remote sensing

Article

Generalized Sparse Convolutional Neural Networks for Semantic Segmentation of Point Clouds Derived from Tri-Stereo Satellite Imagery

Stefan Bachhofner [1,*], Ana-Maria Loghin [2,*], Johannes Otepka [2,*], Norbert Pfeifer [2,*], Michael Hornacek [3], Andrea Siposova [1], Niklas Schmidinger [1], Kurt Hornik [1], Nikolaus Schiller [4], Olaf Kähler [3] and Ronald Hochreiter [5]

1 Research Institute for Computational Methods, Vienna University of Economics and Business, Welthandelsplatz 1, 1020 Vienna, Austria; andrea.siposova@wu.ac.at (A.S.); niklas.schmidinger@wu.ac.at (N.S.); kurt.hornik@wu.ac.at (K.H.)

2 Department of Geodesy and Geoinformation, Vienna University of Technology, Gußhausstraße 27-29, 1040 Vienna, Austria

3 Corporate Technology, Siemens AG Österreich, Siemensstraße 90, 1210 Vienna, Austria; michael.hornacek@siemens.com (M.H.); olaf.kaehler@siemens.com (O.K.)

4 Vermessung Schmid ZT GmbH, Büropark Donau, Inkustraße 1-7, Stiege 3, EG, 3400 Klosterneuburg, Austria; n.schiller@geoserve.co.at

5 Department of Business and Management, Webster Vienna Private University, Praterstraße 23, 1020 Vienna, Austria; ronald.hochreiter@webster.ac.at

* Correspondence: stefan.bachhofner@wu.ac.at (S.B.); ana-maria.loghin@geo.tuwien.ac.at (A.-M.L.); johannes.otepka@geo.tuwien.ac.at (J.O.); norbert.pfeifer@geo.tuwien.ac.at (N.P.)

Received: 6 March 2020; Accepted: 13 April 2020; Published: 18 April 2020

Abstract: We studied the applicability of point clouds derived from tri-stereo satellite imagery for semantic segmentation for generalized sparse convolutional neural networks by the example of an Austrian study area. We examined, in particular, if the distorted geometric information, in addition to color, influences the performance of segmenting clutter, roads, buildings, trees, and vehicles. In this regard, we trained a fully convolutional neural network that uses generalized sparse convolution one time solely on 3D geometric information (i.e., 3D point cloud derived by dense image matching), and twice on 3D geometric as well as color information. In the first experiment, we did not use class weights, whereas in the second we did. We compared the results with a fully convolutional neural network that was trained on a 2D orthophoto, and a decision tree that was once trained on hand-crafted 3D geometric features, and once trained on hand-crafted 3D geometric as well as color features. The decision tree using hand-crafted features has been successfully applied to aerial laser scanning data in the literature. Hence, we compared our main interest of study, a representation learning technique, with another representation learning technique, and a non-representation learning technique. Our study area is located in Waldviertel, a region in Lower Austria. The territory is a hilly region covered mainly by forests, agriculture, and grasslands. Our classes of interest are heavily unbalanced. However, we did not use any data augmentation techniques to counter overfitting. For our study area, we reported that geometric and color information only improves the performance of the Generalized Sparse Convolutional Neural Network (GSCNN) on the dominant class, which leads to a higher overall performance in our case. We also found that training the network with median class weighting partially reverts the effects of adding color. The network also started to learn the classes with lower occurrences. The fully convolutional neural network that was trained on the 2D orthophoto generally outperforms the other two with a kappa score of over 90% and an average per class accuracy of 61%. However, the decision tree trained on colors and hand-crafted geometric features has a 2% higher accuracy for roads.

Keywords: 3D segmentation; deep learning; derived point clouds; tri-stereo; Very High Resolution (VHR) Satellite Imagery; 2.5D segmentation; image segmentation; semantic segmentation; machine learning

1. Introduction

Deep Learning (DL) has drastically improved the state-of-the-art across a variety of applications in Natural Language Processing (NLP) and Computer Vision (CV) [1–4] due to its ability to learn representations in an unsupervised manner [5–7]. In NLP, for example, Tshitoyan et al. showed that unsupervised deep learning is capable of learning word embeddings that capture complex materials science concepts without the insertion of chemical knowledge, and can even recommend materials years before their discovery [8]. There are also recent improvements in language understanding with Bidirectional Encoder Representations from Transformers (BERT) [9] and A Robustly Optimized BERT Pretraining Approach (RoBERTa) [10], and, adding to that, the recent breakthrough in task agnostic transfer learning by Howard et al. [11]. In CV, DL has advanced, inter alia, the tasks of image classification [12,13], object-detection [14–16], object-tracking [17], pose estimation [18–21], superresolution [22], and semantic segmentation [23–28]. These advancements give rise to new applications in, e.g., solid-state materials science and chemical sciences [29,30], meteorology [31], medicine [32–39], seismology [40–42], biology [43], life sciences in general [44], chemistry [45], and physics [46–53], as well as the fashion industry [54–56].

Due to this success, DL also gained popularity in the fields of photogrammetry and remote sensing [57–59], as it plays a key role in a variety of applications. Such applications include, inter alia, classification and segmentation [60–62], urban construction and planning [63–66], agriculture [67–70], the intersection of archaeology and remote sensing [71], predicting poverty [72–74], natural disaster and crisis management, and infrastructure management [75,76]. However, little research has thus far explored the utility of point clouds derived from tri-stereo satellite imagery for DL. One reason for this may be the fact that cross-correlation, the way the convolution operation is implemented in DL libraries, is not directly applicable to the three-dimensional space. In addition, adding geometric information from tri-stereo satellite imagery is a challenging machine learning problem, because the 3D geometry quality is low The poor geometric precision is caused by noise and low radiometric resolution of satellite imagery, as well as smoothing effects of adopted dense image matching algorithms. Nevertheless, this extra geometric information can be a valuable source of features for machine learning algorithms.

Scene understanding based on point clouds derived from tri-stereo satellite imagery is potentially a valuable information source for businesses, as it can enable business processes which were not possible before, or bring efficiency and effectiveness gains to existing ones. For example, businesses can conduct measurement tasks for a construction site where aerial laser scans are not possible, e.g., due to restrictions in flight permission or time constraints. Moreover, businesses can use it for surveillance tasks, e.g., nearby oil pipelines. The military can use this information to protect embassies in the case of an emergency or in the case of a natural disaster. Consequently, point clouds derived form tri-stereo satellite imagery can bring value for companies and public institutions alike, which can incorporate it in the life cycles of their business processes [77], in, e.g., process monitoring [78,79], process compliance [80], process conformance checking [81,82], process mining [83–86], and performance mining [87–89].

In this paper, we study the applicability of point clouds derived from tri-stereo satellite imagery for GSCNNs. To the best of our knowledge, this is the first work that applies DL to derived point clouds from tri-stereo satellite imagery. We learn directly on geometric information without the need to transform it into a format that is suitable for the standard convolution operation. In particular, we train a Fully Convolutional Neural Network (FCNN) [90] that uses generalized sparse convolution as introduced by Choy et al. [91] to distinguish the classes $\mathscr{C}$ road, building, vegetation, vehicle, and clutter, i.e., $\mathscr{C} = \{clutter, road, building, tree, vehicle\}$, where clutter refers to points that do not belong to any of the other four categories. The network architecture follows the U-Net structure from

Ronneberger et al. with an encoder followed by a decoder that is connected via skip connections [92]. In addition, the model incorporates residual learning [93], Rectifier Linear Units (ReLUs) [94,95], and batch normalization [96,97]. We compare the performance of this model to FCN-8s [90] that is trained on an orthophoto and to a decision tree that uses hand-crafted geometric features [98] and color. Hence, we compare the model to a representation learning technique, and once to a non-representation learning technique. In doing so, we report the average precision [99], average recall [99], average F1 score [99,100], average accuracy, overall accuracy [101], kappa score [102], Matthew's Correlation Coefficient (MCC) [103,104], and confusion matrices. We use the Minkowski Engine [91], which is built on top of PyTorch [105], for training the GSCNN. For FCN-8s, we use Caffe [106]. We use R to train the decision tree [107], and the software package OPALS (https://opals.geo.tuwien.ac.at) for computing the hand-crafted geometric features [108]. All trained GSCNNs are made public on PyTorch Hub after each epoch to facilitate research and ease reproducibility on PyTorch Hub. We do the same for the decision trees. For further information, please visit the GitHub project (https://github.com/MacOS/ReKlaSat-3D/) for this paper. A general overview of the workflow is depicted in Figure 1.

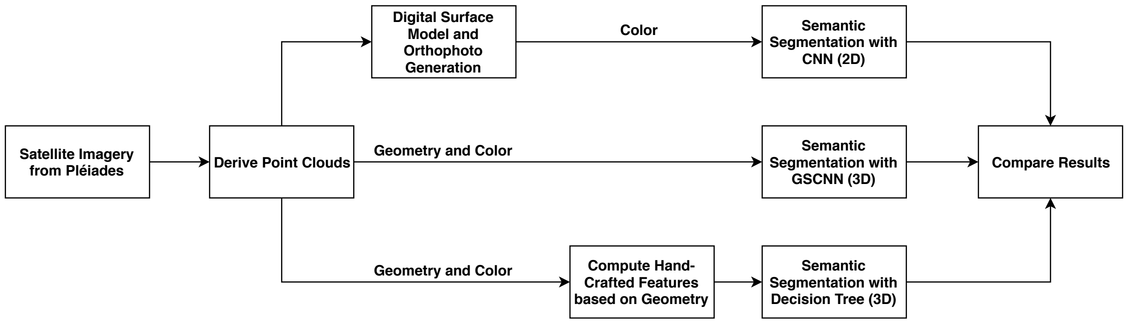
Digital Surface Model and Orthophoto Generation
Color
Semantic Segmentation with CNN (2D)
Satellite Imagery from Pléiades
Derive Point Clouds
Geometry and Color
Semantic Segmentation with GSCNN (3D)
Compare Results
Geometry and Color
Compute Hand-Crafted Features based on Geometry
Semantic Segmentation with Decision Tree (3D)

Figure 1. Overview of the general workflow. We use tri-stereo Pléiades images of the study area, Waldviertel. The images are then processed to derive the 3D point cloud and the corresponding orthophoto. Whereas the Convolutional Neural Network (CNN) approach utilizes the orthophoto, the GSCNN semantic segmentation uses the 3D point cloud as input. For the decision tree, we compute hand-crafted geometric features while we leave the color features as they are. Finally, we compare the GSCNN results with the CNN approach and the decision tree, i.e., a strong baseline (Section 4.2).

The remainder of this paper is structured as follows. In Section 2, we present the background and related work. This includes 3D reconstruction, DL on satellite imagery, the difficulty for DL of learning directly on point clouds, and a discussion about reproducibility and choosing baselines in machine learning. In Section 3, we describe the study site and our dataset, the photogrammetric workflow, our DL architecture and training procedure for the point clouds, and our steps towards a more scientific approach in machine learning. We present our results in Section 4. In Section 5, we discuss our results in the light of generalizing the convolution operation from the two-dimensional grid world to higher dimensions, and we end this section with future research directions. Finally, we conclude and summarize this paper in Section 6.

2. Background and Related Work

Section 2.1 describes the photogrammetric workflow of point cloud reconstruction from tri-stereo satellite images. Section 2.2 gives an overview of how deep learning is currently used and studied in remote sensing. Section 2.3 focuses on DL on point clouds. Finally, Section 2.4 discusses reproducibility in science, especially in machine learning, and the importance of choosing baselines.

2.1. 3D Reconstruction from Satellite Images

The photogrammetric workflow including the 3D reconstruction was performed using the Match-AT and Match-T Digital Surface Model (DSM) modules of the Trimble Inpho software [109]. The software is designed to perform precise image block triangulation, point cloud reconstruction, and orthophoto generation. Overall, the image processing chain consists of the following main steps: (1) import of the images, Rational Polynomial Coefficients (RPCs), and Ground Control Points (GCPs), followed by image pyramids generation; (2) identification of GCPs locations within the images; (3) RPCs refinement based on the GCPs and automatically extracted tie points; (4) dense image matching for 3D reconstruction; and (5) point cloud interpolation for DSM derivation. Finally, the DSM is used for orthorectification of a Nadir image, which is further considered as input data in the FCN-8s approach. The software is highly automated, and hence the user is limited to a few adjusting steps, e.g., the manual identification of GCPs in images and correction of points with high residuals. For dense image matching two strategies were employed [110]: Feature Based Matching (FBM) and Cost Based Matching (CBM). The output of dense image matching are 3D point clouds, which are further used as input for DSM derivation.

2.2. Deep Learning on Satellite Images

DL has been extensively applied on remote-sensing imagery in recent years [57]. Hu et al. investigated the generalization properties of CNN extracted features from ImageNet to high-resolution remote sensing images for scene classification [111]. In that regard, the authors proposed two scenarios for extracting features: one for extracting features from fully-connected layers, and one for extracting features from convolutional layers. The extracted features are then used as an input for a linear Support Vector Machine (SVM). This linear classifier is evaluated on the UC Merced Land Use Dataset and the WHU-RS dataset. According to the paper, the linear SVM outperforms current state-of-the art hand-crafted feature approaches by a large margin, indicating that the features learned by the CNN have better generalization properties, even when the features are learned from a different domain, in this case from ImageNet natural images to the high-resolution remote sensing images from UC Merced Land Use and WHU-RS datasets. Sun et al. explored the use of encoder-decoder architectures for semantic image segmentation [112]. Bischke et al. applied DL to aerial imagery and addressed the problem of sharp object boundaries by proposing a cascaded multi-task loss [113]. Similarly, Yi et al. used a U-Net architecture with Res-Net blocks for urban building segmentation in VHR remote sensing imagery [114]. Guo et al. also segmented buildings but incorporate superresolution in their approach, which shows promising results as the authors report an improvement of roughly 19% for the Jaccard index as well as the kappa score [115]. Eerapu et al. proposed a dense refinement residual network (DRR Net) for road extraction in aerial imagery [116]. In addition, the authors trained their network on a composite loss function surface to place emphasis on small objects. Marmanis et al. studied the use of an ensemble of CNN for semantic segmentation [117].

Li et al. used three-dimensional convolutions for Hyperspectral Image (HSI) semantic segmentation [118]. CNN architectures that use three-dimensional convolutions are called 3D-CNN. They are different from 2D-CNN, as 3D-CNN convolve the spatial and the spectral dimension simultaneously with three-dimensional kernels, while 2D-CNN use two-dimensional kernels. Sellami et al. also trained a 3D-CNN on semantic segmentation and proposed a dimensionality reduction technique for adaptively selecting spectral bands before training the 3D-CNN [119].

2.3. Deep Learning on Point Clouds

Applying DL on point clouds is not straightforward since point clouds do not have two-dimensional grid structures, which is assumed by standard convolutional neural networks. In addition, images have a uniform density distribution, but point clouds do not. Therefore, it is not a surprise that previous research in remote sensing used other machine learning techniques, such as decision trees [98] and random forests [120]. Some researchers focus on developing features (i.e., feature engineering) to boost the performance of these machine learning techniques [121]. However, in CV, many researchers propose techniques that map the original point cloud to a two-dimensional grid structure as an alternative. Zhou et al. proposed dividing the space into three-dimensional voxels, and, for each of these voxels, a feature representation is learned by a voxel feature encoding layer [122]. This leads to a grid structure of features and hence allows the use of standard convolutional layers. Qi et al. proposed a network that learns on unordered point sets by applying symmetric functions [60]. Since the ordering in a set is of no importance, their solution does not assume that neighbors in the input tensor are actually neighbors in the point cloud. However, recent work focuses on the convolution operation itself by altering it such that it can directly be applied on the point cloud. Hua et al. proposed a pointwise convolution which queries the neighborhood of a point on the fly, bins the neighbors into kernel cells, and then applies the convolution operation [123]. While this operation is promising, it is currently difficult to apply on large scale as the current implementation is not optimized.

Vo et al. presented the results of the 2015 Institute of Electrical and Electronics Engineers (IEEE) data fusion contest where participants used Light Detection And Ranging (LiDAR) and RGB data simultaneously [124]. For road detection, Vo et al. applied decision trees on hand crafted features, aerial laser scanning and imagery is the basis for the hand created features [125]. This is similar to our work from the data perspective, because we also investigate the combination of geometric and color information for machine learning in remote sensing. However, our contribution is significantly different for three reasons. First, our geometric information is substantially inferior to that of airborne or terrestrial LiDAR, since it is derived from stereo matching applied to tri-stereo satellite imagery at 0.7 m Ground Sample Distance (GSD). Second, we use DL. Third, we apply DL directly on the point cloud without any preprocessing step, e.g., using voxels.

2.4. Reproducibility and Baseline (Scientific Approach in ML)

Machine Learning (see Maxwell et al. [126] for a practical review of machine learning in remote sensing), and especially DL, has made substantial progress over the past decade. Adding to the examples in the Introduction, Silfer et al. used reinforcement learning, an approach to learning that uses the reward and punishment paradigm [127], to master the game of Go [128]. DL was a crucial part of the work of Nadell et al., as they used it to discover metamaterial designs for energy harvesting [129]. The authors reported an average mean squared error of 1.16×10^{-3}. In addition, the neural network is over five orders of magnitude faster than conventional electromagnetic simulation software. In medicine, the recent work of Stroke et al. shows that DL can discover antibiotics which are structurally divergent (in this case, e.g., halicin) from conventional antibiotics [130]. The researchers tested halicins bacterial activity and found it to work against tuberculosis and strains considered untreatable. However, scholars working in the field have recently pointed out worrying trends [131–133]. Dacrema et al. analyzed the progress of recent neural recommendation approaches [134]. They reported that only 7 out of 18 examined neural recommendation approaches can be reproduced with reasonable effort. In addition, six of these seven can be outperformed with simple heuristic methods such as nearest-neighbor. Fu et al. argued similarly, as they claimed that combining SVM with differential evolution (a method for tuning hyper parameters) achieves similar or sometimes better results than DL while having a significantly lower training time (10 min vs. 14 h) [135]. While the observation of a lack of reproducibility in machine learning, and deep learning specifically, is worrying, it is not unique to this field. Other research fields suffer from the same issue. For example, a reproducibility test conducted in Psychology revealed that over 50% of studies in the field are not reproducible, and two thirds of them should be distrusted [136]. Moreover, a survey in 2016 revealed that 70% of researchers fail to reproduce another scientists experiments and more than 50% fail to reproduce even their own experiments [137]. Gannot et al. gave an equally worrying overview of reproducibility and transparency in biomedical sciences [138]. This controversy invoked a discussion about the question "*What does research reproducibility mean?*" [139] and a discussion about how scientists can avoid fooling themselves [140]. Menke et al. used SciScore (https://www.sciscore.com/), a software tool that automatically scores a research paper on a ten point scale, to analyze 1.6 million PubMed Central papers [141,142]. They also used a Rigor and Transparency Index, which is the average score of analyzed papers in a particular journal. They found that fewer than half of the analyzed research papers address rigor and reproducibility criteria. Furthermore, perhaps surprisingly, their Rigor and Transparency Index did not correlate with the impact factors of journals. In summary, reproducing DL related research takes a sizeable amount of resources, if it is possible at all based on the information given in that research article, and the way baselines are chosen is questionable. We present our approach to address and resolve these issues in the next section, specifically in Section 3.2.

3. Materials and Methods

In this section, we describe how we derive the point clouds and the orthophoto from tri-stereo satellite images, the study site and the dataset, our approach for semantic segmentation in two dimensions, and our approach for semantic segmentation in three dimensions. In Section 3.1, we describe the study site and the dataset, followed by Section 3.2 where we approach to ensure reproducibility and describe our approach for choosing baselines. Finally, in Section 3.3, we present the generalization of the convolution operation and the used U-Net based architecture.

3.1. Study Site and Pléiades Dataset

Pléiades is a European VHR satellite system comprising two identical satellites, Pléiades 1A and Pléiades 1B, which were developed by Airbus Defence and Space for both civil and military uses. With a daily revisit capacity and a swath width of 20 km, the sensor can reach a ground resolution of 0.7 m (in panchromatic mode) from an orbit height of 674 km. Due to its fast rotation, the sensor is able to record three images within the same orbit in less than 1 min. The images provided for this work were acquired in the tri-stereo mode, from different along-track positions of the Pléiades-1B sensor: forward (FW)-, close to nadir (N)-, and backward (BW)-view.

The study area is located in Waldviertel, a region in Lower Austria, which is the northeastern state of the country (48°30′30″ N; 15°08′34″ E; WGS84) (Figure 2). With elevations ranging from a minimum of 572 m above see level (a.s.l.) to a maximum of 749 m a.s.l., the territory is a hilly region covered mainly by forests, agriculture and grasslands. Rural, urban, and water surfaces are also present, but with a lower frequency of occurrence. For this study site, Pléiades images were acquired in the late morning of June 13th 2017. in the north–south direction, within less than 1 min. Therefore, they have the same illumination conditions, and shadow changes are not significant. Images were delivered as pan-sharpened with four bands (Red, Green, Blue, and Near-infrared) and spatial resolutions varying between 0.70 and 0.71 m, depending on the different viewing angles. The images were distributed as a primary product at sensor processing level, i.e., they contain corrections for sensor geometry and radiometric distortion, viewing angles, attitude, and ephemeris data. Detailed information about the acquisition properties of the tri-stereo pair are summarized in Table 1.

For each image, auxiliary data including the RPCs were provided by the supplier. To obtain sub-meter geolocation accuracy, the RPCs were refined by measuring 60 GCPs, i.e., points with known 3D coordinates, within the images. A Digital Terrain Model (DTM) generated from a LiDAR acquisition (December 2015) with 1 m spatial resolution was used for vertical quality assessment and for improving the absolute geolocation of the photogrammetrically derived DSM. We used a digital orthophoto from 2017 at 0.20 m spatial resolution for manually defining the planar locations of Check Points (CPs), whose corresponding heights were extracted from the LiDAR DTM. For this study, we selected a smaller test area (44.5 km^2) on which we trained and evaluated the generalized sparse convolutional neural network (Figure 2).

Table 1. Acquisition properties for the satellite images over the study site.

View	Acquisition Time	GSD [m]	Incidence Angles [°]		Sun Angles [°]		Area [km^2]
			Across	Along	Azimuth	Altitude	
FW	10:09:51.5	0.71	−2.2	−6.7	154.1		158.7
N	10:10:03.7	0.70	−3.31	−1.1	154.5	62.8	158.4
BW	10:10:14.0	0.71	−5.0	4.9	154.5		158.8

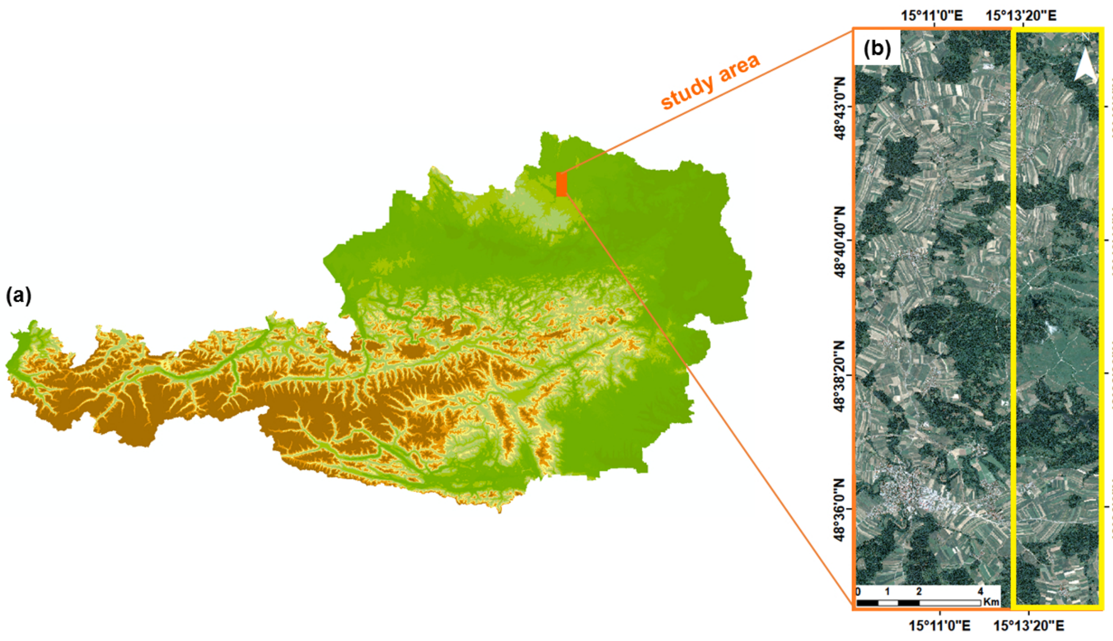

Figure 2. Waldviertel, Lower Austria: (**a**) Overview map of Austria with marked location of study area; and (**b**) Pléiades orthophoto of Waldviertel; the selected area used for semantic segmentation is marked with yellow.

The characteristics for the point cloud dataset are as follows. The whole dataset consists of 196,495,815 points. We followed the established rule of thumb in machine learning, where one splits the dataset into two parts: one part is roughly 80% of the total dataset and the other is roughly 20%. The former is our training dataset, and the latter our test dataset. Based on two polygon regions, the total number of points in the training set is 163,013,266 (82.96%), while the number of points in the test is 33,482,549 (17.03%). The same polygon regions were used to split the orthophoto as well. As shown in Figures 2 and 3, the class distribution is approximately the same for the training set as well as the test set. For the class weights experiment, we used median class weighting. This weighting approach weights each classes frequency $freq_c$ in relation to the median of the class frequencies $median(freq_c)$, in our case to 2,769,222, the occurrence of roads (Table 2). Mathematically, this is written as

$$c_{weight} = \frac{median(freq_c)}{freq_c}$$

These class weights hence downgrade the importance of classes with a high occurrence, and upgrades the importance of classes with a low occurrence. For example, for clutter, this equation is equal to $\frac{2769222}{98718944} = 0.02$, and, for vehicles, to 329.90. The level of gemetric distortion is clearly displayed in Figure 4, and Figure 5 shows the distribution for each feature.

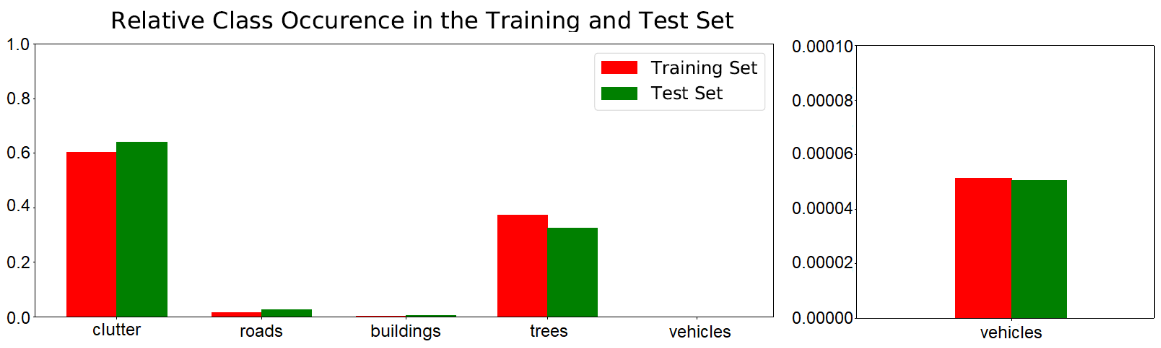

Figure 3. Relative occurrence of the classes in the training (red) and test (green) set: the magnitude of class imbalance is here highlighted as the vehicle bars are not visible (**left**); and by changing the scaling, they are visible (**right**).

Table 2. Quantitative comparison of the training data set with the test set. The right column shows the class weights based on the training set.

Class	Training Data Set	Test Data Set	Total	Class Weight
Clutter	98,718,944	21,512,925	120,231,869	0.02
Roads	2,769,222	907,680	3,676,902	1.00
Buildings	818,172	135,590	953,762	3.38
Trees	60,698,534	10,924,660	71,623,194	0.04
Vehicles	8394	1694	10,088	329.90
Total	163,013,266	33,482,549	196,495,815	

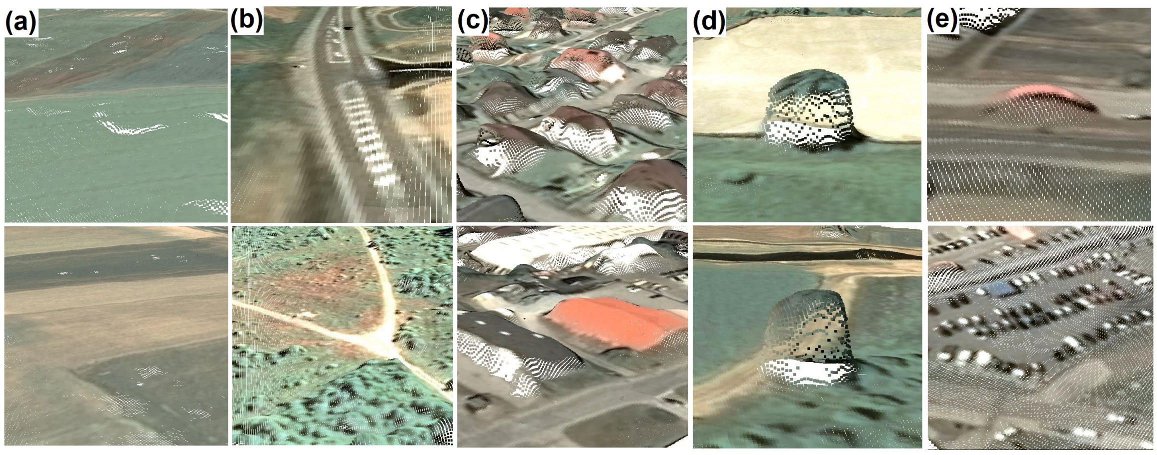

Figure 4. Examples of point clouds derived from tri-stereo satellite imagery for each class: (**a**) clutter; (**b**) roads; (**c**) buildings; (**d**) trees; and (**e**) vehicles.

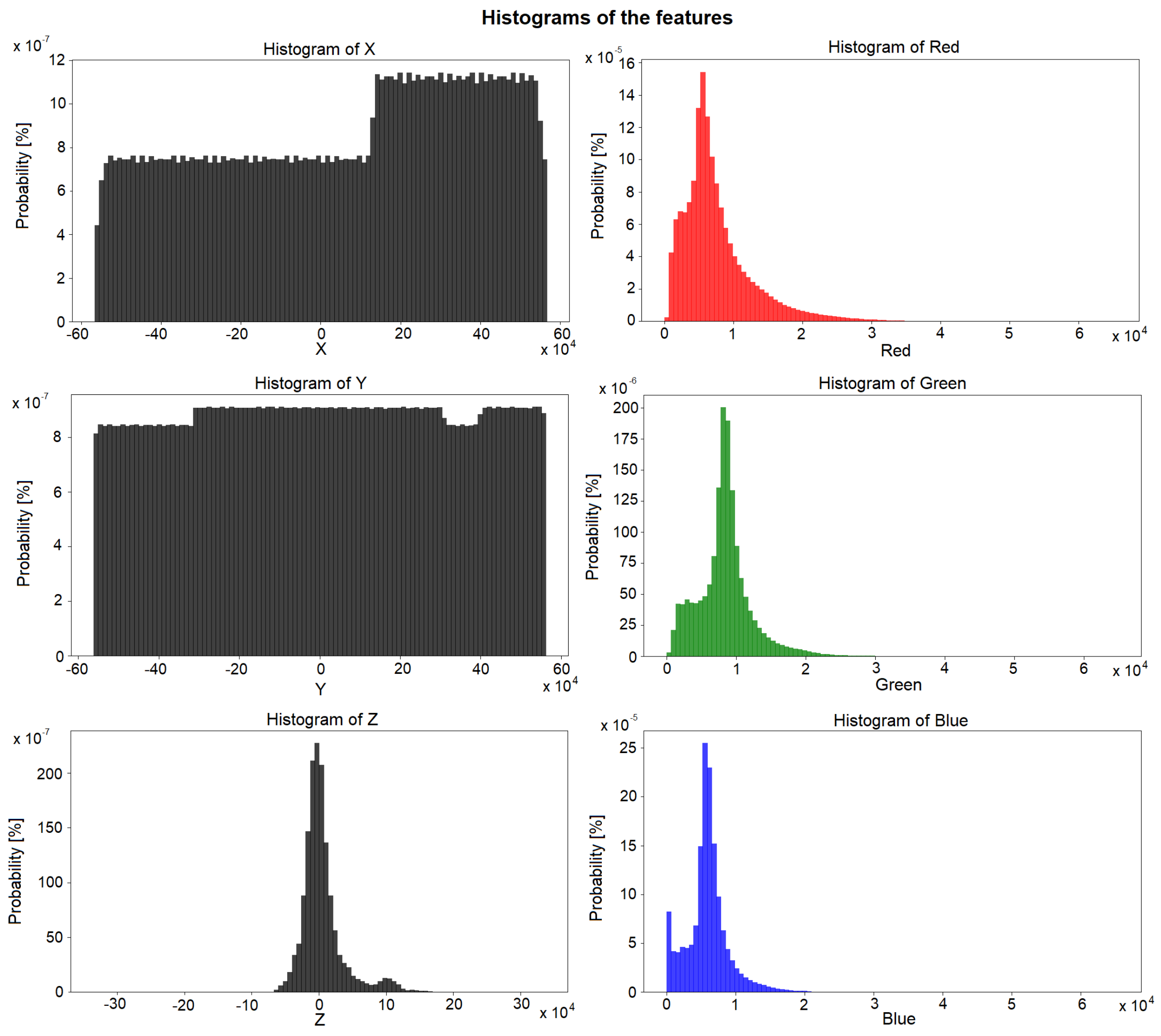

Figure 5. Training set feature distribution.

3.2. *Our Approach for Reproducibility and Choosing Baselines*

In line with the reproducibility discussion and to address the problem of reproducibility, we clearly state that our study falls into the *results reproducibility* category because our data are private, as of this writing. Results reproducibility means, in the context of this paper, that one gets approximately the same results when using approximately the same data and the same training procedure. That said, we acknowledge the fact that acquiring approximately the same dataset requires sizeable amount of resources. However, we are actively working on open sourcing our data in the near future.

To ensure reproducibility and comparability of our GSCNN experiments, we took two actions. The first action is to set seeds for random number generators. This means that, in our case, we set seeds for the Python libraries numpy, as well as for PyTorch with numpy.seed(0) and torch.manual_seed(0), respectively. The second action was to set cuDNN in the deterministic mode with torch.backends.cudnn.deterministic = True and torch.backends.cudnn.benchmark = False. This ensures that cuDNN only uses deterministic functionality and, further, that cuDNN does not automatically choose the best algorithm.

To further address the issues and facilitate research and ease reproducibility, we made all 3D models after each epoch public on PyTorch Hub. Specifically, we open-sourced all obtained network parameters after each epoch for all cases, which totals 500 sets of network parameters (50 from the coordinates experiment, 50 from the coordinates and color experiment, and 400 from the coordinates and color and class weights experiment). This decreases the amount of resources researchers, or others, need to allocate if they want to use or validate our research. For example, if other researchers also work on three-dimensional semantic segmentation with their own dataset and they would like to compare their approach with our method, they only need to download the weights instead of going through the training process themselves. Additionally, we see this as a first step towards transfer learning [7] in the domain of semantic segmentation on derived point clouds from satellite imagery with convolutional neural networks. For further information on system requirements, installation process, and how the weights can be downloaded, please visit the GitHub project (https://github.com/MacOS/ReKlaSat-3D/) for this paper.

As pointed out in Section 2.4, establishing a simple and strong baseline is important. For this reason, we established a strong baseline for the test set in general and for each class. For the overall accuracy, we set 60.56% as the first naive mark for the training set, because we can always achieve that by predicting clutter. We achieve an even better overall accuracy for the test set with that approach (64.25%) (Table 3). This is marked as baseline A. While this prediction method is not useful in practice, it is certainly a good reference point and highlights shortcomings of the respective performance metrics. Additionally, we compared the results with a decision tree approach, a supervised-feature learning algorithm [143]. In particular, we used the approach described by Waldhauser et al. [98]. The list of hand-crafted geometric features include, inter alia, the normal vector including its standard deviation (NormalX, NormalY, NormalZ, and NormalSigma), a local penetration rate (EchoRatio), statistical measures of the local vertical point distribution (ZRank and ZRange), and features derived from the structure tensor T (linearity, planarity, and omnivariance) [144]. For the training phase, we used 20% of the training data for training and the rest for cross validation. For the computation of the hand-crafted geometric features, Opals (Orientation and Processing of Airborne Laser Scanning Data) software is used [108]. We trained the decision tree with the following hyperparameters: (i) 0.00001 complexity factor; (ii) to attempt a split, a minimum of 20 observations has to exist in a node; (iii) each leaf node has at least seven observations; (iv) a maximum depth of 30 for the tree; (v) five competitor splits; and (vi) five surrogate splits. Our hardware set up for this is an Intel Core i7-4770K CPU @ 3.50 GHz, 32.0 GB RAM. Computing the features with this configuration takes 20 h 10 min, while the training time is 7 h 25 min for the geometric features and 9 h 22 min for the geometric and color features. The validation process takes 21 and 23 min for geometric and geometric and color features, respectively. More information on this approach is described in Appendix C.

We used FCN-8s, a FCNN with the standard convolution operation for semantic segmentation of the orthophoto, as proposed by Long et al. [90]. This network architecture combines features from three different levels of granularity for the prediction step. The first level of granularity is from the last layer, hence these features have the lowest level of granularity. These features have to be upsampled with a stride of 32. The second level of granularity are the features of the fourth out of five pool layers, and require to be upsampled with a stride of 16. Finally, the layer with the highest level of granularity to be used for prediction are the features from the third pool layer. These features are upsampled with a stride of 8, where the name FCN-8s originates from. Combining features with different levels of granularity improves the quality of segmentation, e.g., with sharper object boundaries. We used this network to assign a label to each pixel in the image I. If N and M denote the height and width of an Image in pixels I, and C the number of color channels for each pixel, then I can be interpreted as a tensor I with dimensions N, M, C, i.e., I^{NxMxC}. The networks input features are red, green, and blue for each pixel. Consequently, we searched for $\phi : I^{NxMx3} \rightarrow [clutter, road, building, vegetation, vehicle]^{NxM}$. We trained the network with a fixed learning rate of 0.0001 for 50 epochs with Caffe [106]. Appendix D and Figure A3 present a more detailed description of this network.

3.3. Generalized Sparse Convolutional Neural Networks for Derived Point Clouds

In this work, we are interested in the feature extraction capabilities of the generalized sparse convolution operation, as proposed by Choy et al. [91]. Generalized sparse convolution generalizes convolution for generic input and output coordinates, i.e., this operation does not have the implicit grid assumption of the standard convolution operation. In addition, it allows defining arbitrary kernel shapes. The conventional dense convolution in D-Dimensions is defined as:

$$x_u^{out} = \sum_{i \in \mathcal{V}^D(K)} W_i x_{u+i}^{in} \qquad \text{for } \mathbf{u} \in \mathbb{Z}^D$$

where $\mathcal{V}^D(K)$ is the list of offsets in a D-dimensional hypercube. Choy et al. relaxed this equation by allowing arbitrary kernel shapes using predefined input and output coordinates, $\mathcal{C}^{in}$ and $\mathcal{C}^{out}$. This leads to:

$$x_u^{out} = \sum_{i \in \mathcal{N}^D(u, \mathcal{C}^{in})} W_i x_{u+i}^{in} \qquad \text{for } \mathbf{u} \in \mathcal{C}^{out}$$

$\mathcal{N}^D$ is a set that defines the offsets of a kernel at kernel center u, i.e., arbitrary kernel shapes can be defined. This set is restricted to $\mathcal{N}^D(u, \mathcal{C}^{in}) = \{i | u + i \in \mathcal{C}^{in}, i \in \mathcal{N}^D\}$ in the equation, i.e., the current coordinate center plus the offset has to be a coordinate in the input coordinates. Trivially, the offset needs to be defined. We use this formulation as a drop in replacement for the conventional dense convolution.

The network architecture we use in this paper is U-Net based [92] (Figure 6). A variant of this architecture is, at the time of this writing, state-of-the-art on the ScanNet dataset [145]. This network architecture only uses hypercubes as its kernel shape. The circled plus represents element wise tensor addition, and concatenate takes two tensors as an input and appends the two tensors along the channel dimension. For example, the first concatenate from the top receives two tensors as an input: the output tensor of the first sparse convolution down-sampling layer with 32 feature channels via the skip connection and the output tensor of the last sparse transpose convolutional layer with 96 feature channels. Hence, the output of concatenate is a tensor with $32 + 96 = 128$ feature channels. Numbers outside of rectangles indicate the number of feature channels that go in and out. For example, the first layer expects C input channels and outputs 32 channels. All sparse convolutional layers are followed by batch normalization, and, in addition, all have a dilation of 1. A sparse convolution layer has a stride of 1, if not stated otherwise. For a better understanding, we merge layers into the four buildings blocks: SparseConv ResNet Block (green), SparseConv ResNet Block 2 (blue), SpraseConv Downsampling (yellow), and SpraseConv Transpose (red). A SparseConv ResNet Block applies a sparse convolution, with kernel size 3, two times on the input tensor. The resulting tensor is then combined with the input tensor with element wise tensor addition before the ReLU operation. SparseConv ResNet Block 2 deviates from this in one way: it doubles the number of feature channels it receives in the first sparse convolutional layer. We therefore need an extra sparse convolutional layer, with kernel size 1, in the side stream before we can perform element wise tensor addition. SparseConv Downsampling is a sparse convolutional layer with kernel size 2, and, additionally, a stride of 2, which halves the feature maps, i.e., if the input is N points, it outputs $N/2$. This operation is reversed by SparseConv Transpose with the same hyper parameters. We describe the encoder (also down path) and the decoder (also up path) in the next two paragraphs, respectively.

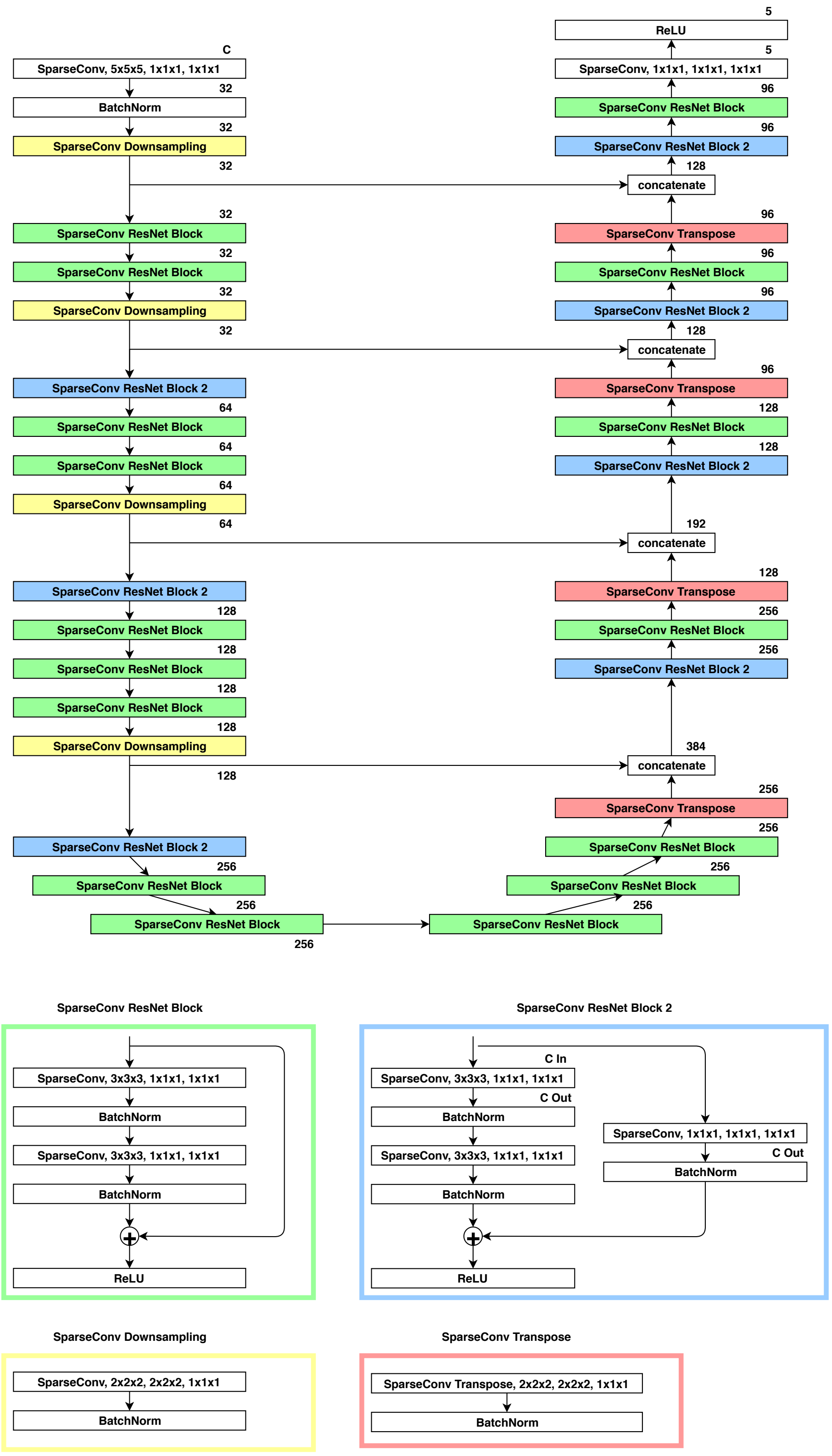

Figure 6. The network architecture used for semantic segmentation of the derived point cloud from tri-stereo satellite images. From its basic structure, it is a U-Net with its encoder and decoder paths, and the skip connections between the two. Further, it incorporates residual learning from Res-Nets in the sparse convolutional blocks and batch normalization.

The encoder, generally, takes the input tensor and down-samples it, while it doubles the feature maps. First, the network performs a sparse convolutional operation on the provided features of the input point cloud with kernel size 5 with 32 kernels, expanding the number of feature channels from C to 32. The kernel size of 5 allows the network to capture patterns in a larger neighborhood. Next, SparseConv Downsampling halves the size of the feature maps. This pattern, i.e., first downsampling the feature maps and then doubling the number of feature maps is repeated four times in total, with a growing number of ResNet Blocks between them: specifically, first by 2, then by 3, and then by 4. The last doubling of feature maps takes place in the first layer of the bottleneck (last blue rectangle in the encoder) which consumes 128 and outputs 256. This is followed by five SparseConv ResNet Blocks. Hence, the bottleneck consists of six ResNet blocks.

The decoder, generally, takes a tensor and upsamples it, while it halves the number of feature maps. The decoder's first layer is the first SparseConv Transpose layer from the bottom, it doubles the size of the tensor it receives while keeping the number of feature maps at 256. The resulting tensor is then concatenated with the output tensor of the encoder's last SparseConv Downsampling layer, which gives us a total of $384 = 256 + 128$. Following that, the first SparseConv ResNet Block 2 in the decoder reduces this to 256. Next, a SparseConv ResNet Block operates on this tensor, and subsequently a SparseConv Transpose layer halves the number of feature maps down to 128. This sequence, concatenate, SparseConv ResNet Block 2, SparseConv ResNet Block, and SparseConv Transpose, is then repeated until the prediction layer. Note that the last SparseConv Transpose layer does not halve the number of feature channels.

We assign a label to each point in a point cloud, i.e., we search for $\phi : F^N \rightarrow [clutter, road, building, vegetation, vehicles]^N$, where F is the points feature tensor and N is the number of points in the point cloud. Note that we differentiate between a point and how a point is represented, i.e., their features f. Consequently, we also need a tensor that keeps track of the coordinates, which allows us to monitor the positions in a three-dimensional space. In the first experiment, a point was represented by its geometric features $f_i = [x_i, y_i, z_i]$ and in the second experiment also by its color $f_i = [x_i, y_i, z_i, re_i, gr_i, bl_i]$. Mathematically, we can write these two tensors as two matrices: one matrix C for the coordinates and one matrix F for the features. When we use geometry and color as features, these matrices are as follows:

$$C = \begin{bmatrix} x_1 & y_1 & z_1 & b_1 \\ x_2 & y_2 & z_2 & b_1 \\ \vdots & & & \\ x_i & y_i & z_i & b_j \\ x_{i+1} & y_{i+1} & z_{i+1} & b_j \\ \vdots & & & \\ x_N & y_N & z_N & b_M \end{bmatrix} \quad F = \begin{bmatrix} x_1 & y_1 & z_1 & re_1 & gr_1 & bl_1 \\ x_2 & y_2 & z_2 & re_1 & gr_1 & bl_1 \\ \vdots & & & & & \\ x_i & y_i & z_i & re_i & gr_i & bl_i \\ x_{i+1} & y_{i+1} & z_{i+1} & re_{i+1} & gr_{i+1} & bl_{i+1} \\ \vdots & & & & & \\ x_N & y_N & z_N & re_N & gr_N & bl_N \end{bmatrix}$$

The b_i denote the batch index a point belongs to. All features were normalized, i.e., min-max scaled, or more formally $\forall f \in F, f \in [0, 1]$. We trained the network for 50 epochs with batch size 3. We optimized the parameters with RMSprop [146] and a learning rate of 0.001. Our hardware set up is 16 GB DIMM DDR 1333 MHz RAM, Intel Core i5-2400 with 3.1 GHz CPU, NVIDIA GTX 1080 Ti with 12 GB GPU, and as a motherboard Dell 0HY9JP version A00. With this configuration, one training loop takes approximately 3 h 45 min, and one evaluation loop approximately 45 min.

4. Results

We present our results in the following two subsections. Section 4.1 describes the results for dense image matching and 3D reconstruction. We then present (and compare) in Section 4.2 the segmentation results for the U-Net based GSCNNs, the decision tree, and for the FCN-8s (in this order).

We also describe the results for the GSCNN model that we trained for 400 epochs with class weights. Finally, we evaluate the overall performance of the models, as well as the class level performance.

4.1. Dense Image Matching and 3D Reconstruction Results

Image orientation and dense matching were performed for all three images covering the study area. The automatic tie point extraction identified approximately 552 points, with standard deviations between 0.38 and 0.52 pixels, while the total standard deviation of the bundle adjustment resulted in a value of 0.54 pixels. The dense image matching process was performed at full image resolution. Hence, for each (matched) pixel, a 3D object point was generated, whose coordinates were calculated by spatial forward intersections, finally resulting in a large photogrammetric point cloud with a high density of 4 points/m^2 (Appendix A Figure A1). Compared with stereo, the use of tri-stereo images results in a higher quality 3D reconstruction, since three-acquisition-direction reduces the risk of occlusions. Nevertheless, difficulties are encountered in regions with poor or repetitive texture, such as water surface or shaded areas, where the final 3D point cloud contains higher noise and outliers. The resulting point cloud describes the terrain surface in the open, free areas (agriculture and grasslands) and the top surface of the objects on it (buildings and trees). The described surface is smooth, with bevelled building edges and missing height information for small individual objects. For example, points corresponding to small vehicles (family cars with lengths smaller than 5 m and heights smaller than 1.5 m) are not elevated compared to their surroundings. Hence, a classification using solely geometry information cannot reliably detect such objects.

For the quality assessment, the point cloud was interpolated into a regular elevation model with 0.5 m resolution, i.e., DSM, by using a robust moving planes interpolation method. Therefore, the vertical Root Mean Square Errors (RMSEs) between the DSM and 50 homogeneously distributed CPs as well as the reference LiDAR DTM was computed (the comparison with the LiDAR DTM was only done in open, free areas, where the DSM can be considered the same with the DTM). Additionally, the distribution of the height difference errors was analyzed by computing statistic measures such as mean, standard deviation (σ), median, and a robust standard deviation based on the median of absolute differences (σ_{MAD}). The height differences between the computed DSM and the LiDAR DTM in open areas show a systematic positive difference (~1 m) in the southern part of the scene (Appendix A Figure A2). Such systematic error can be removed by applying Least Square Matching (LSM) [147] as demonstrated in Loghin et al. [148], where height differences are reduced to a median close to zero and the total Root Mean Square Error (RMSE) decreases from 0.96 m to 0.61 m (Appendix A Table A1). For the 50 CPs, the RMSE dropped from 0.31 to 0.25 m after LSM, which is very close to one-third of the pixel size (0.35 pixel).

For completeness, it should be mentioned that GCPs are not mandatory for processing satellite imagery. Based on the original RPCs and automatically measured tie points, a DSM can be derived as well. However, the geolocation accuracy of such model is poor, as in our case we observed a positive offset of approximately 18 m. Applying a LSM transformation as described before, similar accuracies can be achieved, as with the usage of GCPs.

4.2. Semantic Segmentation Results

In this section, we describe and compare the performance of the trained models. We start with the GSCNN models using class weights based on 50 epochs. Then, we proceed with the decision trees, and follow up with FCN-8s. Finally, we describe the GSCNN model that uses class weights over the 400 epochs. Before that, we give a brief rationale on the chosen performance metrics.

We compare the performance of the machine learning models with metrics that do not take the number of classes into account, and with metrics that do. We do this for many reasons, among them are the following three. The first is to get a more detailed picture on how a particular model actually performs. This is especially important when one or two classes dominate. Second, and tied to the first, performance metrics that do not take the number of classes into account put, a mostly unjustifiable, high importance on the dominant classes. In this paper, for example, adding color as features to the coordinates does improve the performance of the GSCNN model judged by the kappa score and the overall accuracy. However, it decreases the performance judged by metrics that take the number of classes into account, e.g., the average class accuracy. Third, in some cases, we are interested in the prediction performance of a particular class. An organization that is in the oil pipeline surveillance business is generally more interested in vehicles than in buildings. For these reasons, we compare the performance with four metrics that take the number of classes into account by averaging and with two metrics that do not. We also break the performance down to the class level for accuracy, precision, recall, and F1.

Adding color information to geometry does increase the kappa score and the overall accuracy of our U-Net based GSCNN (Table 3). In addition, the color information has increased the class metrics for clutter on accuracy, recall, and F1, while it has decreased the precision (Table 4). In fact, the increase in kappa score and overall accuracy is solely due to the increased performance on predicting clutter. However, this increase has a negative effect on all other metrics. In general, we see lower values for average precision, average recall, average F1, and average accuracy. The same is true for the per class metrics, with the exception of the aforementioned clutter class. The two dominant classes clutter and trees are overwhelmingly present in false positives and false negatives (Tables A3 and A5). Interestingly, adding color features does not have a homogeneous reducing effect on this. For clutter, the number of times tree is predicted increases from 3.6 million to 10.2 million. For trees, we observe the opposite effect as the number of times clutter is predicted, decreasing from 9.3 million to 1.4 million. Adding class weights to the color and coordinates model generally reverses the described effects, albeit not completely, with the roads class being the exception as it increases the performance. This means that, if a value decreases with the additional color features, it increases with the class weights, and vice a versa. In Table 3, for example, the weighted loss increases average precision, average recall, average F1, and average accuracy while it decreases the kappa score and overall accuracy. This effect is also present in Table 4, and when we compare Tables A5 and A7. These decreases can be attributed to the effects of the weighted loss, as the weights decrease the impact of classes with a high occurrence and increase the impact of classes with a low occurrence on the network's loss. Adding color to the coordinates leads to a higher stability over the training process (compare Figures 7 and 8), while median class weights seem to introduce instability again (compare Figures 9 and 10).

Table 3. Overall quantitative comparison of the GSCNN, FCN-8s, and the decision tree. We report (from left to right) average precision, average recall, average F1 score, average accuracy, kappa score, overall accuracy, and MCC. Values in bold format are the highest numbers for the corresponding metrics, with the exception for baseline A. See Table 4 for per class results, and Appendix B for the confusion matrices. Note that we abbreviate weighted loss with (W.L).

Models	**Avg. Precision** %	**Avg. Recall** %	**Avg. F1** %	**Avg. A** %	**Kappa** %	**OA** %	**MCC** $[-1, +1]$
baseline A	12.85	20.00	15.64	20.00	47.33	64.25	0.00
U-Net based GSCNN (3D)							
Coordinates	23.69	24.33	23.20	24.32	38.90	56.01	0.18
Coordinates, Colors	19.31	19.98	17.38	19.97	45.07	62.14	0.00
Coordinates, Colors, W.L	21.92	22.24	21.36	22.22	34.30	51.07	0.09
FCN-8s (2D)							
Colors	**62.43**	**61.15**	**59.12**	**61.15**	**90.76**	**96.11**	0.91
Decision Tree (3D)							
Coordinates	43.89	38.73	39.54	38.73	82.00	89.10	0.76
Coordinates, Colors	61.03	58.72	58.96	58.71	86.60	93.18	0.85

The additional color information boosts the decision tree's performance on all classes, but one, among all metrics (Table 4). Adding color information has no effect on the vehicle class. The per class accuracy improved from 93% to 95% for clutter, from 88% to 93% for trees , from 11% to 79% for buildings, and from 0% to 24% for roads. This 24% for roads is the best performance among the three approaches. While two of these improvements are dramatic, they only lead to a modest improvement for the kappa score and the overall accuracy, due to the class imbalance. However, the average per class accuracy improves by 20% from 38% to 58%, which is only 3% lower than the average per class accuracy of FCN-8s. The additional color information also leads to the best precision and F1 score for buildings and the best recall for roads. Hence, the decision tree outperforms FCN-8s on these cases, when simultaneously trained on both hand-crafted geometric and color features. When only trained on the hand-crafted geometric features, the decision tree suffers from the dominance in false positives and false negatives of the two classes clutter and trees as the GSCNN, although not as strongly as the GSCNN (Table A13). However, the additional color information partially mitigates this (Table A5).

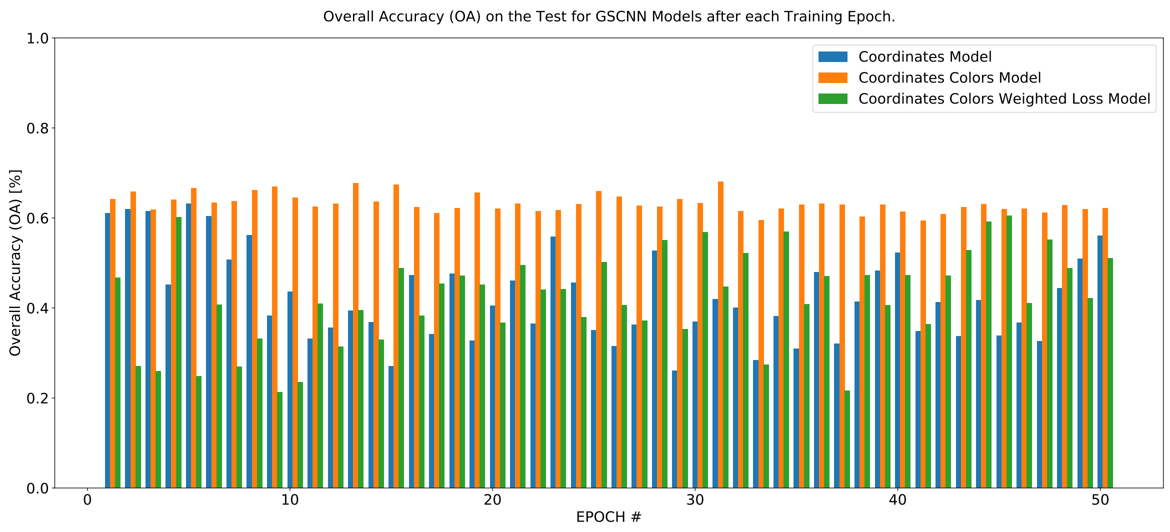

Figure 7. Overall accuracy for the GSCNN models after each epoch.

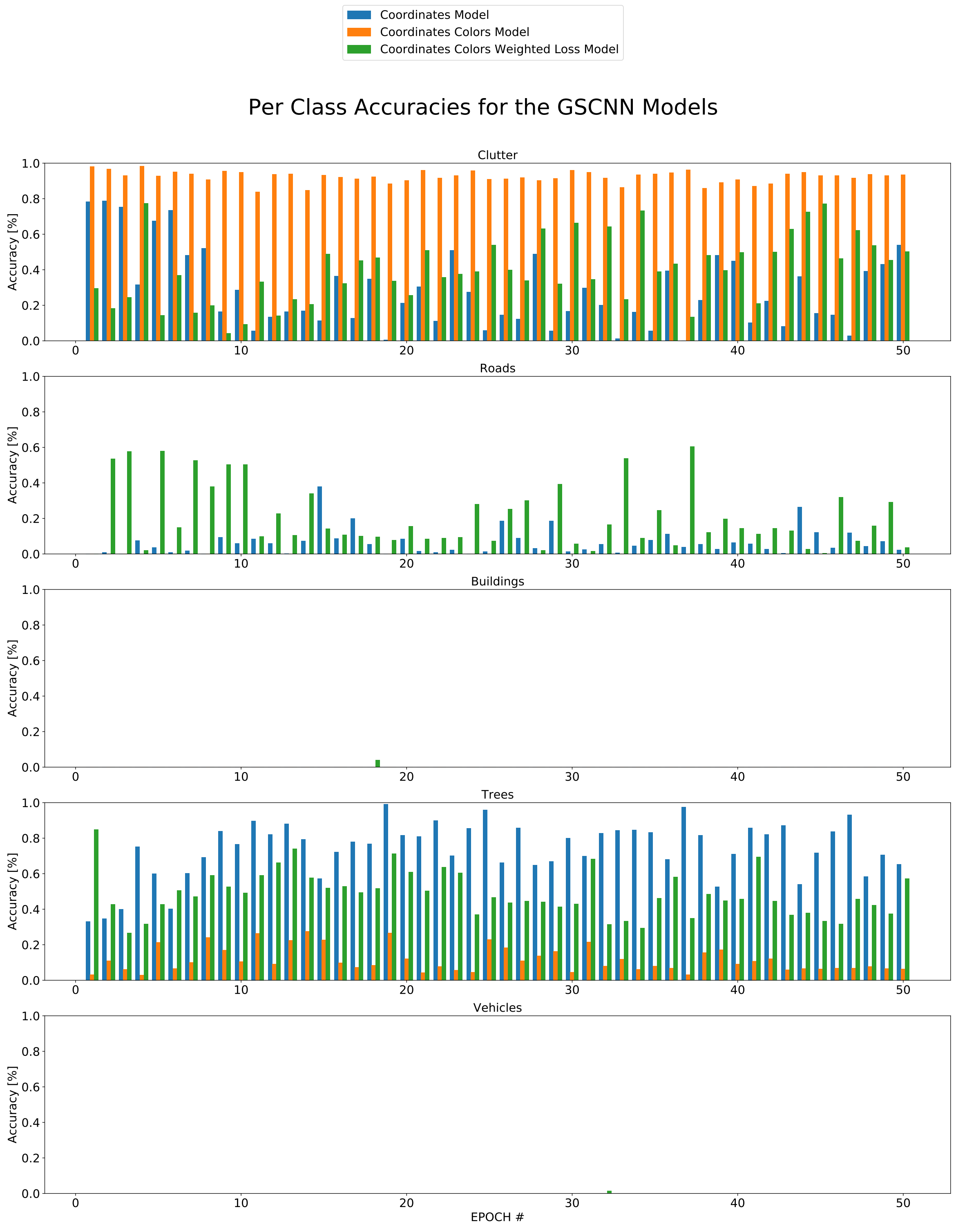

Figure 8. Per class accuracies for the GSCNN models after each epoch.

FCN-8s performs the best among the three approaches in our study area, with an average precision of 62%, an average recall of 61%, an average F1 score of 59%, a kappa score of 90%, an overall accuracy of 96%, and an average per class accuracy of 61% (Table 3). However, the performance of the decision tree when color is included is only 1–3% lower for these metrics. FCN-8s confusion matrix shows that it does not suffer as strongly as the other two from the class imbalance (Table A8). For example, for clutter, the second biggest group of false positives is roads and not trees. FCN-8s also leads in most of the per class metrics (Table 4); it is outperformed by the decision tree when it is trained on hand-crafted geometric features and color on four occasions. First, the decision tree has a higher accuracy for predicting roads. Second, the decision tree has a higher precision for predicting buildings. Third, the decision tree has a higher recall for predicting roads. Fourth, the decision tree has a higher F1 score for buildings.

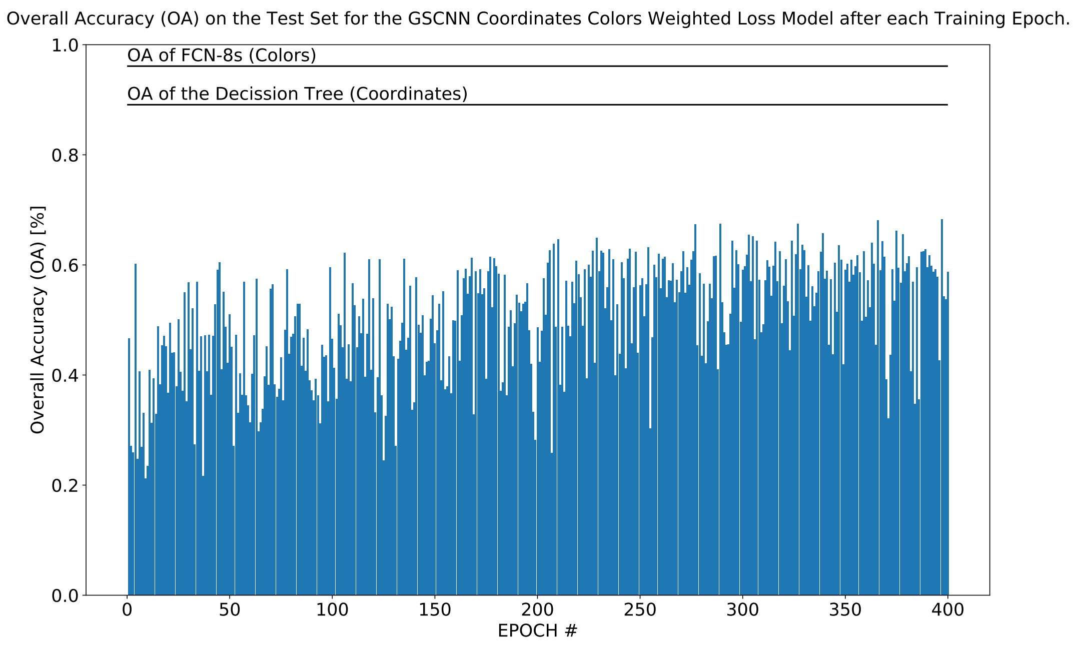

Figure 9. Overall accuracy for the GSCNN model that is trained on coordinates and colors, and uses a weighted loss, after each epoch.

Table 4. Per class metrics for the GSCNN, the FCN-8s, and the decision tree. Values in bold format are the highest numbers for corresponding metric, with the exception for baseline A. See Appendix B for the confusion matrices.

Model	Clutter %	Roads %	Buildings %	Trees %	Vehicles %
Per Class Accuracy					
baseline A	100.00	0.00	0.00	0.00	0.00
U-Net based GSCNN (3D)					
Coordinates	53.87	2.36	00.00	65.40	00.00
Coordinates, Color	93.43	00.00	00.00	6.45	00.00
Coordinates, Colors, W.L	50.22	3.61	00.00	57.34	00.00
FCN-8s (2D)					
Color	**98.19**	22.89	**86.44**	**98.22**	00.00
Decision Tree (3D)					
Coordinates	93.90	00.00	11.71	88.02	00.00
Coordinates, Colors	95.80	**24.75**	79.13	93.88	00.00
Per Class Precision					
baseline A	64.25	0.00	0.00	0.00	0.00

Table 4. *Cont.*

Model	Clutter %	Roads %	Buildings %	Trees %	Vehicles %
U-Net based GSCNN (3D)					
Coordinates	73.58	2.59	00.00	42.26	00.00
Coordinates, Color	64.21	00.00	00.00	32.36	00.00
Coordinates, Colors, W.L	69.03	3.15	00.00	37.41	00.00
FCN-8s (2D)					
Color	**96.09**	**63.85**	54.25	**97.95**	00.00
Decision Tree (3D)					
Coordinates	90.68	00.00	42.67	86.10	00.00
Coordinates, Colors	93.87	49.56	**67.76**	93.89	00.00
Per Class Recall					
baseline A	100.00	0.00	0.00	0.00	0.00
U-Net based GSCNN (3D)					
Coordinates	53.87	2.36	00.00	65.40	00.00
Coordinates, Color	93.44	00.00	00.00	6.46	00.00
Coordinates, Colors, W.L	50.22	3.62	00.00	57.35	00.00
FCN-8s (2D)					
Color	**98.02**	22.89	**86.45**	**98.22**	00.00
Decision Tree (3D)					
Coordinates	93.91	00.00	11.72	88.02	00.00
Coordinates, Colors	95.81	**24.76**	79.13	93.89	00.00
Per Class F1					
baseline A	78.24	0.00	0.00	0.00	0.00
U-Net based GSCNN (3D)					
Coordinates	62.20	2.47	00.00	51.34	00.00
Coordinates, Color	76.12	00.00	00.00	10.77	00.00
Coordinates, Colors, W.L	58.14	3.37	00.00	45.29	00.00
FCN-8s (2D)					
Color	**97.13**	**33.70**	66.66	**98.02**	00.00
Decision Tree (3D)					
Coordinates	92.27	00.00	18.39	87.76	00.00
Coordinates, Colors	94.83	33.02	**73.01**	93.93	00.00

Training the GSCNN on color and geometry, with a weighted loss, mitigates the class imbalance problem (Figure 10 and Table 4). The network commonly has accuracies different from 0 for buildings and vehicles between 100 and 250 epochs. However, after the 250th epoch, they become less and less frequent and smaller. The volatility of the overall accuracy, measured by the standard deviation σ, is five times higher than the coordinates model. In particular, $\sigma = 10.22\%$, which is similar to the coordinates model $\sigma = 9.99\%$, while, for the model that uses coordinates and color, we have $\sigma = 2.02\%$. MCC almost completely recovered after 400 training epochs to 0.17, which is almost twice as high than without colors. In other words, it almost completely recovered.

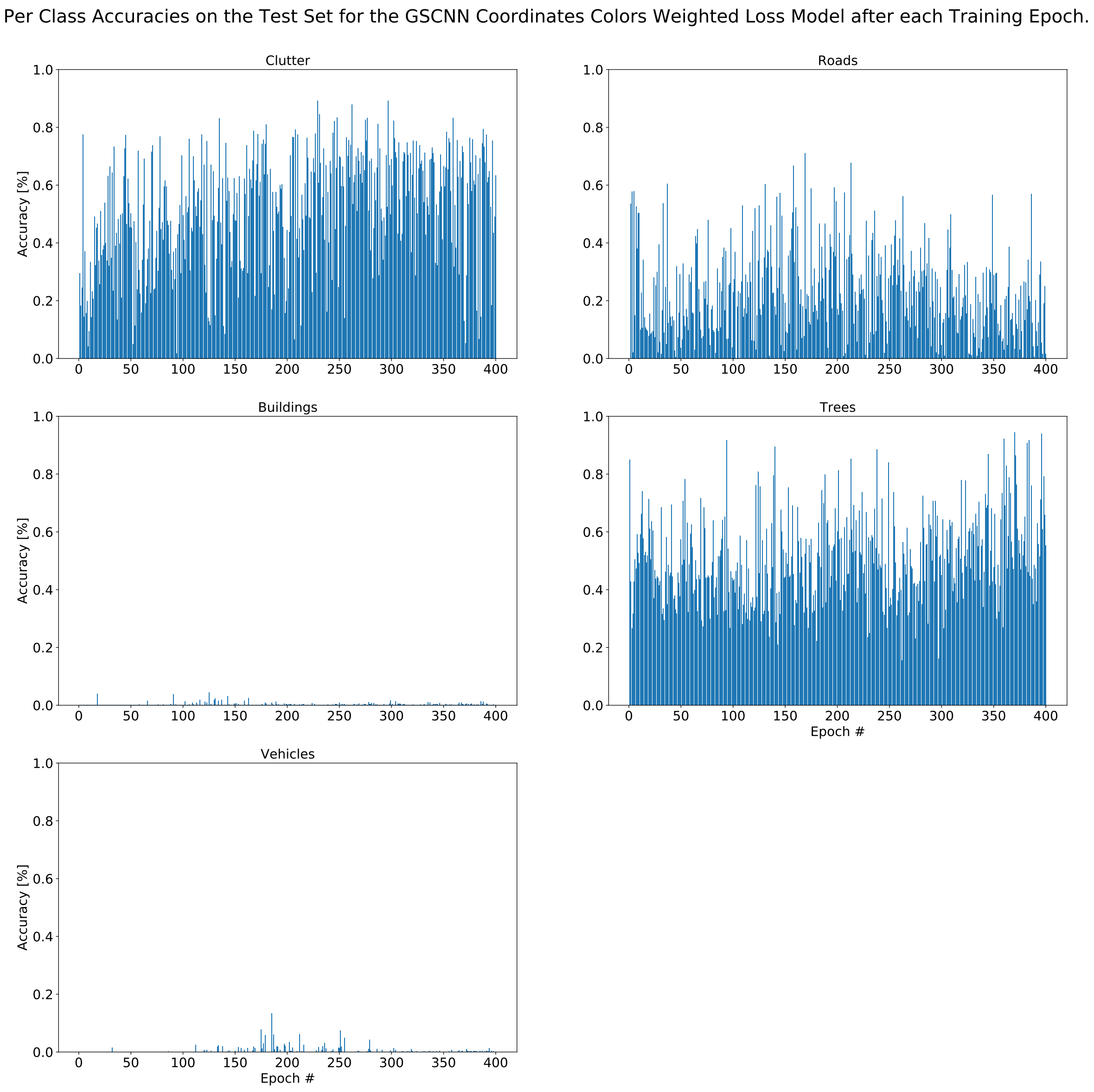

Figure 10. Per class accuracies for the GSCNN model that is trained on coordinates and colors, and uses a weighted loss, after each epoch.

5. Discussion

In this section, we analyze the 3D reconstruction result in the first paragraph, and interpret the semantic segmentation outcome in the context of CV in the second paragraph. Finally, we outline future research directions.

As described in Section 4.1, the accuracy of the computed DSM was analyzed based on 50 homogeneously distributed CPs. We observed a vertical RMSE of 0.25 m. Our result is therefore better than the accuracies of 0.49 m reported by Bernard et al. [149] and of 0.5 m obtained by Hu et al. [150] for 6001 ground LiDAR check points. We picked CPs on flat, smooth, and non-vegetated areas, whereas, at larger scale, the surface land cover type [151] will influence accuracy assessments. Hence, we compared the Pléiades DSM with the reference LiDAR DTM in the open, stable areas, only. However, we did not explicitly exclude low vegetation and agriculture fields, making the larger global RMSE of 0.61 m (0.9 GSD) plausible. Nevertheless, the result corresponds to reported vertical accuracies of 0.44 and 2 GSD for photogrammetrically derived elevation models captured from aerial platforms [151–153].

DL advanced a variety of tasks in NLP and CV, especially in cases where the problem can be formulated as a set of features that can be arranged in a grid structure, e.g., images. Recently, this assumption of a two-dimensional grid is challenged by the machine learning community, as more effort is put into generalizing the convolution operation, e.g., with the previously mentioned pointwise convolutional operation [123], and the generalized sparse convolution operation we use in this paper [91]. While the GSCNN is outperformed by both models, we argue that our results indicate that the generalized sparse convolution operation is a promising candidate as an approach, especially when we acknowledge the fact that learning geometric patterns in a three-dimensional space seems to be a harder learning problem, as indicated by our experiments, which is particularly true for derived point clouds from tri-stereo imagery as the geometric information quality of objects is lower. That leads to difficult decision boundaries, as these boundaries are diffused. Additionally, we have to acknowledge that learning in a higher-dimensional space, 3 versus 6, is per se more difficult due to the curse of dimensionality [154]. Our experiments also indicate that median class weighting improves the performance on rare classes, even with a factor of 1:10.000. However, it requires 2–3 times more epochs (and hence a longer training time) for these performance improvements to be visible in out setting. We want to highlight that data augmentation might drastically improve the performance of the tested model as this is a widely observed behavior in DL. Class imbalance strategies are also likely to greatly boost the performance of DL in our case, e.g., the approach presented by Messay-Kebede et al. greatly boosts the accuracy from 40.5% to 81.0% on class Simda, where only 42 samples are present [155].

Future research directions include the use of DL for 3D reconstruction. For example, Saito et al. presented a technique that takes one or more images as an input, and outputs a 3D reconstruction of an object [156]. The findings of this research direction might give valuable input for improvement of the currently used hand crafted mathematical models that we use in this paper. The applicability of superresolution on the derived point clouds is also worthy of future investigation, similar to what was done in the two-dimensional grid world with images by, e.g., Kim et al. [157]. Not only might this highlight areas for improvement of the mathematical models, but it might also lead to less distorted point clouds, which in turn can significantly improve not only the segmentation results but also other tasks, e.g., object detection. Another research direction is to conduct a comparison study, e.g., comparing our approach with pointwise convolutional networks [123] and pointnet based networks [60]. Investigating the uncertainty of the model is also an interesting and important avenue for further research [158]. Findings from this research help understand in which scenarios the model is likely to be wrong, which, in turn, would lead to strategies that limit and mitigate these scenarios. Information about uncertainty is important for deploying and incorporating machine learning models into business processes (see above). An ablation study on the network's components is also a direction. For example, systematically decreasing the number of filters gives insight to the question of model complexity. Finally, a study on different class imbalance strategies will contribute to answering the question which strategies will improve the performance in settings such as ours.

6. Conclusions

We studied the utility of point clouds that are derived from tri-stereo satellite imagery for semantic segmentation with a U-Net based architecture that uses generalized sparse convolutions. We used an Austrian study area. For this study area, we reported that, ceteris paribus, color and geometry (from derived point clouds) do not lead to an increase in general performance for the GSCNN. In our experiments, we found that, inter alia, adding color to geometry leads to a higher performance only on the dominant class. We also found that median class weighting partially reverts the effects of adding color. For example, MCC drops from 0.18 to 0.00 when color is added, and it raises to 0.09 with median class weighting in our experiments. The network also started to learn the classes with lower occurrences. In addition, FCN-8s and the decision tree outperform the GSCNN. Providing the GSCNN with the additional color information does increase the overall accuracy from 56% to 62%, and the kappa score from 39% to 45%, but this is at the expense of a significantly lower per class accuracy for the decision tree. The accuracy performance for roads drops from a modest 2.25% to 0.00%. This leads to a fall of the average per class accuracy from 24% to 19%. For the decision tree, adding color information to the geometry leads to a significantly higher performance. The kappa score increases from 82% to 86%, the overall accuracy from 89% to 93%, and the average per class accuracy from 38% to 58%. Adding color has a particularly strong effect on the class accuracy performance for roads and buildings. The highest performance on this study area has FCN-8s with a kappa score of 90%, an overall accuracy of 96%, and an average per call accuracy of 61%. This network is only outperformed by the decision tree as the latter has a 2% higher accuracy for roads prediction. While the GSCNN is outperformed by and large by the decision tree, it has major shortcomings. One is the longer prediction phase. With their respective hardware settings, the GSCNN takes 45 min for the test dataset, while computing the features for the decision tree takes approximately 4 h with our hardware set up. Additionally, one has to set up the infrastructure for this. We contribute in multiple ways with our research. Firstly, we open up a new possibility for data fusion in remote sensing, specifically in the case of applying machine learning [159]. To the best of our knowledge, this is the first work that studies point clouds derived from tri-stereo imagery for DL approaches. Second, similarly, our work extends the semantic segmentation literature to satellite derived point clouds. Third, we contribute to transfer learning, by making our class weights for the GSCNN networks open source on PyTorch Hub via https://github.com/MacOS/ReKlaSat-3D. Additionally, it makes our research more accessible to other researchers by drastically decreasing the amount of resources necessary to use our results. We also make the learned decision trees publicly available through this GitHub repository.

Author Contributions: Conceptualization, S.B., A.-M.L., J.O., N.P., and R.H.; Data curation, S.B., A.-M.L., J.O., and N.S. (Nikolaus Schiller); Formal analysis, S.B. and A.-M.L.; Funding acquisition, J.O., N.P., N.S. (Nikolaus Schiller), and R.H.; Investigation, S.B., A.-M.L., J.O., N.P., M.H., and O.K.; Methodology, S.B., J.O., N.P., and M.H.; Project administration, S.B., J.O., N.P., M.H., A.S., and R.H.; Resources, N.P., K.H., and N.S. (Nikolaus Schiller); Software, S.B., A.-M.L., and M.H.; Supervision, N.P., K.H., and R.H.; Validation, S.B. and A.-M.L.; Visualization, S.B. and A.-M.L.; Writing—original draft, S.B., A.-M.L., J.O., A.S., and N.S. (Niklas Schmidinger); and Writing—review and editing, S.B., A.-M.L., J.O., N.P., A.S., K.H., and R.H. All authors have read and agree to the published version of the manuscript.

Funding: This research was funded by the Austrian Research Promotion Agency (FFG) project 3D Reconstruction and Classification from Very High Resolution Satellite Imagery (ReKlaSat 3D) (grant agreement No. 859792).

Acknowledgments: The authors acknowledge Open Access Funding by TU Wien. We would like to thank Gustaf Neumann (Vienna University of Economics and Business) for supplying additional computational resources. Furthermore, we want to express our gratitude to the Austrian Institut für Militarisches Geowesen (Mag. Maximilian Göttlich and Clemens Strauß) for the support in satellite image acquisition.

Conflicts of Interest: Author Nikolaus Schiller was employed by the company Vermessung Schmid ZT GmbH. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DL	Deep Learning
NLP	Natural Language Processing
CV	Computer Vision
GSCNN	Generalized Sparse Convolutional Neural Network
CNN	Convolutional Neural Network
FCNN	Fully Convolutional Neural Network
RNN	Recurrent Neural Network
DRR Net	Dense Refinement Residual Network
BERT	Bidirectional Encoder Representations from Transformers
RoBERTa	A Robustly Optimized BERT Pretraining Approach
SVM	Support Vector Machine
VHR	Very High Resolution
NIR	Near-Infrared

LiDAR	Light Detection And Ranging
DTM	Digital Terrain Model
DSM	Digital Surface Model
RPCs	Rational Polynomial Coefficients
GCPs	Ground Control Points
CPs	Check Points
GSD	Ground Sample Distance
LSM	Least Squares Matching
FBM	Feature Based Matching
CBM	Cost Based Matching
OPALS	Orientation and Processing of Airborne Laser Scanning
RMSE	Root Mean Square Errors

Appendix A

Figure A1 illustrates the results of image matching, with an overview of the reconstructed point cloud together with color and shaded DSM visualizations for the study area. Figure A2 shows the color-coded elevation differences between the photogrammetric-derived DSM and the reference LiDAR DTM with their frequency distribution histogram before and after applying LSM transformation.

Table A1. Vertical accuracy assessment of Pléiades DSM w.r.t. CPs and the entire scene in open areas (values are in meters).

	Before LSM				After LSM			
	Mean	**Std**	σ_{MAD}	**RMSE**	**Mean**	**Std**	σ_{MAD}	**RMSE**
CPs	0.25	0.30	0.29	0.31	0.12	0.23	0.21	0.25
DSM (open areas)	0.80	0.53	0.51	0.96	0.15	0.60	0.50	0.61

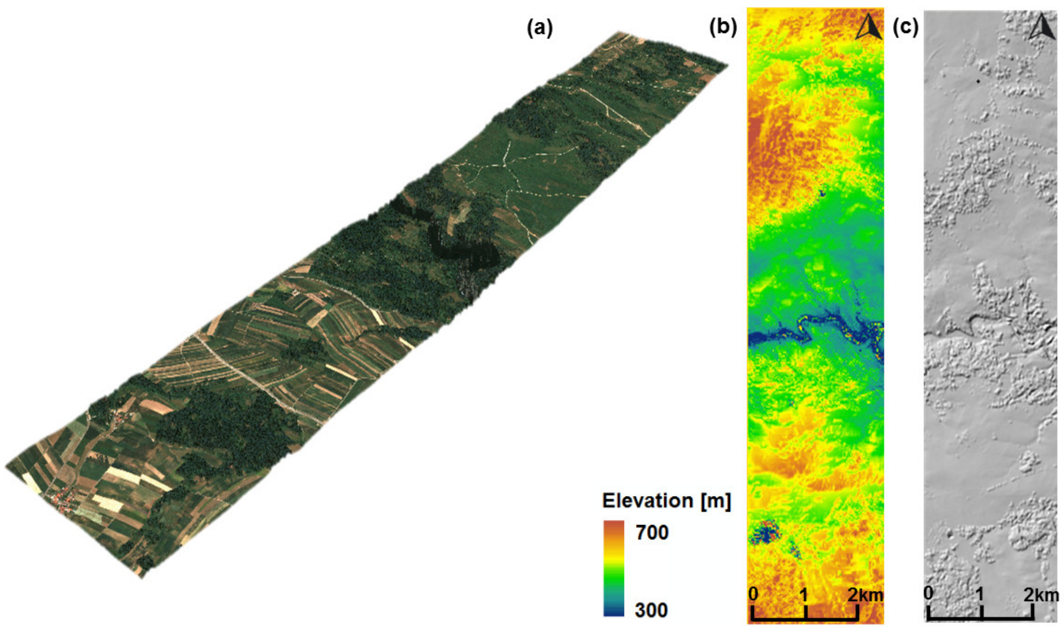

Figure A1. Image matching and 3D reconstruction results for the study area selected for the semantic segmentation: (**a**) 3D point cloud; (**b**) color coded view of the DSM; and (**c**) shaded view of the reconstructed DSM.

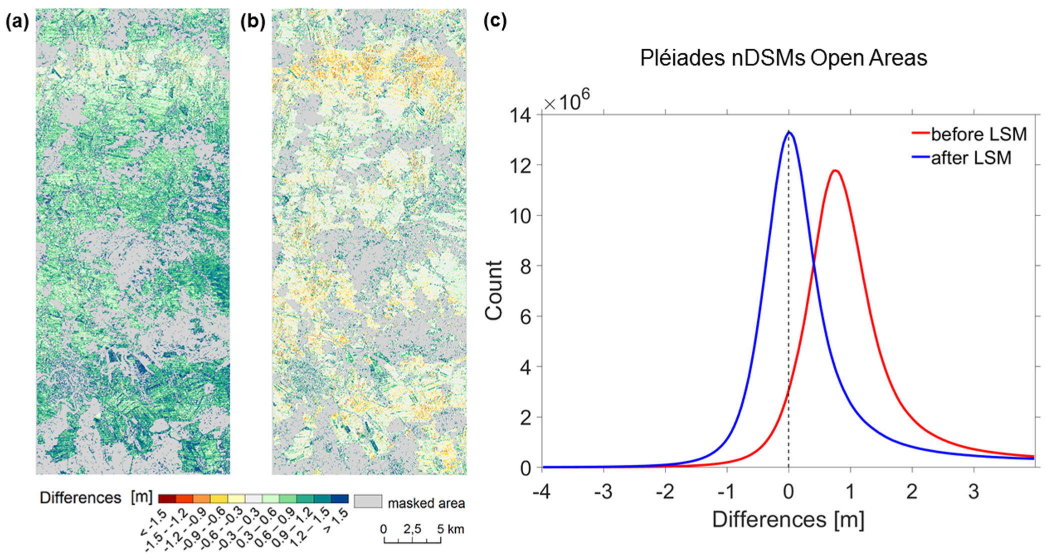

Figure A2. Pléiades nDSM statistics for open areas (masked areas are shown in grey): (**a**) spatial distribution of color-coded height differences before LSM; (**b**) spatial distribution of color-coded height differences after LSM; and (**c**) the frequency distribution histogram of Pléiades nDSM heights.

Appendix B

In this appendix section we depict the confusion matrices for all machine learning methods and each set of input features. Tables A3 and A5 show the confusion matrices for the GSCNN for the input features coordinates, and the input features coordinates and colors, respectively. Table A8 shows the confusion matrix for the two-dimensional CNN approach. Finally, the confusion matrices for the decision tree trained on coordinates, and trained on coordinates and colors are shown in Tables A12 and A13, respectively. Please note that the numbers in Tables A3 and A5 do not add up the numbers presented in Table 2, because the current implementation of the Minkowski Engine cannot process point clouds with duplicate points. For example, it does not allow that $f_1 = [2, 2, 2]$ and $f_2 = [2, 2, 2]$ are in the same point cloud pl, i.e., if $f_1 \in pl \implies f_2 \notin pl$ and if $f_2 \in pl \implies f_1 \notin pl$. We had to delete 859 points for this reason. The numbers in Table A8 do not add up to Table 2, since the numbers reflect pixels and not points. The normalized confusion matrices are normalized along the rows, i.e., along the actual (also true) classes.

Table A2. Confusion matrix for the GSCNN model that is trained on coordinates, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	11,588,872	614,421	0	9,309,164	0
	Roads	463,755	21,421	0	422,481	0
	Buildings	106,155	279	0	29,155	0
	Trees	3,589,286	190,509	0	7,144,498	0
	Vehicles	1060	52	0	582	0

Table A3. Normalized confusion matrix for the GSCNN model that is trained on coordinates, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	53.87	2.86	0	43.27	0
	Roads	51.09	2.36	0	46.55	0
	Buildings	78.29	0.21	0	21.50	0
	Trees	32.86	1.74	0	65.40	0
	Vehicles	62.57	3.07	0	34.36	0

Table A4. Confusion matrix for the GSCNN model that is trained on coordinates and colors, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	20,100,179	0	0	1,412,278	0
	Roads	855,821	0	0	51,836	0
	Buildings	125,225	0	0	10,364	0
	Trees	10,218,858	0	0	705,435	0
	Vehicles	1623	0	0	71	0

Table A5. Normalized confusion matrix for the GSCNN model that is trained on coordinates and colors, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	93.44	0	0	6.56	0
	Roads	94.29	0	0	5.71	0
	Buildings	92.36	0	0	7.64	0
	Trees	93.54	0	0	6.46	0
	Vehicles	95.81	0	0	4.19	0

Table A6. Confusion matrix for the GSCNN model that is trained on coordinates and colors, and uses class weights, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	10,804,589	417,105	61,491	4,367,882	869
	Roads	727,020	32,852	10,115	271,748	48
	Buildings	22,370	1203	0	19,664	0
	Trees	9,958,478	456,497	63,983	6,264,999	777
	Vehicles	0	0	0	0	0

Table A7. Normalized confusion matrix for the GSCNN model that is trained on coordinates and colors, and uses class weights, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	50.22	3.38	0.10	46.29	0.00
	Roads	45.95	3.62	0.13	50.29	0.00
	Buildings	45.35	7.46	0.00	47.19	0.00
	Trees	39.98	2.49	0.18	57.35	0.00
	Vehicles	51.30	2.83	0.00	45.87	0.00

Table A8. Confusion matrix for the CNN model that is trained on colors, after 50 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	21,326,657	110,375	80,506	200,719	0
	Roads	665,108	209,678	16,834	24,317	0
	Buildings	15,848	712	118,464	2012	0
	Trees	186,250	7460	2288	10,824,717	0
	Vehicles	1243	160	278	13	0

Table A9. Normalized confusion matrix for the CNN model that is trained on color.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	98.20	0.51	0.37	0.92	0
	Roads	72.62	22.89	1.84	2.65	0
	Buildings	11.56	0.52	86.45	1.47	0
	Trees	1.69	0.07	0.02	98.22	0
	Vehicles	73.38	9.45	16.41	0.77	0

Table A10. Confusion matrix for the decision tree model that is trained on hand-crafted coordinates features.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	20,201,720	0	4883	1,306,322	0
	Roads	775,555	0	587	131,538	0
	Buildings	6049	0	15,887	113,654	0
	Trees	1,292,805	0	15,870	9,615,985	0
	Vehicles	1181	0	2	511	0

Table A11. Normalized confusion matrix for the decision tree model that is trained on hand-crafted coordinates features.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	93.91	0	0.02	6.07	0
	Roads	85.44	0	0.06	14.49	0
	Buildings	4.46	0	11.72	83.82	0
	Trees	11.83	0	0.15	88.02	0
	Vehicles	69.72	0	0.12	30.17	0

Table A12. Confusion matrix for the decision tree model that is trained on hand-crafted coordinates features and color features.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	20,610,793	223,248	38,686	640,198	0
	Roads	657,941	224,697	9981	15,061	0
	Buildings	25,486	1027	107,293	1784	0
	Trees	661,638	3867	2274	10,256,881	0
	Vehicles	1049	522	109	14	0

Table A13. Normalized confusion matrix for the decision tree model that is trained on hand-crafted coordinates features and color.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	95.81	1.04	0.18	2.98	0
	Roads	72.49	24.76	1.10	1.66	0
	Buildings	18.80	0.76	79.13	1.32	0
	Trees	6.06	0.04	0.02	93.89	0
	Vehicles	61.92	30.81	6.43	0.83	0

Table A14. Confusion matrix for the GSCNN model that is trained on coordinates and colors, and uses class weights, after 400 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	13,627,522	615,499	106,489	4,793,811	1151
	Roads	189,498	14,642	2471	23,374	232
	Buildings	13,721	572	213	17,642	0
	Trees	7,660,978	276,146	26,308	6,045,457	310
	Vehicles	20,738	798	108	44,009	1

Table A15. Normalized confusion matrix for the GSCNN model that is trained on coordinates and colors, and uses class weights, after 400 epochs.

				Predicted		
		Clutter	**Roads**	**Buildings**	**Trees**	**Vehicles**
Actual	**Clutter**	63.35	0.88	0.06	35.61	0.10
	Roads	67.81	1.61	0.06	30.42	0.09
	Buildings	78.54	1.82	0.16	19.40	0.08
	Trees	43.88	0.21	0.16	55.34	0.40
	Vehicles	67.95	13.70	0.00	18.30	0.06

Appendix C

As a pre-processing step, additional geometric features were computed for each 3D point in the matched cloud. To this point, we considered for the spatial query of a point neighborhood an infinite cylinder with 7 m radius. These features include, e.g., the normal vector (NormalX, NormalY, NormalZ, NormalSigma), EchoRatio, ZRank, ZRange, and features derived from the structure tensor T (linearity, planarity, and omnivariance). All these features describe the point distribution [144] and are required for the separability of the classes. Surface normals are an important geometric property for the description of a surface. The local tangent plane is estimated by computing the best fitting plane for the nearest points and its corresponding normal vectors (termed NormalX, NormalY, and NormalZ) together with the standard deviation of the fit are used as additional descriptions of the points (NormalSigma). The distribution of the points in the neighborhood is derived from the structure tensor T, from which three main features are extracted: linearity, planarity, and omnivariance. The linearity feature is used to characterize 3D line objects such as power lines, planarity is a feature describing the smoothness of the surface, and omnivariance gives information about the volumetric point distribution. EchoRatio is a measure that describes the vertical point distribution, ZRange represents the maximum height difference between the points in the neighborhood, while ZRank is the rank of the point corresponding to its height in the neighborhood [160].

Appendix D

We describe FCN-8s as introduced by Long et al. in greater detail in this appendix section (Figure A3). Neural network layers are depicted in rectangles, the rectangles are colored if we have collapsed multiple layers into one to increase readability. Numbers outside of rectangles indicate the number of feature maps that go in and out of a layer. For example, the first layer expects three input channels and outputs 64 channels. Crop takes two tensors as an input, and crops the larger one of the two down to the size of the smaller one. The circled plus represents element-wise tensor addition. The down path is described in the next paragraph, and the up-path in the last paragraph of this appendix section.

A major point of reference for this architecture are the five max pooling layers with a kernel size of 2×2, a stride of 2, and no padding (colored red). These max pooling layers successively downsample the feature maps. The first two max pool layers are preceded by two convolutional layers each, while the last three are preceded by three convolutional layers. A convolutional layer consists of a convolution with a kernel of size 3 × 3, a stride of 1, and a padding of 1, followed by the activation ReLU function (colored green). The last max pool layer is continued by a convolutional layer with a kernel of size 7 × 7, a stride of 1, a padding of 1, and outputs 4096 feature channels. The output of this layer is then put through the ReLU activation function, dropout is then applied with a rate of 0.5 [161]. This is followed by a convolutional layer with a kernel of size 1 × 1, with a stride of 1, and a padding of 0, on which ReLU and 0.5 dropout rate is applied (colored blue). The down path ends with a convolution layer with a kernel of size 1 × 1, with a stride of 1 and a padding of 0, that outputs 21 features channels.

The 21 output channels of the last layer are upsampled by a deconvolutional layer with a kernel of size 4 × 4, a stride of 2, and a padding of 0. This is then cropped to fit the output size of the fourth max pool layer, before element-wise tensor addition is used to merge the two tensors. After that, a deconvolutional layer with the same hyperparameters as before is used to upsample the tensor again. The same operations are performed to merge this resulting tensor with the output of the third max pooling layer. Finally, this tensor is then upsampled by a deconvolutional layer with kernel size 16 × 16, stride 8, and padding 0. This final tensor is then cropped to fit the size of the input tensor.

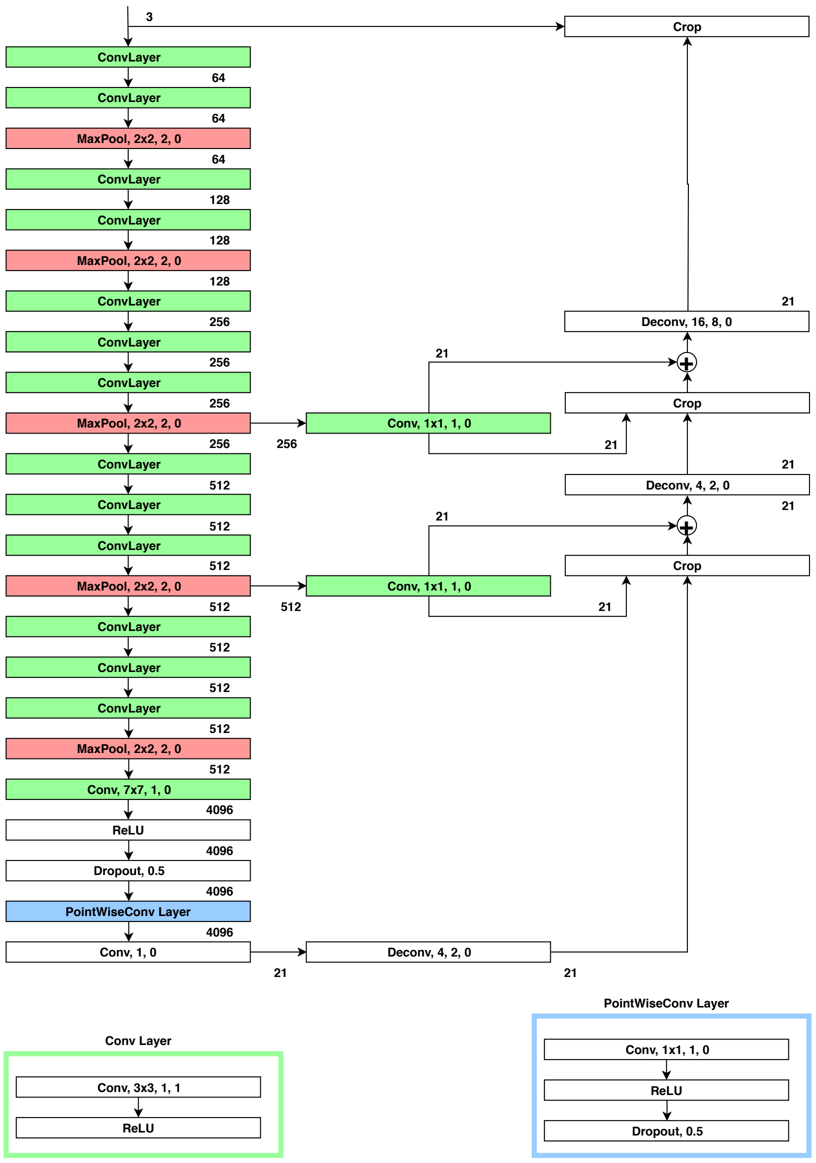

Figure A3. FCN-8s as introduced by Long et al.

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef]
2. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef]
3. Jordan, M.I.; Mitchell, T.M. Machine learning: Trends, perspectives, and prospects. *Science* **2015**, *349*, 255–260. [CrossRef]
4. Hirschberg, J.; Manning, C.D. Advances in natural language processing. *Science* **2015**, *349*, 261–266. [CrossRef]
5. Bengio, Y.; Courville, A.; Vincent, P. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [CrossRef] [PubMed]
6. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
7. Bengio, Y. Deep learning of representations for unsupervised and transfer learning. *Proc. ICML Workshop Unsupervised Transf. Learn.* **2012**, *27*, 17–36.
8. Tshitoyan, V.; Dagdelen, J.; Weston, L.; Dunn, A.; Rong, Z.; Kononova, O.; Persson, K.A.; Ceder, G.; Jain, A. Unsupervised word embeddings capture latent knowledge from materials science literature. *Nature* **2019**, *571*, 95. [CrossRef] [PubMed]
9. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
10. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
11. Howard, J.; Ruder, S. Universal Language Model Fine-tuning for Text Classification. *arXiv* **2018**, pp. 328–339. arXiv:1801.06146.
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*; Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q., Eds.; Curran Associates, Inc.: New York, NY, USA, 2012; pp. 1097–1105.
13. Huang, Y.; Cheng, Y.; Chen, D.; Lee, H.; Ngiam, J.; Le, Q.V.; Chen, Z. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *arXiv* **2018**, arXiv:1811.06965.
14. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788.
15. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
16. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
17. Sauer, A.; Aljalbout, E.; Haddadin, S. Tracking Holistic Object Representations. *arXiv* **2019**, arXiv:1907.12920.
18. Moon, G.; Chang, J.; Lee, K.M. Camera Distance-aware Top-down Approach for 3D Multi-person Pose Estimation from a Single RGB Image. In Proceedings of the IEEE Conference on International Conference on Computer Vision (ICCV), Long Beach, CA, USA, 16–20 June 2019.
19. Kocabas, M.; Karagoz, S.; Akbas, E. Multiposenet: Fast multi-person pose estimation using pose residual network. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 417–433.
20. Lugaresi, C.; Tang, J.; Nash, H.; McClanahan, C.; Uboweja, E.; Hays, M.; Zhang, F.; Chang, C.L.; Yong, M.G.; Lee, J.; et al. MediaPipe: A Framework for Building Perception Pipelines. *arXiv* **2019**, arXiv:1906.08172.
21. Dapogny, A.; Bailly, K.; Cord, M. DeCaFA: Deep Convolutional Cascade for Face Alignment in the Wild. *arXiv* **2019**, arXiv:1904.02549.
22. Wang, Z.; Chen, J.; Hoi, S.C. Deep learning for image super-resolution: A survey. *arXiv* **2019**, arXiv:1902.06068.
23. Liu, C.; Chen, L.C.; Schroff, F.; Adam, H.; Hua, W.; Yuille, A.L.; Fei-Fei, L. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–21 June 2019; pp. 82–92.
24. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 834–848.
25. Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349.

26. Badrinarayanan, V.; Kendall, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495.
27. Vu, T.H.; Jain, H.; Bucher, M.; Cord, M.; Pérez, P. DADA: Depth-aware Domain Adaptation in Semantic Segmentation. In Proceedings of the IEEE International Conference on Computer Vision, Soul, Korea, 27 October–2 November 2019.
28. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
29. Schmidt, J.; Marques, M.R.; Botti, S.; Marques, M.A. Recent advances and applications of machine learning in solid-state materials science. *npj Comput. Mater.* **2019**, *5*, 1–36.
30. Butler, K.T.; Davies, D.W.; Cartwright, H.; Isayev, O.; Walsh, A. Machine learning for molecular and materials science. *Nature* **2018**, *559*, 547.
31. Wimmers, A.; Velden, C.; Cossuth, J.H. Using Deep Learning to Estimate Tropical Cyclone Intensity from Satellite Passive Microwave Imagery. *Mon. Weather Rev.* **2019**, *147*, 2261–2282.
32. Topol, E.J. High-performance medicine: The convergence of human and artificial intelligence. *Nat. Med.* **2019**, *25*, 44.
33. Esteva, A.; Robicquet, A.; Ramsundar, B.; Kuleshov, V.; DePristo, M.; Chou, K.; Cui, C.; Corrado, G.; Thrun, S.; Dean, J. A guide to deep learning in healthcare. *Nat. Med.* **2019**, *25*, 24.
34. Chen, P.; Gadepalli, K.; MacDonald, R.; Liu, Y.; Kadowaki, S.; Nagpal, K.; Kohlberger, T.; Dean, J.; Corrado, G.S.; Hipp, J.D.; et al. An augmented reality microscope with real-time artificial intelligence integration for cancer diagnosis. *Nat. Med.* **2019**, *25*, 1453–1457.
35. Esteva, A.; Kuprel, B.; Novoa, R.A.; Ko, J.; Swetter, S.M.; Blau, H.M.; Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **2017**, *542*, 115.
36. Porumb, M.; Iadanza, E.; Massaro, S.; Pecchia, L. A convolutional neural network approach to detect congestive heart failure. *Biomed. Signal Process. Control* **2020**, *55*, 101597.
37. Avati, A.; Jung, K.; Harman, S.; Downing, L.; Ng, A.; Shah, N.H. Improving palliative care with deep learning. *BMC Med. Inform. Decis. Mak.* **2018**, *18*, 122.
38. Siddiquee, M.M.R.; Zhou, Z.; Tajbakhsh, N.; Feng, R.; Gotway, M.B.; Bengio, Y.; Liang, J. Learning Fixed Points in Generative Adversarial Networks: From Image-to-Image Translation to Disease Detection and Localization. In Proceedings of the IEEE International Conference on Computer Vision, Soul, Korea, 27 October–2 November 2019.
39. Wiens, J.; Saria, S.; Sendak, M.; Ghassemi, M.; Liu, V.X.; Doshi-Velez, F.; Jung, K.; Heller, K.; Kale, D.; Saeed, M.; et al. Do no harm: A roadmap for responsible machine learning for health care. *Nat. Med.* **2019**, *25*, 1337–1340.
40. DeVries, P.M.; Viégas, F.; Wattenberg, M.; Meade, B.J. Deep learning of aftershock patterns following large earthquakes. *Nature* **2018**, *560*, 632.
41. Kong, Q.; Trugman, D.T.; Ross, Z.E.; Bianco, M.J.; Meade, B.J.; Gerstoft, P. Machine learning in seismology: Turning data into insights. *Seismol. Res. Lett.* **2018**, *90*, 3–14.
42. Ross, Z.E.; Meier, M.A.; Hauksson, E.; Heaton, T.H. Generalized seismic phase detection with deep learning. *Bull. Seismol. Soc. Am.* **2018**, *108*, 2894–2901.
43. Senior, A.; Evans, R.; Jumper, J.; Kirkpatrick, J.; Sifre, L.; Green, T.; Qin, C.; Zidek, A.; Nelson, A.; Bridgland, A.; et al. Improved protein structure prediction using potentials from deep learning. *Nature* **2020**, *577*, 706–710.
44. Moen, E.; Bannon, D.; Kudo, T.; Graf, W.; Covert, M.; Van Valen, D. Deep learning for cellular image analysis. *Nat. Methods* **2019**, *16*, 1233–1246.
45. Goh, G.B.; Hodas, N.O.; Vishnu, A. Deep learning for computational chemistry. *J. Comput. Chem.* **2017**, *38*, 1291–1307.
46. Baldi, P.; Sadowski, P.; Whiteson, D. Searching for exotic particles in high-energy physics with deep learning. *Nat. Commun.* **2014**, *5*, 4308.
47. Tacchino, F.; Macchiavello, C.; Gerace, D.; Bajoni, D. An artificial neuron implemented on an actual quantum processor. *npj Quantum Inf.* **2019**, *5*, 26.
48. Xie, Y.; Franz, E.; Chu, M.; Thuerey, N. tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow. *ACM Trans. Graph. (TOG)* **2018**, *37*, 95.

49. Mrowca, D.; Zhuang, C.; Wang, E.; Haber, N.; Fei-Fei, L.F.; Tenenbaum, J.; Yamins, D.L. Flexible neural representation for physics prediction. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 8799–8810.
50. Lutter, M.; Ritter, C.; Peters, J. International Conference on Learning Representations Learning. *arXiv* **2019**, arXiv:1907.04490.
51. Goy, A.; Arthur, K.; Li, S.; Barbastathis, G. Low photon count phase retrieval using deep learning. *Phys. Rev. Lett.* **2018**, *121*, 243902.
52. Guest, D.; Cranmer, K.; Whiteson, D. Deep learning and its application to LHC physics. *Annu. Rev. Nucl. Part. Sci.* **2018**, *68*, 161–181.
53. Kutz, J.N. Deep learning in fluid dynamics. *J. Fluid Mech.* **2017**, *814*, 1–4.
54. Ferreira, B.Q.; Baía, L.; Faria, J.; Sousa, R.G. A Unified Model with Structured Output for Fashion Images Classification. *arXiv* **2018**, arXiv:1907.04490.
55. Hsiao, W.L.; Grauman, K. Learning the Latent "Look": Unsupervised Discovery of a Style-Coherent Embedding from Fashion Images. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
56. Liu, Z.; Luo, P.; Qiu, S.; Wang, X.; Tang, X. DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1096–1104.
57. Ma, L.; Liu, Y.; Zhang, X.; Ye, Y.; Yin, G.; Johnson, B.A. Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS J. Photogramm. Remote. Sens.* **2019**, *152*, 166–177.
58. Zhu, X.X.; Tuia, D.; Mou, L.; Xia, G.S.; Zhang, L.; Xu, F.; Fraundorfer, F. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geosci. Remote Sens. Mag.* **2017**, *5*, 8–36.
59. Zhang, L.; Zhang, L.; Du, B. Deep learning for remote sensing data: A technical tutorial on the state of the art. *IEEE Geosci. Remote Sens. Mag.* **2016**, *4*, 22–40.
60. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
61. Graham, B.; Engelcke, M.; van der Maaten, L. 3D Semantic Segmentation With Submanifold Sparse Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018.
62. Winiwarter, L.; Mandlburger, G.; Schmohl, S.; Pfeifer, N. Classification of ALS Point Clouds Using End-to-End Deep Learning. *PFG J. Photogramm. Remote Sens. Geoinf. Sci.* **2019**, *87*, 75–90. [CrossRef]
63. Stubbings, P.; Peskett, J.; Rowe, F.; Arribas-Bel, D. A Hierarchical Urban Forest Index Using Street-Level Imagery and Deep Learning. *Remote Sens.* **2019**, *11*, 1395.
64. He, J.; Li, X.; Yao, Y.; Hong, Y.; Jinbao, Z. Mining transition rules of cellular automata for simulating urban expansion by using the deep learning techniques. *Int. J. Geogr. Inf. Sci.* **2018**, *32*, 2076–2097.
65. Madu, C.N.; Kuei, C.h.; Lee, P. Urban sustainability management: A deep learning perspective. *Sustain. Cities Soc.* **2017**, *30*, 1–17.
66. Zhou, K.; Ming, D.; Lv, X.; Fang, J. CNN-based Land Cover Classification Combining Stratified Segmentation and Fusion of Point Cloud and Very High-Spatial Resolution Remote Sensing Image Data. *Remote Sens.* **2019**, *11*, 2065.
67. Dyson, J.; Mancini, A.; Frontoni, E.; Zingaretti, P. Deep Learning for Soil and Crop Segmentation from Remotely Sensed Data. *Remote Sen.* **2019**, *11*, 1859.
68. Kussul, N.; Lavreniuk, M.; Skakun, S.; Shelestov, A. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 778–782.
69. Kamilaris, A.; Prenafeta-Boldú, F.X. Deep learning in agriculture: A survey. *Comput. Electron. Agric.* **2018**, *147*, 70–90.
70. Weiss, M.; Jacob, F.; Duveiller, G. Remote sensing for agricultural applications: A meta-review. *Remote Sens. Environ.* **2020**, *236*, 111402.
71. Lambers, K.; Verschoof-van der Vaart, W.B.; Bourgeois, Q.P. Integrating Remote Sensing, Machine Learning, and Citizen Science in Dutch Archaeological Prospection. *Remote Sens.* **2019**, *11*, 794.
72. Ajami, A.; Kuffer, M.; Persello, C.; Pfeffer, K. Identifying a Slums' Degree of Deprivation from VHR Images Using Convolutional Neural Networks. *Remote Sens.* **2019**, *11*, 1282.

73. Wurm, M.; Stark, T.; Zhu, X.X.; Weigand, M.; Taubenböck, H. Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks. *ISPRS J. Photogramm. Remote Sens.* **2019**, *150*, 59–69.
74. Jean, N.; Burke, M.; Xie, M.; Davis, W.M.; Lobell, D.B.; Ermon, S. Combining satellite imagery and machine learning to predict poverty. *Science* **2016**, *353*, 790–794.
75. Grinias, I.; Panagiotakis, C.; Tziritas, G. MRF-based segmentation and unsupervised classification for building and road detection in peri-urban areas of high-resolution satellite images. *ISPRS J. Photogramm. Remote Sens.* **2016**, *122*, 145–166.
76. Montoya-Zegarra, J.A.; Wegner, J.D.; Ladickỳ, L.; Schindler, K. Semantic segmentation of aerial images in urban areas with class-specific higher-order cliques. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2015**, *2*, 127.
77. Dumas, M.; La Rosa, M.; Mendling, J.; Reijers, H.A. *Fundamentals of Business Process Management*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2018.
78. Cabanillas, C.; Di Ciccio, C.; Mendling, J.; Baumgrass, A. Predictive Task Monitoring for Business Processes. In *International Conference on Business Process Management*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 424–432.
79. Satyal, S.; Weber, I.; young Paik, H.; Ciccio, C.D.; Mendling, J. Business process improvement with the AB-BPM methodology. *Inf. Syst.* **2019**, *84*, 283–298.
80. Weidlich, M.; Polyvyanyy, A.; Desai, N.; Mendling, J.; Weske, M. Process compliance analysis based on behavioural profiles. *Inf. Syst.* **2011**, *36*, 1009–1025.
81. Mannhardt, F.; de Leoni, M.; Reijers, H.A.; van der Aalst, W.M.P. Balanced multi-perspective checking of process conformance. *Computing* **2016**, *98*, 407–437.
82. Rozinat, A.; Van der Aalst, W.M. Conformance testing: Measuring the fit and appropriateness of event logs and process models. In *BPM*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 163–176.
83. Van Der Aalst, W.; Adriansyah, A.; De Medeiros, A.K.A.; Arcieri, F.; Baier, T.; Blickle, T.; Bose, J.C.; Van Den Brand, P.; Brandtjen, R.; Buijs, J.; et al. Process mining manifesto. In *BPM*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 169–194.
84. Klinkmüller, C.; Ponomarev, A.; Tran, A.B.; Weber, I.; van der Aalst, W. Mining Blockchain Processes: Extracting Process Mining Data from Blockchain Applications. In *International Conference on Business Process Management*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 71–86.
85. Mühlberger, R.; Bachhofner, S.; Di Ciccio, C.; García-Bañuelos, L.; López-Pintado, O. Extracting Event Logs for Process Mining from Data Stored on the Blockchain. In *Business Process Management Workshops*; Springer: Berlin/Heidelberg, Germany, 2019.
86. Klinkmüller, C.; Müller, R.; Weber, I. Mining Process Mining Practices: An Exploratory Characterization of Information Needs in Process Analytics. In *International Conference on Business Process Management*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 322–337.
87. Del-Río-Ortega, A.; Resinas, M.; Cabanillas, C.; Ruiz-Cortés, A. On the definition and design-time analysis of process performance indicators. *Inf. Syst.* **2013**, *38*, 470–490.
88. Kis, I.; Bachhofner, S.; Di Ciccio, C.; Mendling, J. Towards a data-driven framework for measuring process performance. In *Enterprise, Business-Process and Information Systems Modeling*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 3–18.
89. Bachhofner, S.; Kis, I.; Di Ciccio, C.; Mendling, J. Towards a Multi-parametric Visualisation Approach for Business Process Analytics. In Proceedings of the Advanced Information Systems Engineering Workshops, Essen, Germany, 12–16 June 2017; pp. 85–91.
90. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
91. Choy, C.; Gwak, J.; Savarese, S. 4D Spatio Temporal ConvNet: Minkowski Convolutional Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–21 June 2019.
92. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.

93. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
94. Glorot, X.; Bordes, A.; Bengio, Y. Deep sparse rectifier neural networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL, USA, 22–23 April 2011; pp. 315–323.
95. Dahl, G.E.; Sainath, T.N.; Hinton, G.E. Improving deep neural networks for LVCSR using rectified linear units and dropout. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 8609–8613.
96. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
97. Bjorck, N.; Gomes, C.P.; Selman, B.; Weinberger, K.Q. Understanding batch normalization. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 7694–7705.
98. Waldhauser, C.; Hochreiter, R.; Otepka, J.; Pfeifer, N.; Ghuffar, S.; Korzeniowska, K.; Wagner, G. Automated classification of airborne laser scanning point clouds. In *Solving Computationally Expensive Engineering Problems*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 269–292.
99. Goutte, C.; Gaussier, E. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *European Conference on Information Retrieval*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 345–359.
100. Zhang, D.; Wang, J.; Zhao, X. Estimating the uncertainty of average F1 scores. In Proceedings of the 2015 International Conference on the Theory of Information Retrieval, New York, NY, USA, 27–30 September 2015; pp. 317–320.
101. Yue, Y.; Finley, T.; Radlinski, F.; Joachims, T. A support vector method for optimizing average precision. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 23–27 July 2007; pp. 271–278.
102. Cohen, J. A coefficient of agreement for nominal scales. *Educ. Psychol. Meas.* **1960**, *20*, 37–46.
103. Plaza-Leiva, V.; Gomez-Ruiz, J.A.; Mandow, A.; García-Cerezo, A. Voxel-based neighborhood for spatial shape pattern classification of lidar point clouds with supervised learning. *Sensors* **2017**, *17*, 594.
104. Jurman, G.; Riccadonna, S.; Furlanello, C. A comparison of MCC and CEN error measures in multi-class prediction. *PLoS ONE* **2012**, *7*, e41882.
105. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic Differentiation in PyTorch. In Proceedings of the Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
106. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional Architecture for Fast Feature Embedding. In Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 675–678.
107. R Core Team. *R: A Language and Environment for Statistical Computing*; R Foundation for Statistical Computing: Vienna, Austria, 2018.
108. Pfeifer, N.; Mandlburger, G.; Otepka, J.; Karel, W. OPALS—A framework for Airborne Laser Scanning data analysis. *Comput. Environ. Urban Syst.* **2014**, *45*, 125–136.
109. Trimble. *Match-T DSM Reference Manual*; Trimble, Inc.: Sunnyvale, CA, USA, 2016
110. Trimble Geospatial. Available online: http://www.inpho.de (accessed on 7 March 2019).
111. Hu, F.; Xia, G.S.; Hu, J.; Zhang, L. Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sens.* **2015**, *7*, 14680–14707.
112. Sun, Y.; Tian, Y.; Xu, Y. Problems of encoder-decoder frameworks for high-resolution remote sensing image segmentation: Structural stereotype and insufficient learning. *Neurocomputing* **2019**, *330*, 297–304.
113. Bischke, B.; Helber, P.; Folz, J.; Borth, D.; Dengel, A. Multi-Task Learning for Segmentation of Building Footprints with Deep Neural Networks. *arXiv* **2017**, arXiv:1709.05932.
114. Yi, Y.; Zhang, Z.; Zhang, W.; Zhang, C.; Li, W.; Zhao, T. Semantic Segmentation of Urban Buildings from VHR Remote Sensing Imagery Using a Deep Convolutional Neural Network. *Remote Sens.* **2019**, *11*, 1774.

115. Guo, Z.; Wu, G.; Song, X.; Yuan, W.; Chen, Q.; Zhang, H.; Shi, X.; Xu, M.; Xu, Y.; Shibasaki, R.; et al. Super-Resolution Integrated Building Semantic Segmentation for Multi-Source Remote Sensing Imagery. *IEEE Access* **2019**, *7*, 99381–99397.
116. Eerapu, K.K.; Ashwath, B.; Lal, S.; Dell'acqua, F.; Dhan, A.N. Dense Refinement Residual Network for Road Extraction from Aerial Imagery Data. *IEEE Access* **2019**, *7*, 151764–151782.
117. Marmanis, D.; Wegner, J.D.; Galliani, S.; Schindler, K.; Datcu, M.; Stilla, U. Semantic segmentation of aerial images with an ensemble of CNNs. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2016**, *3*, 473.
118. Li, Y.; Zhang, H.; Shen, Q. Spectral–spatial classification of hyperspectral imagery with 3D convolutional neural network. *Remote Sens.* **2017**, *9*, 67.
119. Sellami, A.; Farah, M.; Farah, I.R.; Solaiman, B. Hyperspectral imagery classification based on semi-supervised 3-D deep neural network and adaptive band selection. *Expert Syst. Appl.* **2019**, *129*, 246–259.
120. Yu, X.; Hyyppä, J.; Vastaranta, M.; Holopainen, M.; Viitala, R. Predicting individual tree attributes from airborne laser point clouds based on the random forests technique. *ISPRS J. Photogramm. Remote. Sens.* **2011**, *66*, 28–37.
121. Otepka, J.; Ghuffar, S.; Waldhauser, C.; Hochreiter, R.; Pfeifer, N. Georeferenced point clouds: A survey of features and point cloud management. *ISPRS Int. J. Geo-Inf.* **2013**, *2*, 1038–1065.
122. Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4490–4499.
123. Hua, B.S.; Tran, M.K.; Yeung, S.K. Pointwise convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 984–993.
124. Vo, A.V.; Truong-Hong, L.; Laefer, D.F.; Tiede, D.; d'Oleire Oltmanns, S.; Baraldi, A.; Shimoni, M.; Moser, G.; Tuia, D. Processing of extremely high resolution LiDAR and RGB data: Outcome of the 2015 IEEE GRSS data fusion contest—Part B: 3-D contest. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2016**, *9*, 5560–5575.
125. Vo, A.V.; Truong-Hong, L.; Laefer, D.F. Aerial laser scanning and imagery data fusion for road detection in city scale. In Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milano, Italy, 26–31 July 201; pp. 4177–4180.
126. Maxwell, A.E.; Warner, T.A.; Fang, F. Implementation of machine-learning classification in remote sensing: An applied review. *Int. J. Remote Sens.* **2018**, *39*, 2784–2817.
127. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement learning: A survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285.
128. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484.
129. Nadell, C.C.; Huang, B.; Malof, J.M.; Padilla, W.J. Deep learning for accelerated all-dielectric metasurface design. *Opt. Express* **2019**, *27*, 27523–27535.
130. Stokes, J.M.; Yang, K.; Swanson, K.; Jin, W.; Cubillos-Ruiz, A.; Donghia, N.M.; MacNair, C.R.; French, S.; Carfrae, L.A.; Bloom-Ackerman, Z.; et al. A Deep Learning Approach to Antibiotic Discovery. *Cell* **2020**, *180*, 688–702.
131. Lipton, Z.C.; Steinhardt, J. Troubling trends in machine learning scholarship. *arXiv* **2018**, arXiv:1807.03341.
132. Gundersen, O.E.; Kjensmo, S. State of the art: Reproducibility in artificial intelligence. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LO, USA, 2–7 February 2018.
133. Gibney, E. This AI researcher is trying to ward off a reproducibility crisis. *Nature* **2019**, *577*, 14.
134. Ferrari Dacrema, M.; Cremonesi, P.; Jannach, D. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019), Copenhagen, Denmark, 16–20 September 2019.
135. Fu, W.; Menzies, T. Easy over hard: A case study on deep learning. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, Paderborn, Germany, 4–8 September 2017; pp. 49–60.
136. Baker, M. Over half of psychology studies fail reproducibility test. *Nat. News* **2015**. Available online: https://www.nature.com/news/over-half-of-psychology-studies-fail-reproducibility-test-1.18248 (accessed on 7 April 2020). [CrossRef]
137. Baker, M. 1,500 scientists lift the lid on reproducibility. *Nat. News* **2016**, *533*, 452. [CrossRef] [PubMed]

138. Gannot, G.; Cutting, M.A.; Fischer, D.J.; Hsu, L.J. Reproducibility and transparency in biomedical sciences. *Oral Dis.* **2017**, *23*, 813.
139. Goodman, S.N.; Fanelli, D.; Ioannidis, J.P. What does research reproducibility mean? *Sci. Transl. Med.* **2016**, *8*, 341ps12.
140. Nuzzo, R. How scientists fool themselves–and how they can stop. *Nat. News* **2015**, *526*, 182.
141. Menke, J.; Roelandse, M.; Ozyurt, B.; Martone, M.; Bandrowski, A. Rigor and Transparency Index, a new metric of quality for assessing biological and medical science methods. *bioRxiv* **2020**. [CrossRef]
142. Chawla, D.S. *Software Searches out Reproducibility Issues in Scientific Papers*. 2020. Available online: https://www.nature.com/articles/d41586-020-00104-6 (accessed on 7 April 2020).
143. Breiman, L. *Classification and Regression Trees*; Routledge: Abingdon-on-Thames, UK, 2017.
144. Weinmann, M.; Jutzi, B.; Mallet, C. Feature relevance assessment for the semantic interpretation of 3D point cloud data. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2013**, *5*, 1.
145. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
146. Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA Neural Netw. Mach. Learn.* **2012**, *4*, 26–31.
147. Ressl, C.; Mandlburger, G.; Pfeifer, N. Investigating Adjustment of Airborne Laser Scanning Strips without Usage of GNSS/IMU Trajectory Data. In Proceedings of the ISPRS Workshop Laserscanning 09, Paris, France, 1–2 September 2009; pp. 195–200.
148. Loghin, A.M.; Otepka, J.; Karel, W.; Pöchtrager, M.; Pfeifer, N. Analysis of Digital Elevation Models from Very High Resolution Satellite Imagery. In Proceedings of the Dreiländertagung OVG–DGPF–SGPF, Vienna, Austria, 20–22 February 2019; pp. 123–137.
149. Bernard, M.; Decluseau, D.; Gabet, L.; Nonin, P. 3D capabilities of Pleiades satellite. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci* **2012**, *39*, B3.
150. Hu, F.; Gao, X.; Li, G.; Li, M. Dem Extraction From Worldview-3 Stereo-Images And Accuracy Evaluation. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2016**, *41*. Available online: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLI-B1/327/2016/isprs-archives-XLI-B1-327-2016.pdf (accessed on 7 April 2020).
151. Hobi, M.L.; Ginzler, C. Accuracy assessment of digital surface models based on WorldView-2 and ADS80 stereo remote sensing data. *Sensors* **2012**, *12*, 6347–6368.
152. Hoffmann, A.; Lehmann, F. Vom Mars zur Erde-die erste digitale Orthobildkarte Berlin mit Daten der Kamera HRSC-A. *Kartographische Nachrichten* **2000**, *50*, 61–71.
153. Ressl, C.; Brockmann, H.; Mandlburger, G.; Pfeifer, N. Dense Image Matching vs. Airborne Laser Scanning–Comparison of two methods for deriving terrain models. *Photogramm.-Fernerkund.-Geoinf.* **2016**, *2016*, 57–73.
154. Donoho, D.L. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Chall. Lect.* **2000**, *1*, 32.
155. Messay-Kebede, T.; Narayanan, B.N.; Djaneye-Boundjou, O. Combination of traditional and deep learning based architectures to overcome class imbalance and its application to malware classification. In Proceedings of the NAECON 2018-IEEE National Aerospace and Electronics Conference, Dayton, OH, USA, 23–26 July 2018; pp. 73–77.
156. Saito, S.; Huang, Z.; Natsume, R.; Morishima, S.; Kanazawa, A.; Li, H. PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization. *arXiv* **2019**, arXiv:1905.05172.
157. Kim, J.; Kwon Lee, J.; Mu Lee, K. Accurate image super-resolution using very deep convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1646–1654.
158. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; Volume 48, pp. 1050–1059.
159. Schmitt, M.; Zhu, X.X. Data fusion and remote sensing: An ever-growing relationship. *IEEE Geosci. Remote Sens. Mag.* **2016**, *4*, 6–23.
160. Tran, T.; Ressl, C.; Pfeifer, N. Integrated change detection and classification in urban areas based on airborne laser scanning point clouds. *Sensors* **2018**, *18*, 448.

161. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.