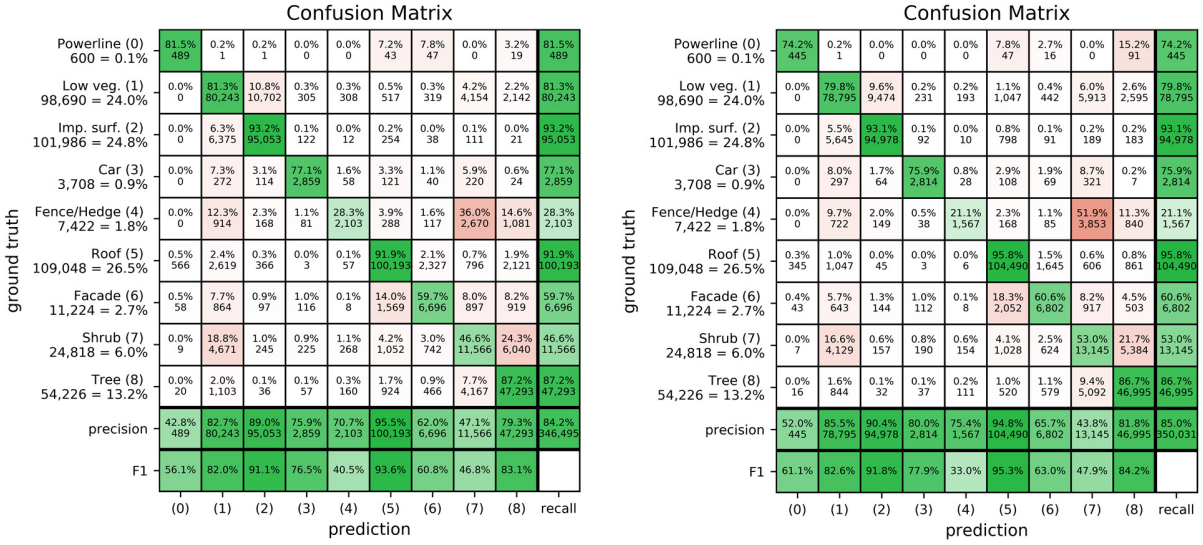
peln trainiert wurden, und zwar um durchschnittlich 1,8 Prozentpunkte. Bei 32 m bzw. 64 m Kantenlänge jedoch stieg sie um 0,8 bzw. 0,6. Das sich daraus ergebende beste Netzwerk hat dieselbe Konfiguration wie in Tabelle 1, erzielt jedoch **84,2 %** (Abb. 4a, 5b). Daraus lässt sich schließen, dass dieser Vorteil nur erzielt werden kann, wenn die Trainingsbeispiele ausreichend räumlichen Kontext besaßen.

5.2 Einfluss der Geometrie

Um den Einfluss der reinen Geometrie zu untersuchen, trainierten und testeten wir ein Netz der besten Konfiguration aus Tabelle 1, jedoch ohne die Echomerkmale. Jedes belegte Voxel besitzt dabei als einziges Merkmal den Wert 1. Die Klassifizierungsgenauigkeit beträgt **79,8 %** für ein Ensemble, bzw. etwa **75 %** für ein einzelnes Netz. Das größte Problem bereitet hier die Unterscheidung von Low Vegetation und Impervious Surface, beides sind Klassen mit sehr flacher und Bodennaher räumlicher Verteilung

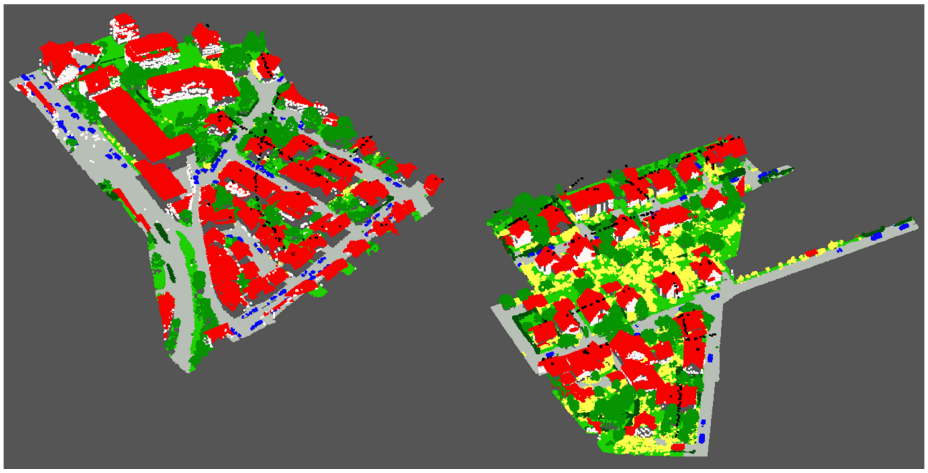
5.3 Einfluss spektraler Merkmale

Das führende Verfahren in der Benchmark-Liste des ISPRS 3D Semantic Labeling Contest verwendet eine mit spektralen Informationen angereicherte Punktwolke. Zum Vergleich wiederholten wir die Versuche der Sample-Größe 32 × 32 × 64 m ebenfalls mit zusätzlichen CIR-Werten aus dem zugehörigen TOP. Ein generelles Problem dabei sind die unterschiedlichen Aufnahmezeitpunkte von Bild und LiDAR-Punktwolke, was insbesondere bei Fahrzeugen zu falschen Einfärbungen führt. Des Weiteren werden Fassaden teilweiße wie die darüber liegenden Dächer eingefärbt.

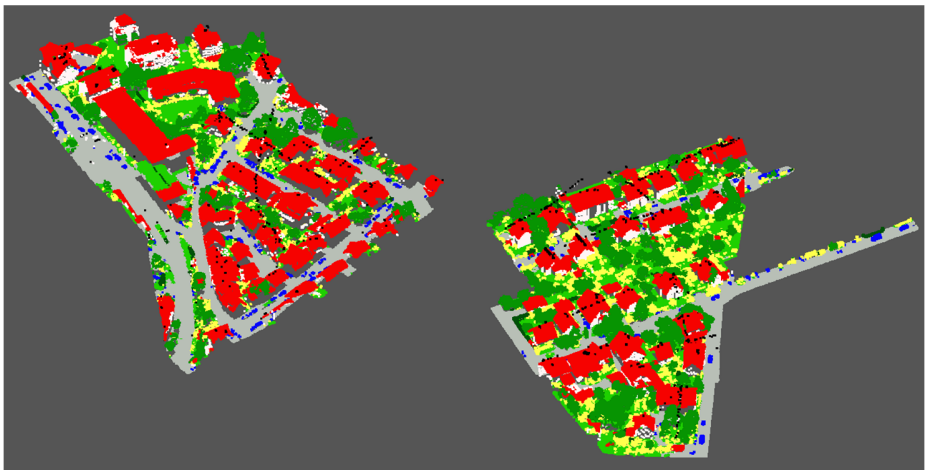


(a) Ohne CIR (b) Mit CIR

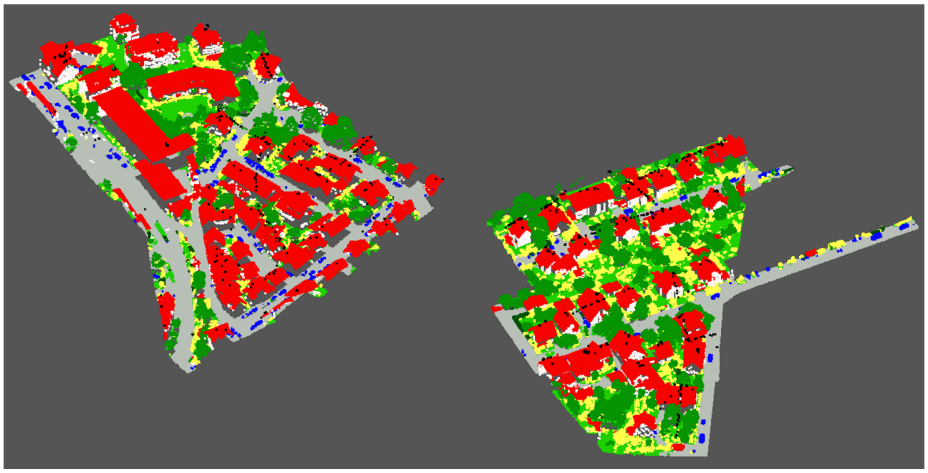
Abb. 4: Fehlermatrizen der klassifizierten Test-Punktwolke.



(a) Ground Truth



(b) Klassifizierungsergebnis ohne CIR



(c) Klassifizierungsergebnis mit CIR

Abb. 5: Visueller Vergleich der klassifizierten Punktwolke, eingefärbt nach Klassen: Powerline (schwarz), Low. veg. (hellgrün), Imp. surf. (grau), Car (blau), Fence/Hedge (dunkelgrün), Roof (rot), Facade (weiß), Shrub (gelbgrün), Tree (grün)

Bei der sampleweisen Inferenz steigt die OA im Mittel um 1,9 Prozentpunkte, insbesondere aber bei einer Voxelgröße von 2 m. Die beste Konfiguration (siehe Tabelle 1) steigert sich von 83,5 % auf **84,6 %**. Wird die Test-Voxelwolke als Ganzes vom SSCN prozessiert, steigt die Genauigkeit im Schnitt um 1,5 Prozentpunkte, das beste Ergebnis um 0,8 auf **85,0 %** (Abb. 4b, 5c). Damit erreichen wir knapp das zum Zeitpunkt dieser Arbeit führende Ergebnis des Benchmarks, welches 85,2 % erzielt.
Fassaden werden zwar etwas häufiger als Dächer interpretiert, dafür aber seltener als Vegetation. Die Verwechslungen zwischen Straße und Fahrzeugen wird sogar geringfügig besser.

5.4 Rechenzeit und Speicher

Das Training dauert pro Netz für die verschiedenen Voxelgrößen ca. 6, 14, 30, 60 bzw. 100 Minuten. Die reine Inferenzzeit für das beste Ensemble (Voxelgröße 0,5 m) beträgt 11 s, hinzu kommen für Auswertung und Input/Output 19 s, weitere 0,1 s für die Voxelierung (plus 4 s für I/O), sowie gegebenenfalls 5 s fürs Sampling. Die letzteren Zeiten bieten implementierungs-bedingt noch einigen Raum zur Optimierung.
Bei größeren Datenmengen kann Parallelität der GPU besser genutzt werden. Wir vergrößerten den Testdatensatz durch Kopieren künstlich auf ca. 10 Millionen Punkte (Faktor 25). Die reine Inferenzzeit betrugt daraufhin bei ansonsten gleicher Konfiguration etwa 100 s, also nur das 8-fache.
Wenn geringe Einbußen bei der Genauigkeit akzeptabel sind, erlaubt die Einstellung der Voxelgröße und der Anzahl an Netzen im Ensemble eine simple Abwägung zwischen Rechenzeit und Genauigkeit.
Bei allen Berechnungen kam die folgende Hardware zum Einsatz: Intel-Core i7-6800K @ 6/12x 3,40 GHz mit 64 GB RAM und eine NVIDIA Titan X Pascal mit 12 GB Grafikspeicher.

6 Fazit und Ausblick

In dieser Arbeit konnten wir die Eignung von Submanifold Sparse Convolutional Networks zur semantischen Segmentierung von ALS-Punktwolken zeigen. Die erreichte Klassifizierungs-genauigkeit von 85,0 % ist das zum Zeitpunkt dieser Arbeit zweitbeste veröffentlichte Benchmark-Ergebnis.
Seltene Objektkategorien können durch eine gewichtete Zielfunktion trotzdem gut erkannt werden. In zukünftigen Arbeiten könnten weitere Ansätze wie die Dice Zielfunktion (MILLETARI et al. 2016) untersucht werden.
Die implizite Geometrie der Punktwolke hat sich als das primäre Merkmal erwiesen. Schwierige Klassen sind insbesondere Sträucher und Hecken bzw. Zäune, die oft als andere Vegetationsarten interpretiert werden. Niedrige Vegetation und versiegelte Flächen weisen aufgrund ihrer ähnlichen Geometrie starke Verwechslungen auf. Fassaden sind in diesem Datensatz nur schlecht repräsentiert und würden von einem stärker schräg messenden System profitieren.
Der in dieser Arbeit verwendete Datensatz ist, verglichen mit den Datensätzen, auf denen Deep Learning Methoden üblicherweise trainiert werden, sehr klein. Dies macht das Training instabil und Generalisierung schwierig. In weiteren Arbeiten werden wir die Anwendung auf größere Datensätze untersuchen.

7 Danksagungen

Der Vaihingen Datensatz wurde zur Verfügung gestellt von der Deutschen Gesellschaft für Photogrammetrie und Fernerkundung (DGPF) [Cramer, 2010]: http://www.ifp.uni-stuttgart.de/dgpf/DKEP-Allg.html.
Die Titan X Pascal verwendet für diese Arbeit wurde gespendet von der NVIDIA Corporation.
Der Autor dankt Philipp-Roman Hirt für seine Unterstützung bei der Voxel-Visualisierung.

8 Literaturverzeichnis

BLOMLEY, R. & WEINMANN, M., 2017: Using Multi-Scale Features for the 3D Semantic Labeling of Airborne Laser Scanning. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, **IV**-2/W4, 43-55.

CHELLAPILLA, K., PURI, S. & SIMARD, P., 2006: High performance convolutional neural networks for document processing. Tenth International Workshop on Frontiers in Handwriting Recognition, Lorette, G. (Hrsg.), October, La Baule (France).

CHEHATA, N., GUO, L. & MALLET, C., 2009: Airborne Lidar feature Selection for urban classification using random forests. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, **38**(3/W8), 207-212.

CHETLUR, S., WOOLLEY, C., VANDERMERSCH, P., COHEN, J., TRAN, J., CATANZARO, B. & SHELHAMER E., 2014: cuDNN: Efficient Primitives for Deep Learning. CoRR (arXiv:1410.0759).

CRAMER, M., 2010: The DGPF-Test on Digital Airborne Camera Evaluation - Overview and Test Design. Photogrammetrie-Fernerkundung-Geoinformation, **2010**(2), 73-82.

EIGEN, D. & FERGUS, R., 2015: Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture. IEEE International Conference on Computer Vision, 2650-2658.

GRAHAM, B., 2015: Sparse 3D convolutional neural networks. Proceedings of the British Machine Vision Conference (BMVC), Xie, X., Jones, M.W., Tam, G.K.L (Hrsg.), BMVA Press, September, 150.1-150.9.

GRAHAM, B., ENGELCKE, M. & VAN DER MAATEN, L., 2018: Semantic Segmentation with Submanifold Sparse Convolutional Networks. IEEE Conference on Computer Vision and Pattern Recognition, 18-22.

HACKEL, T., WEGNER, J.D. & SCHINDLER, K., 2016: Fast Semantic Segmentation of 3D Point Clouds with Strongly Varying Density. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, **III**-3, 177-184.

HU, X. & YUAN, Y., 2016: Deep-Learning-Based Classification for DTM Extraction from ALS Point Cloud. Remote Sensing, **8**(9), Beitrag 730.

LONG, J., SHELHAMER, E. & DARREL, T., 2015: Fully convolutional networks for semantic segmentation. IEEE Conference on Computer Vision and Pattern Recognition, 3431-3440.

NIEMEYER, J., ROTTENSTEINER, F. & SÖRGEL, U., 2014: Contextual classification of lidar data and building object detection in urban areas. ISPRS Journal of Photogrammetry and Remote Sensing, **87**, 152-165.

NIEMEYER, J., ROTTENSTEINER, F., SOERGEL, U. & HEIPKE, C., 2016: Hierarchical Higher Order CRF for the Classification of Airborne LiDAR Point Clouds in Urban Areas. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, **41**(B3), 655-662.

MILLETARI, F., NAVAB, N. & AHMADI, S., 2016: V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. 2016 Fourth International Conference on 3D Vision (3DV), 565-571.

PFEIFER, N., MANDLBURGER, G., OTEPKA, J. & KAREL, W., 2014: OPALS - A framework for Airborne Laser Scanning data analysis. Computers, Environment and Urban Systems, **45**(2014), 125-136.

POLITZ, F. & SESTER, M., 2018: Exploring ALS and DIM Data for Semantic Segmentation using CNNs. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, **42**(1), 347-354.

RIZALDY, A., PERSELLO, C., GEVAERT, C.M. & OUDE ELBERINK, S.J., 2018: Fully Convolutional Network for Ground Classification from LiDAR Point Clouds. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Science, **IV**-2, 231-238.

RONNEBERGER, O., FISCHER, P. & BROX, T., 2015: U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention (MICCAI), **9351**(LNCS), Springer Verlag, 234-241.

YI, L., SHAO, L., SAVVA M., HUANG, H., ZHOU, Y. & WANG, Q. ET AL., 2017: Large-Scale 3D Shape Reconstruction and Segmentation from ShapeNet Core55. CoRR (arXiv:1710.06104).