remote sensing

MDPI

Article

D-Net: A Density-Based Convolutional Neural Network for Mobile LiDAR Point Clouds Classification in Urban Areas

Mahdiye Zaboli [1,*], Heidar Rastiveis [1,2], Benyamin Hosseiny [1], Danesh Shokri [1], Wayne A. Sarasua [3] and Saeid Homayouni [4]

[1] Department of Photogrammetry and Remote Sensing, School of Surveying and Geospatial Engineering, College of Engineering, University of Tehran, Tehran 141663, Iran; hrasti@ut.ac.ir (H.R.); ben.hosseiny@ut.ac.ir (B.H.); daneshshokri72@ut.ac.ir (D.S.)
[2] Lyles School of Civil Engineering, Purdue University, West Lafayette, IN 47907, USA
[3] Glenn Department of Civil Engineering, Clemson University, Clemson, SC 29634, USA; sarasua@clemson.edu
[4] Centre Eau Terre Environnement, Institut National de la Recherche Scientifique, 490 Rue de la Couronne, Quebec City, QC G1K 9A9, Canada; saeid.homayouni@inrs.ca
* Correspondence: m_zaboli@ut.ac.ir

Abstract: The 3D semantic segmentation of a LiDAR point cloud is essential for various complex infrastructure analyses such as roadway monitoring, digital twin, or even smart city development. Different geometric and radiometric descriptors or diverse combinations of point descriptors can extract objects from LiDAR data through classification. However, the irregular structure of the point cloud is a typical descriptor learning problem—how to consider each point and its surroundings in an appropriate structure for descriptor extraction? In recent years, convolutional neural networks (CNNs) have received much attention for automatic segmentation and classification. Previous studies demonstrated deep learning models' high potential and robust performance for classifying complicated point clouds and permutation invariance. Nevertheless, such algorithms still extract descriptors from independent points without investigating the deep descriptor relationship between the center point and its neighbors. This paper proposes a robust and efficient CNN-based framework named D-Net for automatically classifying a mobile laser scanning (MLS) point cloud in urban areas. Initially, the point cloud is converted into a regular voxelized structure during a preprocessing step. This helps to overcome the challenge of irregularity and inhomogeneity. A density value is assigned to each voxel that describes the point distribution within the voxel's location. Then, by training the designed CNN classifier, each point will receive the label of its corresponding voxel. The performance of the proposed D-Net method was tested using a point cloud dataset in an urban area. Our results demonstrated a relatively high level of performance with an overall accuracy (OA) of about 98% and precision, recall, and F1 scores of over 92%.

Keywords: point cloud classification; deep learning; voxelization; automated object detection; mobile laser scanning

Citation: Zaboli, M.; Rastiveis, H.; Hosseiny, B.; Shokri, D.; Sarasua, W.A.; Homayouni, S. D-Net: A Density-Based Convolutional Neural Network for Mobile LiDAR Point Clouds Classification in Urban Areas. *Remote Sens.* **2023**, *15*, 2317. https://doi.org/10.3390/rs15092317

Academic Editors: Wei Yao, John Trinder, Wenbing Tao and Jie Shao

Received: 16 March 2023
Revised: 19 April 2023
Accepted: 26 April 2023
Published: 27 April 2023

1. Introduction

Mobile Laser Scanning (MLS) systems provide high-density three-dimensional spatial data from nearby surroundings. They can provide reliable and accurate depth information for object localization and structure characterization. MLS has been widely used for collecting roadside assets; however, the manual processing of such huge point cloud data can be time-consuming and tedious. Recently, several techniques have been proposed for automatic and individual object extraction from MLS point clouds, including power lines, traffic signs, building facades, trees, and pavement markings [1–3]. These applications are highly specialized and utilized in urban management, making it cumbersome to classify different objects simultaneously because a different classifier would be required to run for

each object. Thus, there is a need for a comprehensive object classifier for MLS point cloud semantic segmentation.

Object diversity and LiDAR point distribution variations make classifying multiple objects challenging [2,4]. In particular, MLS point clouds have significant properties which must be considered, such as uneven point density, vacant holes caused by occlusions, and complex structures (e.g., points of poles mixed with points of trees). Additionally, diverse object sizes (e.g., small objects 1 m long, large ones with a length of up to hundreds of meters) in urban environments are another noticeable characteristic of the point clouds. Figure 1 illustrates the point clouds of different types of objects and incomplete or variable point distributions.

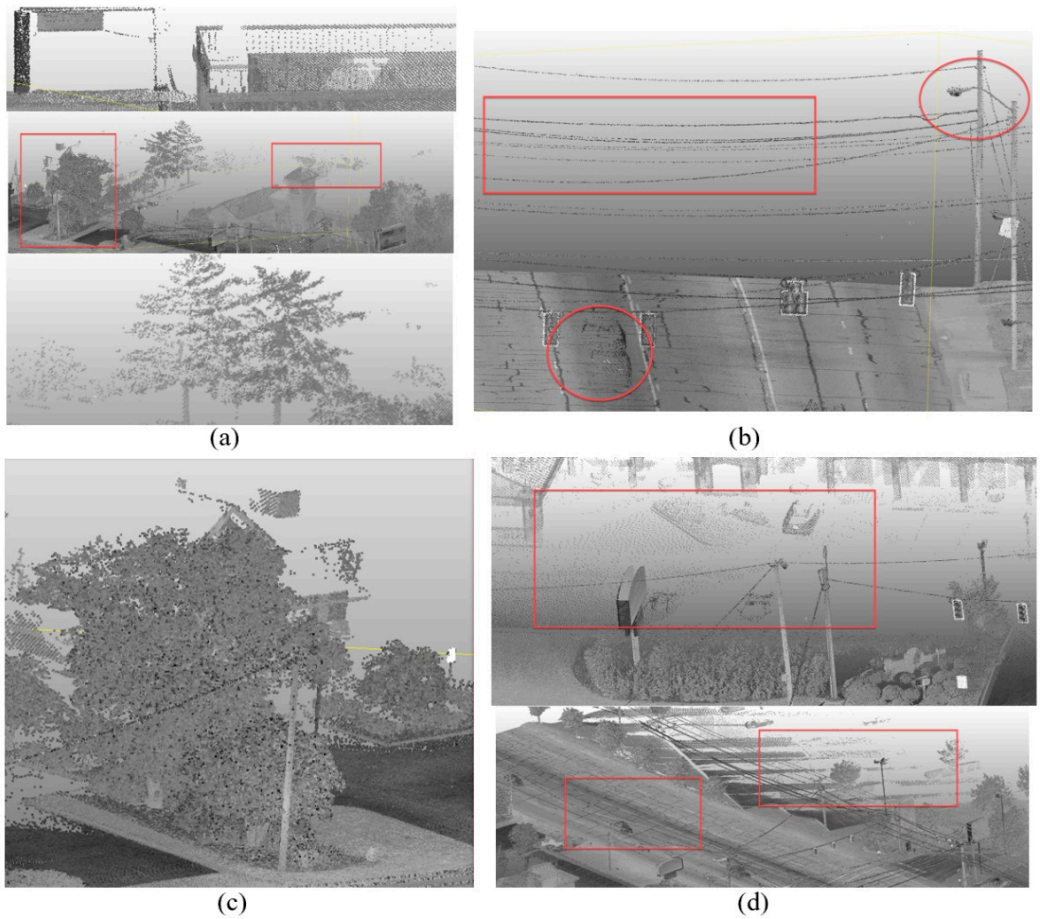


Figure 1. The MLS challenges: (**a**,**b**) imperfect objects, (**c**) mixed structures (poles and trees), and (**d**) variable point densities.

Generally, MLS data classification algorithms are grouped into two categories. First there are the descriptor-based classification methods that consist of two steps: hand-crafted descriptors extraction from LiDAR data, and the classification of descriptors using conventional machine learning (ML) algorithms. The second category contains deep learning (DL)-based methods that generate many descriptors unsupervised and then automatically perform a neural network classification.

This paper focuses on MLS data classification through a fully automated deep neural network architecture. The following background literature review of both descriptor-based and deep learning classification methods provides context for the methodology, implementation, and experimental results discussed later in the paper.

2. Point Cloud Classification

2.1. Descriptor-Based Methods

Descriptor-based methods include two separate steps: descriptor extraction and semantic classification [2,5]. In the descriptor extraction step, several geometric or radiometric descriptors, e.g., eigenvalues, height, intensity, curvature, linearity, and planarity, are generated from the local neighborhood of each point [5,6]. These descriptors are commonly used in ML-based classifiers such as Support Vector Machine (SVM), Random Forest (RF), K Nearest Neighbor (K-NN), Multi-Layer Perceptron (MLP), and Gaussian Naive Bayes (GNB). These methods predict the class of each object from user-extracted descriptors, where ignoring label consistency is a drawback [3,4,7]. Thus, both noisy points and label inconsistency may reduce the output accuracy of the classification process. Many research studies have incorporated contextual information into the classification process to deal with this issue. Conditional Random Fields, Non-Associative Markov Networks, and Associative Markov Networks are three examples of contextual models for improving machine learning classifiers [4,5]. Since these ML-based algorithms use hand-crafted descriptors to classify point clouds, their applications for large-scale areas are a primary limitation of these methods. Additionally, overfitting will result in the poor implementation of multi-class classification due to insignificant model generalization to novel data. Consequently, models are inaccurate when trying to predict outputs when noise or random fluctuations are learned and picked up in the training data as concepts.

Various descriptor-based methods have been developed for object classification. Lehtomäki et al. [2] divided non-ground points into segments using three sets of descriptors and created local descriptor histograms (LDHs), point distributions, general shape, and spin images. Sun et al. [8] presented an automated method for generating multi-scale super voxels using point attributes such as intensity. Günen et al. [9] used detrended geometric descriptors extracted from super voxel-based local contexts to classify point clouds. Additionally, Zolanvari [10] introduced a new geometric descriptor set composed of 14 descriptors and an adaptive neighborhood size selection method to improve classification accuracy.

In summary, descriptor-based classification methods can be sensitive to point cloud noise, and the manual definition of optimal descriptors is time-consuming and labor-intensive. Further, generalization in large-scale areas may result in the lower accuracy of classification output.

2.2. Deep Learning Based Methods

Unlike a descriptor-based method, DL-based methods automatically extract high-level descriptors without needing descriptor engineering. These high-level descriptors are then used for 3D point cloud classification. DL algorithms have been used extensively in various remote sensing applications for image and point cloud data classification [1,4,11].

Several approaches have been identified in the literature that use DL processing of the raw point cloud without converting into a regular representation such as grid images or voxels. For example, Reference [12] designed a 3D CNN system for point cloud labeling that does not need prior knowledge. Reference [4] designed a PointNet that consumes the point cloud directly for various applications, including object classification and segmentation, while respecting the input data's permutation invariance. The PointNet++ structure was introduced for learning multi-scale features, which is the main drawback of PointNet [13]. Wen et al. [14] proposed a direct point cloud processing architecture named OG-PointNet++, which is based on PointNet and PointNet++ but improves the efficiency when dealing with point cloud density that varies with location. This structure divides the point cloud before inputting it into the network to reduce computation. Ref. [15] imported 3D coordinate data directly from the XYZ coordinate space and the intensity as input data of a CNN network named directionally constrained fully CNN (D-FCN). Their method introduced a direct point cloud convolution module for extracting local descriptors from the projected 2D fields.

Wang et al. [16] defined a novel CNN via multi-scale occupancy to enable multi-scale point descriptor learning. They proposed a training method for balancing several points per class during each epoch. These achievements were realized by locally concentrating on shapes and considering them in a multi-scale mode. Wang et al. [17] incorporated the local and global constraints by designing a multi-scale convolutional network (MSNet) that directly considers the position of the points.

To prevent the rasterization step, Li et al. [18] constructed a deep neural network with spatial pooling (DNNSP), which extracted descriptors. Then, minimum distance spanning tree-based pooling extracted spatial information. Point-based features were determined by the Multi-Layer Perceptron (MLP) to ensure that the DNNSP was independent of the sizes of the point clusters. Li et al. [19] developed a geometric graph convolution network for exploring high-level geometric correlations. Their study proposed a fully automated end-to-end graph neural network that they named the Taylor Gaussian Mixture Model Network (TGNet). Employing a multi-scale hierarchical architecture based on a Taylor Gaussian Convolution (TGConv) operation at several scales has improved the scale invariance challenge. A geometry-attentional network was applied to an aerial LiDAR scanning (ALS) point cloud for multi-scale classification by Boulch et al. [20]. They used a dense end-to-end hierarchy and an elevation-oriented unit to improve classification performance. Song [21] proposed continuous kernels and presented a generalization of CNN for point cloud processing. Their presented formulation is simple and suitable for designing neural networks and enabling various applications such as classification and large-scale segmentation.

Geng et al. [22] designed a CNN-based method by transforming 3D point data into Hough space. Then, the accumulator count of grids of the rasterized Hough space was computed and used as an input for a CNN model for object classification. Hoang et al. [23] proposed a Multi-Scale Attentive Aggregation Network (MSAAN) to acquire the consistency of the descriptors. Aijazi et al. [24] developed a novel framework, GSV-NET, for extracting and combining both global and regional descriptors using a 3D wide-inception CNN structure.

Overall, a directly consuming LiDAR point cloud is a primary advantage of point-based classification methods despite using intensity values as input data in some cases. These methods have a drawback regarding multiscale descriptor representation for more and better descriptor learning.

2.3. Summary of the Previous Works

Table 1 summarizes the reviewed methods based on their study area type, data, method, and results. Some research studies generate 2D descriptor images from 3D point clouds, resulting in lost spatial information. This requires prior knowledge with a weak capability for more complex and varied structural segmentation. Others voxelized point clouds into regular grids for importing 3D data into a CNN for extracting descriptors and carrying out classification. Some methods directly apply to the 3D point cloud for classification, but do not consider multiscale descriptors. They may need to integrate manually defined descriptors to improve classification performance in complicated tasks.

Table 1. Summary of related studies on point clouds classification.

	Authors	Area Type	Size (M pts.)	Point Cloud Type	Algorithm Details	Semantic Classes	Overall Accuracy (%)
Descriptor-Based Methods	Weinmann et al. [25]	Urban Area	0.027~0.11	ALS and TLS	Points, feature-based	Building, Road, Tree, Pole, Car	90.1~95.5 92.8~94.5
	Lehtomäki et al. [2]	Road environment	9	MLS	Points, feature-based	Tree, Car, Pedestrian, Lamp post, Hoarding, Traffic pole, Undefined	87.9
	Yang et al. [26]	Urban Area	0.2	MLS	Supervoxels, feature-based	Tree, Street lamp, Building, Utility poles, Enclosure, Car, Traffic sign, Other	91.1, 92.3
	Han et al. [6]	Road infrastructure	-	MLS	Points, feature-based	Pole, Street Lamp, Sign Board, Direction Sign,	94.3, 93.3
	Sun et al. [8]	Urban Area	~50	MLS	Points, feature-based	Buildings, Terrain, Scanning artifacts, Massive vegetation, Hardscape, Natural terrain, Cars	92
Deep-learning Methods	Qi and Yi [12]	Urban Area	-	ALS + MLS	Voxels, deep learning (CNN)	Plane, Tree, Building, Car, Pole, Wire, Others	93
	Boulch et al. [27]	Urban Area	~0.75	ALS	Points, deep learning (CNN)	Car, Powerline, Façade, Fence/Hedge, Low vegetation, Shrub, Impervious Surfaces, Tree, Roof	82.3
	Shukor et al. [28]	Urban Area	-	ALS and TLS	Points, deep learning (CNN) (Snapnet)	Natural terrain, High vegetation, Scanning artifacts, Buildings, Hardscape, Cars, Man-made terrain, Low vegetation,	88.6
	Li et al. [18]	Urban Area	7.5~8.1	ALS and TLS	Points, deep learning (DNNSP)	Building, Low vegetation, Hardscape, Scanning artifacts, High vegetation, Cars, Natural Terrain,	98.2 97.4
	Wang et al. [16]	Urban Area	79.5	ALS and oblique aerial photogrammetry	Points, deep learning, and machine learning	Natural terrain, Scanning artifacts, Low vegetation, Buildings, Hardscape, Man-made terrain, High vegetation, Cars	84.8
	Li et al. [19]	Urban Area	140	ALS and TLS	Points, deep learning (TGNet)	Ground, Pedestrian, Car, Pole, Bollard, Barrier, Building, Natural, Trash can,	96.97
	Wang et al. [1]	Urban Area	1.36	MLS	Points, deep learning (CNN)	Vegetation, Wire, Pole, Ground, Facade	94.75
	Song [21]	Urban Area	16	TLS	Points, deep learning (continuous CNN)	High veg, Low veg, Natural, Man-made, Buildings, Cars, Hardscape, Artefacts	93.4
	Geng et al. [22]	Urban Area	-	LiDAR Sensors	Rasters, deep learning (CNN)	Wall, Bush, Pedestrian, Tree	93.3
	Reference [15]	Urban Area	0.41	ALS	Points, deep learning (CNN)	Power, Car, Shrub, Roof, Façade, Imp_surf, Fence_hedge, Low_veg, Tree	82.2
	Yang et al. [7]	Urban Area	80	MLS	Points, deep learning	Natural, Building, Pole, Road, Road Markings, Car, Fence, Utility line,	93.6
	Aijazi et al. [24]	Computer and Real-World Data	-	LiDAR Sensors	Points, deep learning	Airplane, Chair, Bottle, Bed, Bench, Bowl, Car, Bookshelf, Bathtub, Cone	92.7

The considered methods have been evaluated regarding data type, proposed algorithm, selected semantic classes, and acquisition of accuracies. The input data for all cases were acquired with LiDAR sensors, but in Boulch et al. [4,6,18,20,21,28] several datasets were used, including both indoor and outdoor data. Some methods, such as those by [11,21,25,27], used a preprocessing step that involves voxelization or creating feature

images. Except [22], all methods have many considered classes for classification. Additionally, in this comparison, all cases involved a similar type and number of outdoor objects except [4], which included indoor datasets.

The related works have obtained acceptable outputs on different point cloud datasets. However, we propose a density-based CNN-based framework (D-Net) that can predict point labels for arbitrary point cloud sizes to address some of the issues with previous methods. Compared with the proposed deep learning approaches for point cloud classification, D-Net will take only raw 3D coordinates as the input and does not need additional, manually defined descriptors. Accordingly, the main positive contributions of our work are as follows:

A density-based voxelization procedure was implemented to handle the negative effects of the sparse distribution and small size of objects. This strategy can even be beneficial in the classification of high-density objects.

An efficient density-based CNN architecture is proposed to extract robust descriptors through point density and classify all points despite the imbalance between majority and minority classes.

3. Proposed D-Net Method

Figure 2 provides an overview of the proposed algorithm. In the first step, preprocessing, 3D voxels were generated by considering a specified voxel size after removing outliers from the point cloud. Then, deep nonlinear descriptors were extracted from the input voxel network and imported into fully connected (FC) Layers to label points in each voxel.

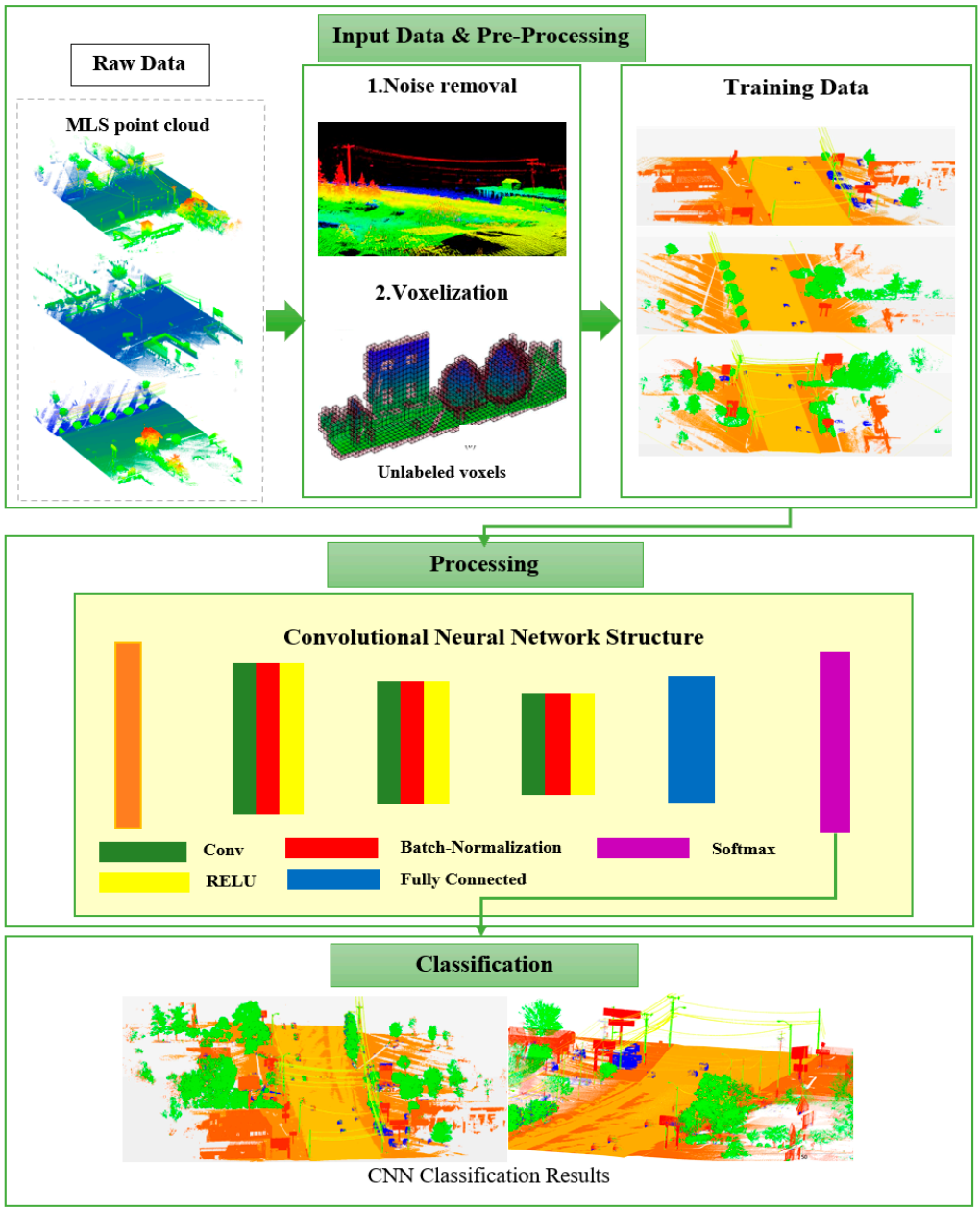


Figure 2. The workflow of the proposed MLS point cloud classification method.

3.1. Pre-Processing

Raw 3D LiDAR point clouds are usually contaminated with noise and outliers due to several factors, such as moving objects, changes in lighting, and sensor characteristics. In this research, we used the proposed method by Shukor and Rushforth [29] for outlier removal. Their framework generates a 3D histogram of point cloud data to count the number of acquired points in multiple projections. The noise data have lower frequencies in this histogram, with a sparser density than other recorded points, and these points can be removed.

The filtered point cloud was then voxelized at specified intervals in 3D space to convert an irregular point cloud to a regular structure. Different values can be considered for the voxel dimensions to select the best value for implementation, considering object sizes and point cloud density. For instance, using a $10 \times 10 \times 10$ cm^3 as the neighborhood region increases the focus on details and retains sampling resolution. The point density of MLS data can change by varying the distance between the laser scanner and scanned

objects. Due to occlusion, some objects may not be completely observed. In addition, some voxels may be empty, and some may have a high point density depending on the selected voxel sizes. Figure 3 illustrates the density-based voxelization process for several irregular sample points. In the next step, the extracted density-based voxels are classified using the designed D-Net classifier.

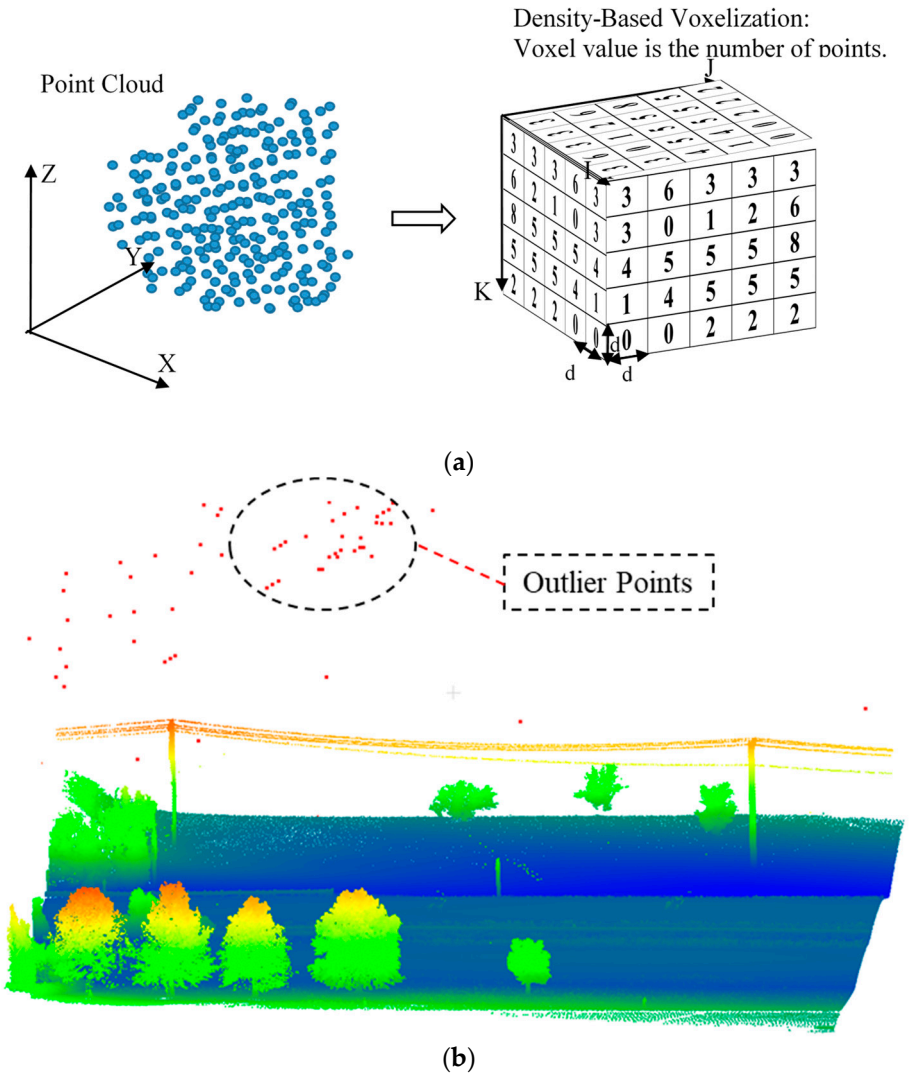


Figure 3. Visualization of (**a**) density-based point cloud voxelization and (**b**) outlier points.

3.2. Network Architecture

Deep learning CNNs consist of multiple adaptive filters which provide the capability of feature learning without needing prior knowledge [30]. Typically, each CNN layer includes three steps. First, a set of linear activations is produced by performing several parallel convolutions. In the second step, a nonlinear activation function such as the sigmoid function is employed for running each linear activation. In the third step, further output modification is carried out using a pooling function [31,32].

The proposed CNN-based classifier consists of three main sections: input data, the deep descriptor extractor, and the output discriminator (classifier) layer. In this study, the input cube data are considered as 3D patches with $r \times c \times d$ dimensions, where r, c, and d correspond to the patch's rows, columns, and depth (height) in the density-based voxel format of the point cloud. In other words, around each candidate voxel, a cube patch with a size of $r \times c \times d$ voxels is considered the input patch of the designed CNN. In this research, a cube patch with a size of 15 × 15× 11 cm^3 was determined to be the optimum patch

size through a trial-and-error process. The corresponding patch of each voxel was passed through the convolutional layers to measure deep features.

The next section of the D-Net framework is the deep descriptor extraction section, where combining multiple convolutional layers can extract deep and nonlinear descriptors at different levels. Convolutional layers transform the input data using a dot product between the weights and the neurons. Filters, parameter sharing, activation, and hyper-parameters are the major components of the convolutional layers. As shown in Figure 4, each input patch was imported into the network. After three convolutional layers with different filters (7×7, 5×5, and 3×3 kernels), deep descriptors were eventually achieved. The designed network was adopted from the Visual Geometry Group (VGG) [33] as a simpler version with a modified architecture for MLS point cloud classification and fewer parameters to avoid overfitting when using a few training samples.

The obtained deep descriptors were imported into the classification section (FC layers) to produce class predictions or scores. The classification section usually includes one or more FC layers, which compute class scores as the output of the network with the dimension of ($1 \times 1 \times$ N), where N is the number of classes. In our proposed framework, there are two FC layers with 100 neurons in each layer and an eight-neuron Softmax layer to predict the class of each point (a total of eight semantic classes).

Table 2 summarizes the hyperparameters used for the proposed D-Net classifier. The mini-batch Adam optimizer used a learning rate of 0.00005 and a batch size of 32 [34]. Categorical cross-entropy was the loss function used to measure the error rate in each epoch. A 10% dropout at each layer and batch normalization were also considered to increase the generalization and stability. It is worth mentioning that dropout by regularization increased the learning independency between neurons.

Table 2. The considered parameters for implementing the proposed algorithm.

Optimizer	**Adam**
Learning rate	0.00005
Epochs	<500
Mini-batch size	32
Loss function	Categorical Cross-Entropy
Conv. Filters	448 (64 + 128 + 256)
Dropout	10%
FC Neurons	200 (100 + 100)
Activation Function	ReLU
Kernel size	7×7, 5×5, 3×3
Batch Normalization	True

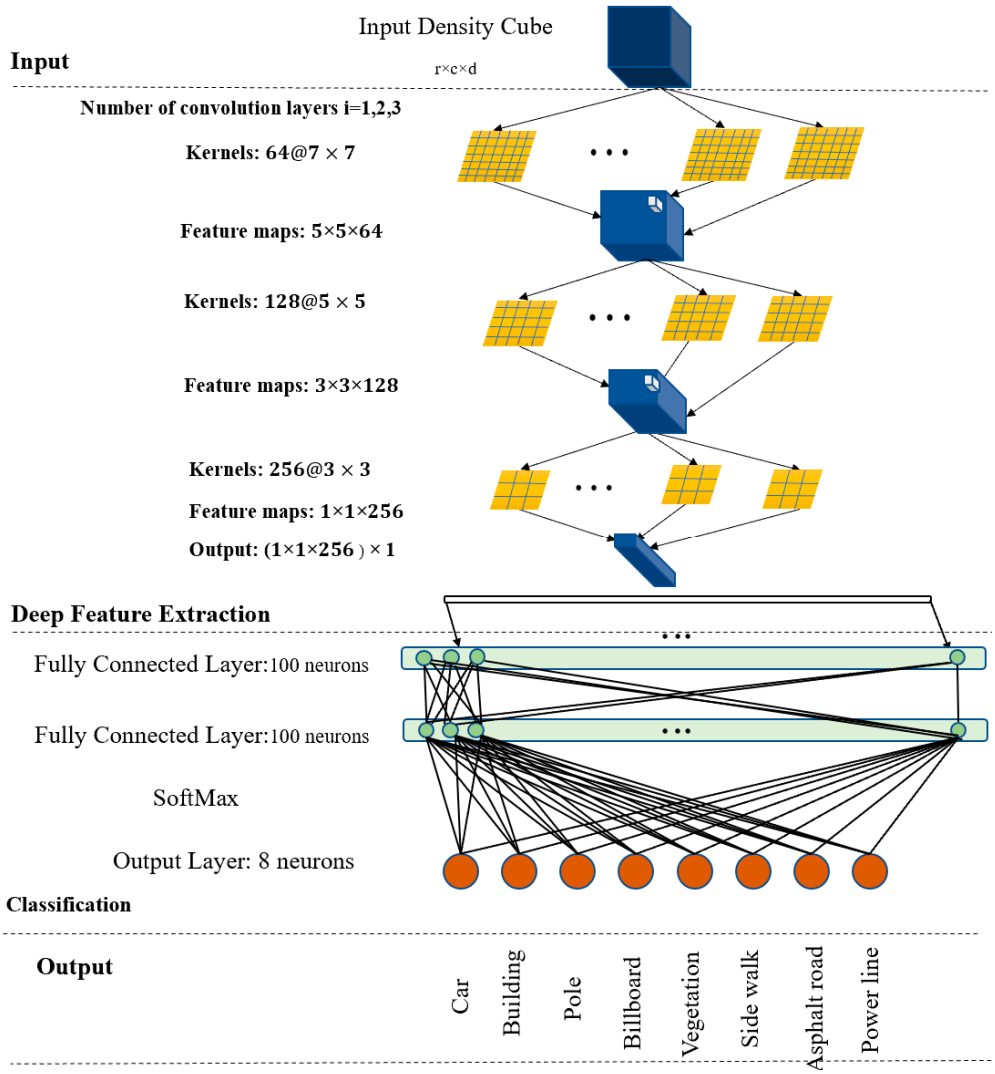


Figure 4. Structure of CNN-based classifier used for the MLS point cloud classification.

4. Experiments and Results

4.1. MLS Dataset

An MLS dataset of a 600 m section of US Highway Route 76 (Clemson Blvd) in Anderson, South Carolina, USA, was used to evaluate the proposed D-Net. This four-lane urban arterial begins at Forest Hill Drive and ends at the intersection with East West Parkway. The setting contains various objects, such as powerlines, poles, buildings, and vegetation. Although each dataset point included various properties such as intensity, position, and GPS time, only the 3D coordinates were utilized as inputs in the examination. Additionally, to evaluate the generalization of the designed D-Net classifier framework, the data were initially divided into three subsections, and the training sample data were only selected from the first subsection. Figure 5 shows an overview of the dataset used in this research.

4.2. Training Data Collection

Collecting sample data for training classifiers is inevitable in every supervised classification algorithm. This study manually classified all the points in the first subsection to create a training and test dataset. The classified point cloud was then voxelized based on density to create a voxelized point cloud. The label of each voxel of the training data was selected based on the most common semantic class available in the voxel. Around each voxel, a cube patch was considered as sample data for the corresponding class label of that

voxel. This paper considered eight classes (buildings, cars, powerlines, vegetation, asphalt road, poles, billboards, and sidewalk). For each class, 70%, 5%, and 25% of the voxels were chosen randomly and considered as the training, validation, and test data, respectively. Additionally, all points of Sections 2 and 3 were used to estimate the generalization of the algorithm. Table 3 summarizes the number of sample points for each class in the first section.

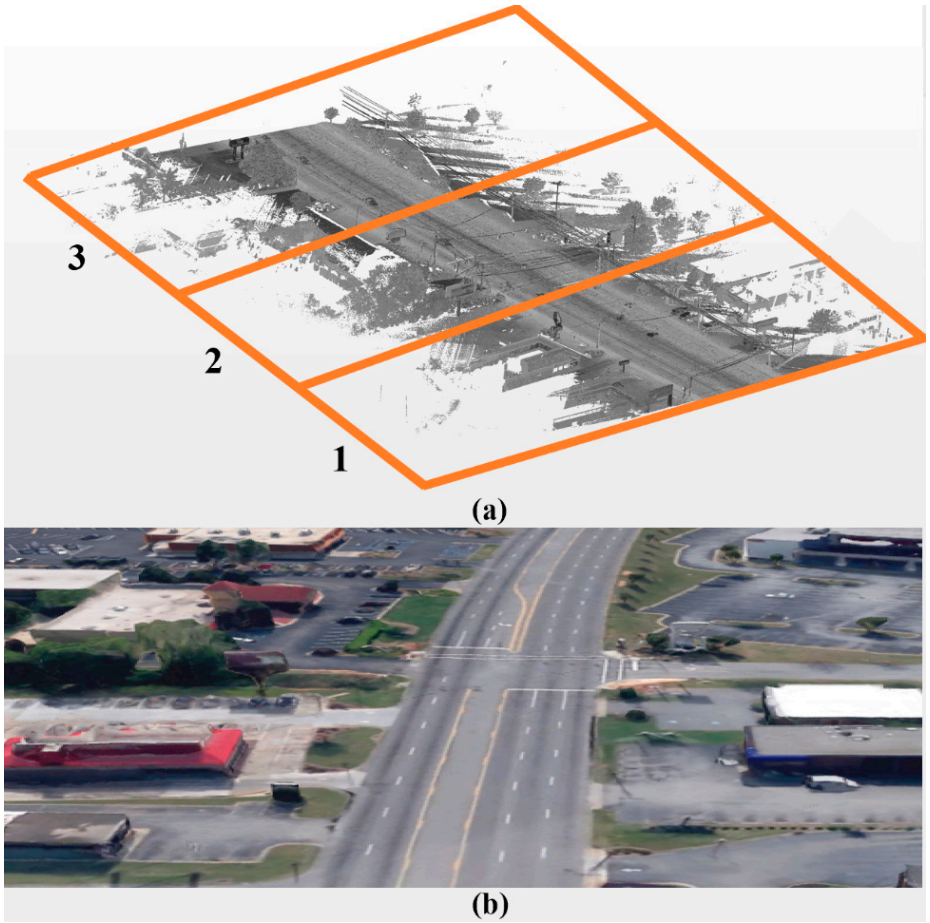


Figure 5. The chosen test section (70% of Section 1 for complete training and other points for validation and test). (**a**) MLS data; (**b**) sample route of the test area (Google Maps, 2023).

Table 3. Quantity of considered sample points in the first section.

Class Points	*Cars*	*Vegetation*	*Poles*	*Powerlines*	*Asphalt road*	*Sidewalk*	*Buildings*	*Billboards*	*Total*
Training	24,722	46,337	24,153	21,200	1,614,692	546,699	125,033	49,952	2,452,788
Validation	1766	3310	1725	1514	115,335	39,050	8931	3568	175,200
Test	79,463	148,941	77,634	68,141	5,190,082	1,757,248	401,893	160,560	7,873,961
All	105,951	198,588	103,512	90,855	6,920,109	2,342,997	535,857	214,080	10,501,949

4.3. Parameter Tuning

Choosing the optimum neighborhood size is essential for accurately identifying the class of points using the D-Net method. This is carried out in two steps: (1) using different patch sizes to find the most optimal classification results; and (2) the optimal selection of

voxel dimensions to regulate the used point cloud. It is worth mentioning that exploiting the optimal optimizer in the CNN will dramatically increase the calculation speed and accuracy. Selecting the optimal structure for the network is one of the vital essential stages in the deep-learning-based classification method. The following section discusses the voxel network size, the 3D sliding patch size, the optimizer used, and their influence on the resulting classified data.

4.3.1. Voxel Size

In this research, five different input voxel sizes with 1000 training samples were tested within the D-Net, and the outputs are demonstrated in Table 4. The processing time per epoch was the same for all voxel sizes. The voxel size equal to 10 cm achieved the best accuracy and seemed to be the most optimal value.

Table 4. Comparison of different input voxel sizes.

Voxel Size (cm)	Processing Time Per Epoch (s)	Test Accuracy (%)
5	3	88.35
8	3	90.23
10	3	**93.82**
15	3	92.49
20	3	92.40

4.3.2. Input Patch Size

Three input patch sizes of 13 × 13 × 11, 15 × 15 × 11, and 17 × 17 × 11 were tested with 70% of the dataset and three convolution layers, and two FC layers. The reason for choosing various patch size values was to investigate the effect of considering the various neighbor samples in correctly identifying the labels. Choosing a constant value of 11 in the z-direction was due to the computational efficiency in the classification. The provided results in Figure 6 indicate a direct relationship between the accuracy and patch dimensions. Therefore, by modifying the patch size, the classification accuracy improves, which may be because the network's receptive field is increased; however, considering a larger patch size may result in more noise and disturbance and a higher computational time of the classifier. Table 5 compares the procedure time and efficiency of the designed classifier for several input patch sizes. In Table 5, the OA of the test data increased from 97.96% for a patch size of 13 × 13 × 11 to 98.49% for 17 × 17 × 11. A patch size of 15 × 15 × 11 was also considered. Increasing the input patch size increased the processing time, and the evaluation parameters' values changed from 15 to 17. Figure 6 also indicates that precision values improved from 97.50% to 98% by changing input patch sizes.

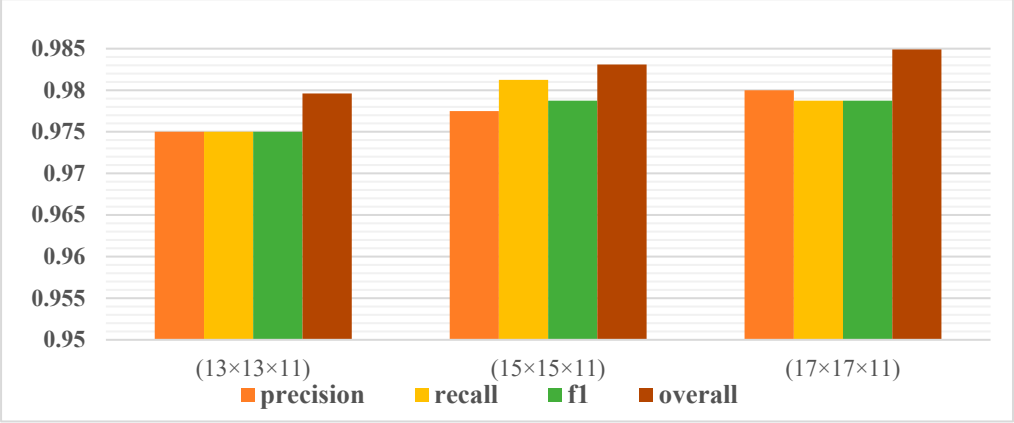


Figure 6. Accuracy assessment of the D-Net method for MLS point cloud classification in the urban area.

Table 5. Comparison of different patch sizes.

Patch Size	Processing Time Per Epoch (min)	Test Accuracy (%)
13 × 13 × 11	1.33	97.96
15 × 15 × 11	1.5	98.31
17 × 17 × 11	1.66	98.49

The evaluation parameters of all classes for the three selected patch sizes are shown in Table 6. A comparison of recall and F1-score in the considered patch sizes shows that the highest recall value occurs with a patch size of 15 × 15 × 11, and F1-score values are equal in sizes 15 × 15 × 11 and 17 × 17 × 11. Thus, selecting the 15 × 15 × 11 patch size is optimal for performing calculations.

Table 6. Evaluating the outputs of the presented algorithm with different patch sizes (section one from the MLS dataset).

Patch Sizes	Measures	Classes (%)							
		Cars	Vegetation	Poles	Powerlines	Asphalt Road	Sidewalk	Buildings	Billboards
13 × 13 × 11	Precision	98.3	99.1	99.6	97.4	100.0	90.4	98.3	99.7
	Recall	90.8	99.3	97.3	99.6	98.4	98.9	99.4	99.4
	F1-score	94.4	99.2	98.4	98.5	99.2	94.5	98.8	99.6
15 × 15 × 11	Precision	98.6	98.4	100.0	95.8	100.0	92.3	99.6	100.0
	Recall	95.3	100.0	95.7	100.0	98.2	99.7	99.6	99.3
	F1-score	96.9	99.2	97.8	97.9	99.1	95.9	99.6	99.7
17 × 17 × 11	Precision	99.4	98.4	100.0	95.7	99.3	95.3	99.4	99.4
	Recall	94.8	98.9	95.4	100.0	99.7	96.4	100.0	99.5
	F1-score	97.0	99.5	97.7	97.9	99.5	95.8	99.7	99.5

4.3.3. Optimizing Methods

The optimizer computes and applies loss gradients to variables [35]. In this study, five different optimizers were evaluated for the proposed D-Net method, and the convergence diagram of the network in each case is shown in Figure 7.

The Adam optimizer was computationally effective and requires little memory [34]. This optimizer approached its optimal value in a shorter time but was accompanied by small fluctuations during convergence, achieving an accuracy of 93.38%. The stochastic gradient descent (SGD) optimizer [36] took more epochs than Adam to converge to its optimal value. It has been associated with small fluctuations during convergence with an accuracy of 93.81%. Although the root means square proportion (RMSprop) optimizer [35] was associated with more fluctuations than the Adam and SGD optimizers, it had an acceptable speed to reach its optimal value with an accuracy of 91.00%. Adadelta is a stochastic gradient descent method based on a per-dimension adaptive learning rate [35]. It converged with the least number of epochs compared to other optimizers but had severe fluctuations, and its resulting accuracy was 91.75%. Adamax, a variant of Adam [34], was associated with the least amount of fluctuation in convergence and an accuracy of 93.94%, but approached its desired value at a slower pace speed than Adam and RMSprop. It is noteworthy that all optimizers had the same processing time in each epoch, but with

different fluctuations. To conclude, Adam, SGD, and Adamax obtained similar results while the Adamax optimizer had the highest accuracy. Adam had the fastest convergence to its optimal value. Thus, in the present study, because of the numerous training samples, Adam optimizer was preferable due to its suitable accuracy with high convergence speed.

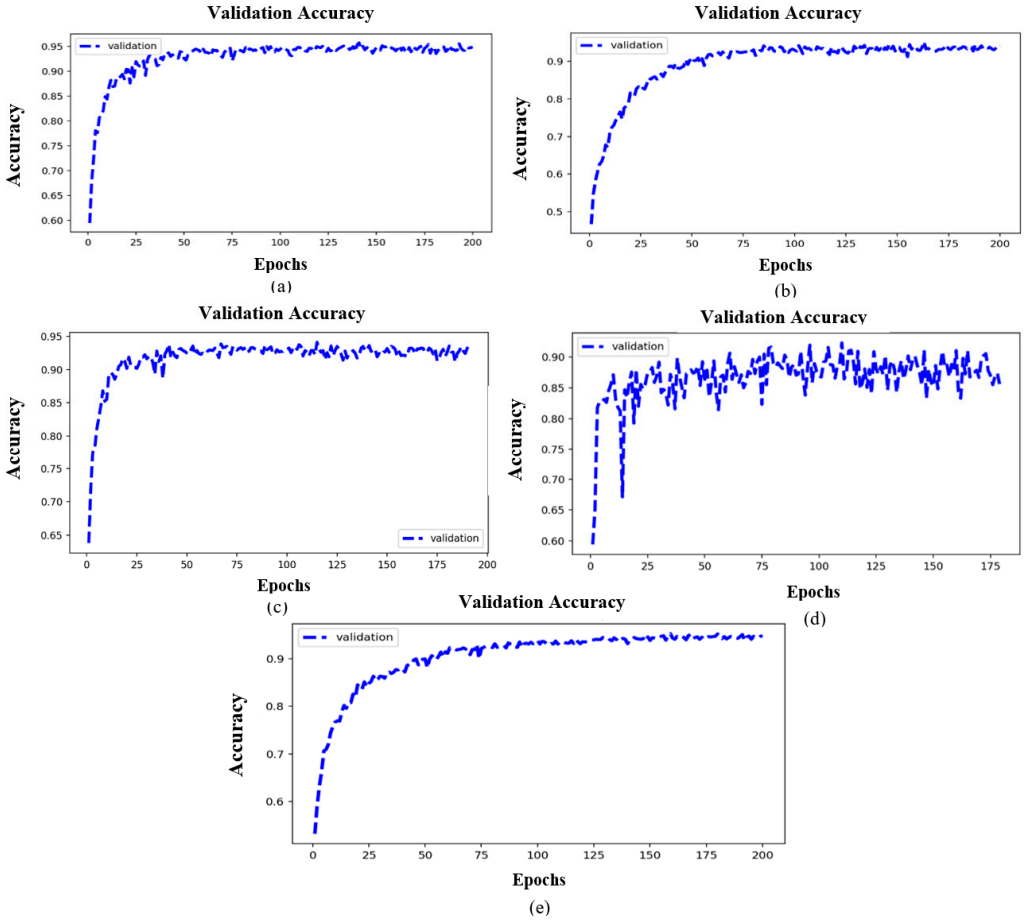


Figure 7. Comparisons of different settings (s = 1000, p = 15 × 15 × 11): (**a**) Adam, (**b**) SGD, (**c**) RMSprop, (**d**) Adadelta, (**e**) Adamax.

4.4. Classification Results

Figure 8 shows the ground truth and the classification output with the D-Net method by considering 70% of Section 1 as the training data. Figure 8 shows that vegetation, buildings, cars, asphalt road, sidewalks, billboards, powerlines, and poles were all segmented well, despite their various dimensions. The objects' spatial context in Figure 8 was well characterized due to the optimum voxelization and the proposed method's deep descriptor extraction capability. As shown in Figure 9, objects such as vegetation, cars, and sidewalks with rough shapes were also correctly classified, and objects such as poles, powerlines, and billboards were labeled with high precision. Almost all the points were consistent with the ground truth. However, the unclassified points were excluded due to the lack of separation in the training step.

4.5. Accuracy Assessment

To evaluate the results, the dataset was labeled manually and several accuracy assessment measures were evaluated, including precision (Pr), recall (R.), and F1-score (F1). These parameters were measured with true positive (TP), true negative (TN), false positive (FP), and false-negative (FN) from the confusion matrix according to equations in [37]. The quantitative results of the conducted experiments for the test samples of Section 1 are gathered in Figure 10. The results showed that the D-Net method classified all points

in the eight considered classes with precision, recall, and F1 values higher than 0.9. The input patch size of 15 × 15 × 11 was considered. The assessment results indicate that the D-Net method exhibited satisfactory classification results with high precision and recall for most classes.

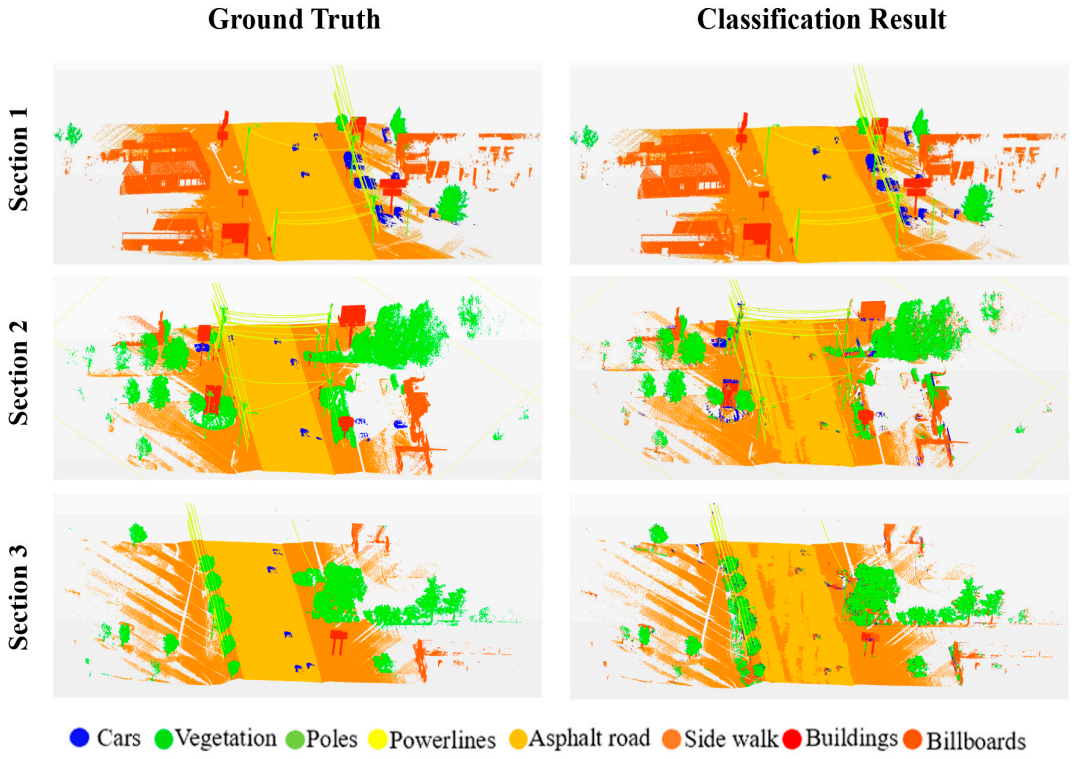


Figure 8. Ground truth data and classification results of Sections 1, 2, and 3.

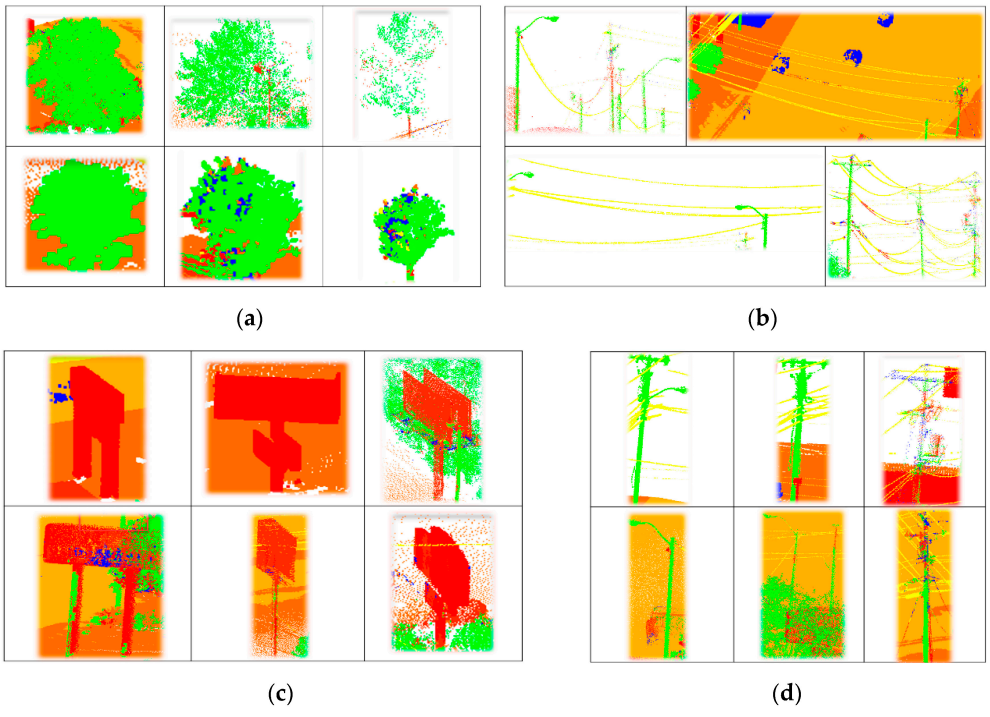


Figure 9. *Cont.*

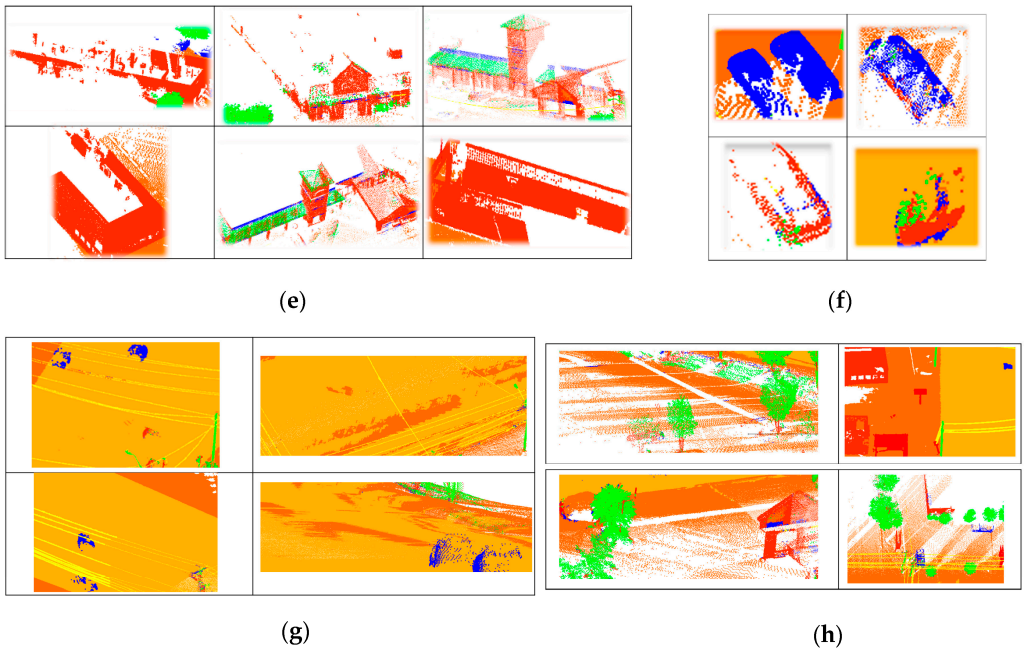


Figure 9. Part of the obtained outputs of the presented algorithm: (**a**) vegetation; (**b**) powerlines; (**c**) billboards; (**d**) poles; (**e**) building; (**f**) car; (**g**) asphalt road; (**h**) sidewalk.

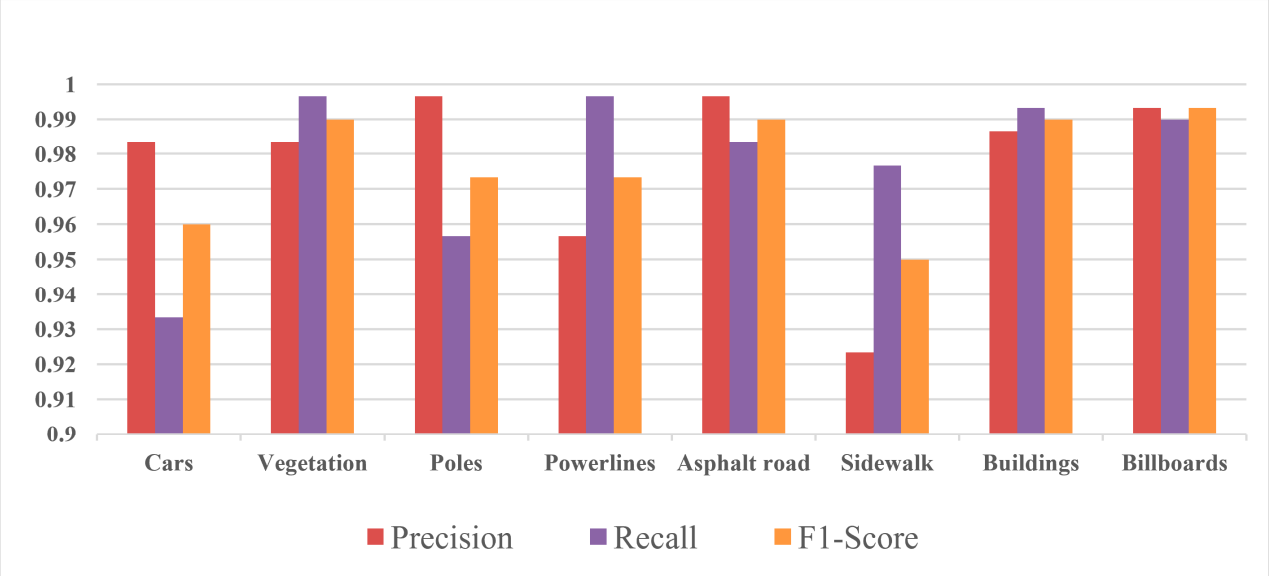


Figure 10. Evaluation of the D-Net method on the test samples of Section 1 for considered semantic classes with 15 × 15 × 11 input patch size.

The generalization of the D-Net method was evaluated by classifying the other two sections, which were not used in the training process. Table 7 shows the accuracy of the results of each section. The outputs indicate that vegetation, powerlines, and asphalt road classes were separated with high precision, recall, and F1-score values. Additionally, the D-Net method achieved acceptable performance in extracting poles, sidewalks, and billboard classes. The extraction and classification of buildings and cars in these sections were less accurate than in other classes.

Table 7. Evaluation of the D-Net method for Sections 2 and 3 of MLS point cloud classification with $15 \times 15 \times 11$.

Patch Sizes	Accuracy	Classes (%)							
		Cars	Vegetation	Poles	Powerlines	Asphalt Road	Sidewalk	Buildings	Billboards
Section 2	Precision	34.5	93.6	71.4	90.6	90.5	68.9	51.2	83.7
	Recall	43.8	69.3	72.2	91.3	93.2	67.4	77.3	60.2
	F1-score	38.6	79.6	71.8	90.9	91.8	68.1	61.6	70.0
Section 3	Precision	36.6	97.8	60.3	80.2	92.9	67.6	55.6	56.3
	Recall	43.2	74.6	85.3	87.3	91.6	73.3	82.7	60.4
	F1-score	39.6	84.6	70.7	83.6	92.2	70.3	66.5	58.3

5. Discussion

5.1. Dataset

This paper proposed a deep learning procedure using density information named D-Net to classify roadside objects in MLS data. This method evaluated the classification of challenging items ranging from lengthy power lines to tall trees and efficiently achieved a 98% accuracy. This high precision was obtained when both near and midrange objects in proximity to the MLS vehicle were classified correctly. It is noteworthy that objects further from the MLS system have a lower point density. This indicates that the algorithm may work adequately on point clouds collected by MLS systems with lower sampling rates and, more importantly, ALS LiDAR point clouds that are typically lower in density. Further, the algorithm does not need additional information such as intensity or trajectory. Thus, it should be suitable for classifying objects from 3D point cloud datasets acquired from other remote sensing technologies.

5.2. Network Structure

A comparison was made of the different numbers of convolution and hidden layers based on accuracy and computation speed to investigate the efficiency of the aggregating approach. Figure 11 shows the results of considering two comparative aspects: accuracy and timely processing. The figure shows that the D-Net method takes more than 20 to 30 s just by increasing one and two hidden layers. Additionally, this method requires 25 s to exploit four convolution layers. Concerning average accuracy, the proposed D-Net method achieved a high performance considering low hidden and convolution layers. Considering both efficiency and effectiveness, three convolutional and two hidden layers were chosen for implementation.

5.3. Comparison with Descriptor-Based Methods

Feature-based methods are frequently used for separating objects in urban areas. This section compares the acquired results from the D-Net framework with descriptor-based methods. This study implemented five popular machine learning techniques (k-NN, GNB, SVM, MLP, and RF) using the presented features in [37], and the outputs are demonstrated in Table 8. It should be noted that the primary data sizes in the training step for these five methods were the same as for the D-Net method for adequate evaluation. Figure 12 shows the visual outputs of the considered methods.

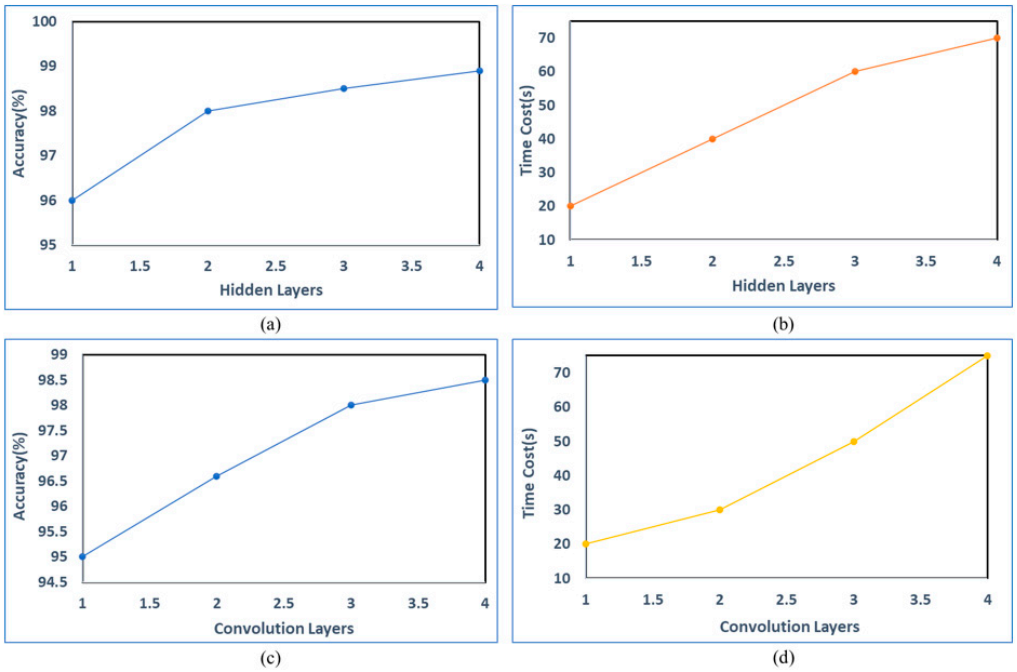
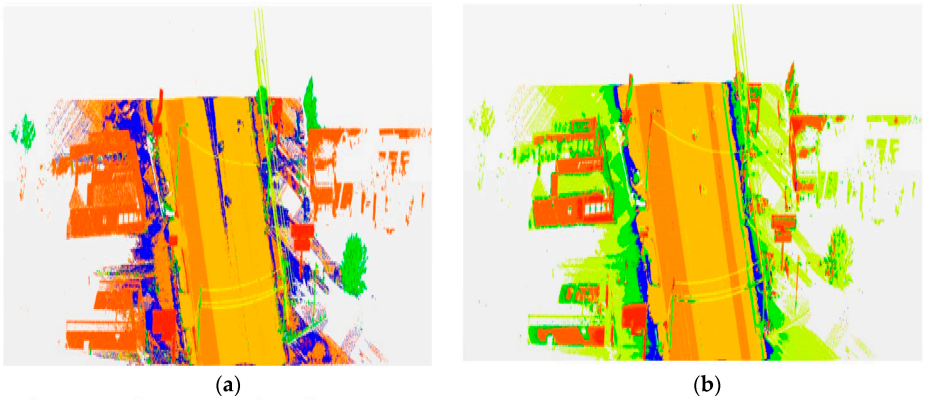


Figure 11. Evaluation of the implemented deep learning network with the number of layers: (**a**,**b**) accuracy and time cost assessment for FC layers, (**c**,**d**) accuracy and time cost assessment for Convolution Layers.

Table 8. Accuracy comparison between our proposed algorithm and other common deep neural network structures.

	Measures				
Methods	**mIOU**	**OA (%)**	**Pr. (%)**	**R. (%)**	**F1(%)**
PointNet++	47.2	90.8	91.5	91.9	92.8
PointConv	48.3	91.3	91.9	91.4	92.5
PointSeg	39.9	92.1	92.3	93.5	92.7
TGNet	58.5	91.4	92.32	92.1	91.3
MS-TGNet	61.2	93.3	94.69	93.2	92.9
RangeNet++	58.3	93	92.9	92.3	94.5
FPS-Net	69.1	97.6	**98.2**	96.66	**97.7**
D-Net	**69.7**	**98**	98.125	97.5	97.625



(**a**) (**b**)

Figure 12. *Cont.*

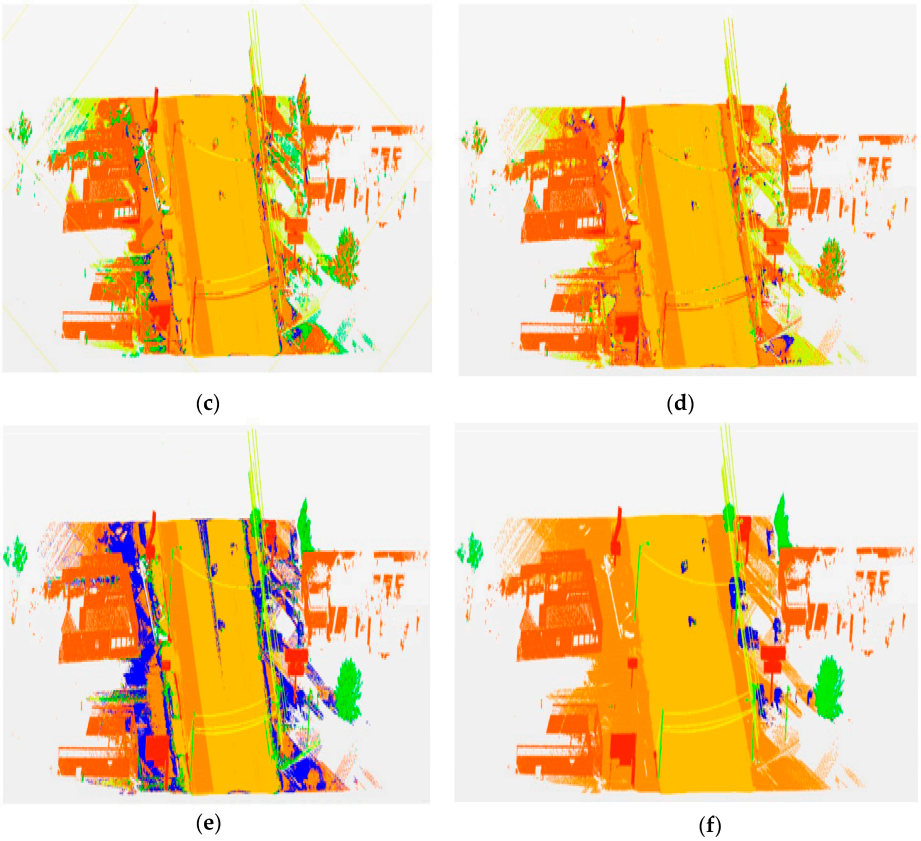


Figure 12. The outputs of the classification methods: (**a**) *k-NN*; (**b**) GNB; (**c**) SVM; (**d**) MLP; (**e**) RF; (**f**) D-Net.

Figure 13 compares the implemented descriptor-based and D-Net methods' accuracies. The figure shows that the D-Net method gave each measure the best average accuracy. The method achieved an accuracy of over 97%, while the highest-performing descriptor-based method was the RF, with the accuracy value ranging from 82% to 90%.

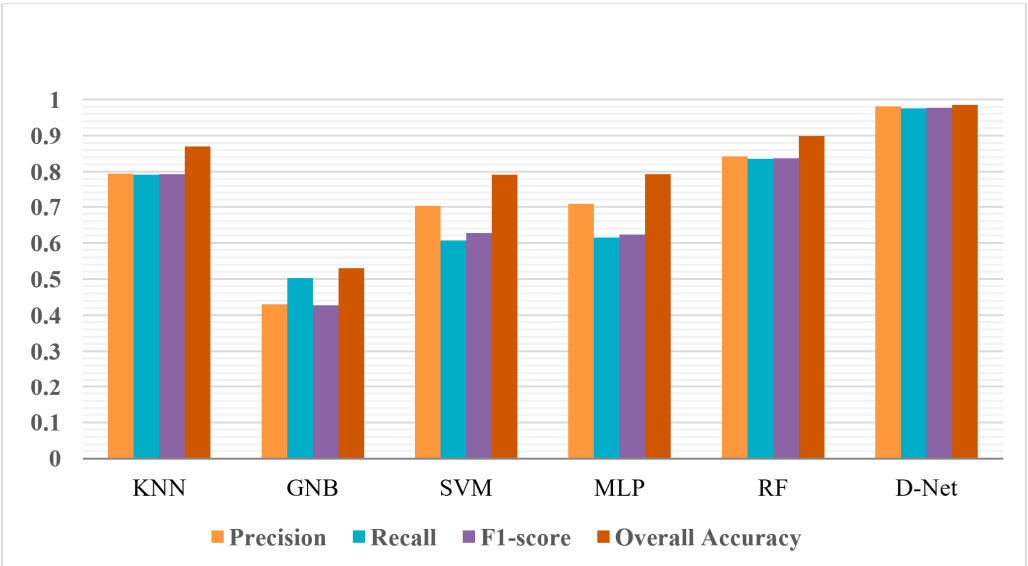


Figure 13. Accuracy comparison between the proposed D-net algorithm and common machine learning procedures.

5.4. Comparison with DL-Based Methods

The D-Net method was also compared with other DL-based methods. Seven widely-used DL networks were selected for performance evaluation (PointNet++ [13], PointConv [38], PointSeg [39], TGNet [19], MS-TGNet [40], RangeNet++ [41], FPS-Net [40]). Table 8 gives these results based on the mean intersection over union (mIoU) [33], F1-score, precision, recall, and OA. The results indicate that the D-Net algorithm obtains better mean IoU (69.7%) and OA (98%). It also outperforms other models in detecting asphalt roads, poles, sidewalks, and man-made objects. Among the evaluated DL-based methods, the MS-TGNet, RangeNet++, and FPS-Net acquired OA and mIOU over 93% and 58%, respectively. The TGNet performed modestly and achieved an OA of 87.59%. On the other hand, PointNet had the lowest performance in object classification, at around 90%. Noticeably, our D-Net algorithm accomplished encouraging outputs on the considered regions.

6. Conclusions

This study proposed a novel method, i.e., the D-Net framework, for classifying MLS point clouds in an urban area based on density information. This method applied the pre-processing step to the raw LiDAR point cloud to remove the noisy points. Furthermore, a voxelization stage converted the irregular LiDAR point cloud into a regular format. Then, the density of each voxel was calculated, and after extracting deep descriptors the class of each voxel was detected using the trained D-Net framework. Finally, all points received the same label as their corresponding voxel.

The proposed D-Net framework was evaluated on an MLS dataset in an urban area with a length of 182 m. It included numerous objects such as tall buildings, vehicles, vegetation, pole-shaped objects, roadway infrastructure, sidewalk, powerlines, roads, and billboards. Reported precision, recall, and F1-scores of over 92% and OA of over 98% were achieved. The method was highly accurate for all classes. However, it struggled with complex structures attached to other objects due to ambiguous category definitions.

The study's results show that using the D-Net method for urban area point cloud classification is feasible with promising results. Obtaining sufficient data for each class in a dataset may be problematic when considering that a certain percentage is needed for training. The investigation of the D-Net method for other datasets, the size of training data, and the CNN structure may be studied in the future. The direct use of a point cloud for importing into the three-dimensional CNN is also suggested.

Author Contributions: Conceptualization, M.Z. and H.R.; methodology, M.Z., B.H. and H.R.; software, M.Z., B.H. and D.S.; validation, M.Z., H.R. and B.H.; formal analysis, H.R.; investigation, D.S., M.Z., B.H., W.A.S. and S.H; resources, M.Z. and H.R.; data curation, M.Z.; writing—original draft preparation, M.Z.; writing—review and editing, D.S., B.H., H.R., W.A.S. and S.H.; visualization, M.Z. and H.R.; supervision, H.R. and S.H.; project administration, H.R. and S.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data sharing is not applicable to this paper.

Acknowledgments: We would like to thank Alireza Shams for providing us with the data used in this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, L.; Meng, W.; Xi, R.; Zhang, Y.; Ma, C.; Lu, L.; Zhang, X. 3D Point Cloud Analysis and Classification in Large-Scale Scene Based on Deep Learning. *IEEE Access* **2019**, *7*, 55649–55658. [CrossRef]
2. Lehtomäki, M.; Jaakkola, A.; Hyyppä, J.; Lampinen, J.; Kaartinen, H.; Kukko, A.; Puttonen, E.; Hyyppä, H. Object Classification and Recognition from Mobile Laser Scanning Point Clouds in a Road Environment. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 1226–1239. [CrossRef]

3. Shokri, D.; Rastiveis, H.; Sarasua, W.A.; Shams, A.; Homayouni, S. A Robust and Efficient Method for Power Lines Extraction from Mobile LiDAR Point Clouds. *PFG J. Photogramm. Remote Sens. Geoinform. Sci.* **2021**, *89*, 209–232. [CrossRef]
4. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*; IEEE: New York, NY, USA, 2017; pp. 652–660.
5. Li, X.; Wang, L.; Wang, M.; Wen, C.; Fang, Y. DANCE-NET: Density-aware convolution networks with context encoding for airborne LiDAR point cloud classification. *ISPRS J. Photogramm. Remote Sens.* **2020**, *166*, 128–139. [CrossRef]
6. Han, X.; Dong, Z.; Yang, B. A Point-Based Deep Learning Network for Semantic Segmentation of MLS Point Clouds. *ISPRS J. Photogramm. Remote Sens.* **2021**, *175*, 199–214. [CrossRef]
7. Yang, B.; Dong, Z.; Zhao, G.; Dai, W. Hierarchical Extraction of Urban Objects from Mobile Laser Scanning Data. *ISPRS J. Photogramm. Remote Sens.* **2015**, *99*, 45–57. [CrossRef]
8. Sun, Z.; Xu, Y.; Hoegner, L.; Stilla, U. Classification of mls point clouds in urban scenes using detrended geometric features from supervoxel-based local contexts. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2018**, *IV-2*, 271–278. [CrossRef]
9. Günen, M.A. Adaptive Neighborhood Size and Effective Geometric Features Selection for 3D Scattered Point Cloud Classification. *Appl. Soft Comput.* **2022**, *115*, 108196. [CrossRef]
10. Zolanvari, S.I.; Laefer, D.F.; Natanzi, A.S. Three-dimensional building façade segmentation and opening area detection from point clouds. *ISPRS J. Photogramm. Remote Sens.* **2018**, *143*, 134–149. [CrossRef]
11. Huang, J.; You, S. Point Cloud Labeling Using 3D Convolutional Neural Network. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; pp. 2670–2675.
12. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Curran Associates, Inc.: New York, NY, USA, 2017; Volume 30.
13. Yao, X.; Guo, J.; Hu, J.; Cao, Q. Using Deep Learning in Semantic Classification for Point Cloud Data. *IEEE Access* **2019**, *7*, 37121–37130. [CrossRef]
14. Wen, C.; Yang, L.; Li, X.; Peng, L.; Chi, T. Directionally Constrained Fully Convolutional Neural Network for Airborne LiDAR Point Cloud Classification. *ISPRS J. Photogramm. Remote Sens.* **2020**, *162*, 50–62. [CrossRef]
15. Classification of Point Cloud for Road Scene Understanding with Multiscale Voxel Deep Network—Archive Ouverte HAL. Available online: https://hal.science/hal-01763469/ (accessed on 11 April 2023).
16. Wang, L.; Huang, Y.; Shan, J.; He, L. MSNet: Multi-Scale Convolutional Network for Point Cloud Classification. *Remote Sens.* **2018**, *10*, 612. [CrossRef]
17. Wang, Z.; Zhang, L.; Zhang, L.; Li, R.; Zheng, Y.; Zhu, Z. A Deep Neural Network With Spatial Pooling (DNNSP) for 3-D Point Cloud Classification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 4594–4604. [CrossRef]
18. Li, Y.; Ma, L.; Zhong, Z.; Cao, D.; Li, J. TGNet: Geometric Graph CNN on 3-D Point Cloud Segmentation. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 3588–3600. [CrossRef]
19. Li, W.; Wang, F.-D.; Xia, G.-S. A Geometry-Attentional Network for ALS Point Cloud Classification. *ISPRS J. Photogramm. Remote Sens.* **2020**, *164*, 26–40. [CrossRef]
20. Boulch, A. ConvPoint: Continuous Convolutions for Point Cloud Processing. *Comput. Graph.* **2020**, *88*, 24–34. [CrossRef]
21. Song, W.; Zhang, L.; Tian, Y.; Fong, S.; Liu, J.; Gozho, A. CNN-Based 3D Object Classification Using Hough Space of LiDAR Point Clouds. *Hum. Cent. Comput. Inf. Sci.* **2020**, *10*, 19. [CrossRef]
22. Geng, X.; Ji, S.; Lu, M.; Zhao, L. Multi-Scale Attentive Aggregation for LiDAR Point Cloud Segmentation. *Remote Sens.* **2021**, *13*, 691. [CrossRef]
23. Hoang, L.; Lee, S.-H.; Lee, E.-J.; Kwon, K.-R. GSV-NET: A Multi-Modal Deep Learning Network for 3D Point Cloud Classification. *Appl. Sci.* **2022**, *12*, 483. [CrossRef]
24. Aijazi, A.K.; Checchin, P.; Trassoudaine, L. Segmentation Based Classification of 3D Urban Point Clouds: A Super-Voxel Based Approach with Evaluation. *Remote Sens.* **2013**, *5*, 1624–1650. [CrossRef]
25. Weinmann, M.; Jutzi, B.; Mallet, C. Feature Relevance Assessment for the Semantic Interpretation of 3D Point Cloud Data. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inform. Sci.* **2013**, *II-5/W2*, 313–318. [CrossRef]
26. Yang, Z.; Jiang, W.; Xu, B.; Zhu, Q.; Jiang, S.; Huang, W. A Convolutional Neural Network-Based 3D Semantic Labeling Method for ALS Point Clouds. *Remote Sens.* **2017**, *9*, 936. [CrossRef]
27. Boulch, A.; Guerry, J.; Le Saux, B.; Audebert, N. SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks. *Comput. Graph.* **2018**, *71*, 189–198. [CrossRef]
28. Shukor, S.A.A.; Rushforth, E.J. Adapting Histogram for Automatic Noise Data Removal in Building Interior Point Cloud Data. *AIP Conf. Proc.* **2015**, *1660*, 070074. [CrossRef]
29. Griffiths, D.; Boehm, J. A Review on Deep Learning Techniques for 3D Sensed Data Classification. *Remote Sens.* **2019**, *11*, 1499. [CrossRef]
30. Deep Learning—Ian Goodfellow, Yoshua Bengio, Aaron Courville—Google Books. Available online: https://books.google.com/books?hl=en&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=Goodfellow,+I.%3B+Bengio,+Y.%3B+Courville,+A.+Deep+learning%3B+MIT+press,+Cambridge,+Massachusetts,+London,+England:+2016&ots=MNU-bvmEQT&sig=0U23Gi-E3iJ82UHyrchZqKwto7s#v=onepage&q&f=false (accessed on 11 April 2023).

31. Wichrowska, O.; Maheswaranathan, N.; Hoffman, M.W.; Colmenarejo, S.G.; Denil, M.; Freitas, N.; Sohl-Dickstein, J. Learned Optimizers That Scale and Generalize. In Proceedings of the 34th International Conference on Machine Learning; PMLR, Sydney, NSW, Australia, 6–11 August 2017; pp. 3751–3760.
32. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980.
33. PointPAVGG: An Incremental Algorithm for Extraction of Points' Positional Feature Using VGG on Point Clouds. SpringerLink. Available online: https://link.springer.com/chapter/10.1007/978-3-030-84529-2_60 (accessed on 11 April 2023).
34. Girija, S.S. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv* **2016**, arXiv:1603.04467v2.
35. Zhang, M.; Lucas, J.; Ba, J.; Hinton, G.E. Lookahead Optimizer: K Steps Forward, 1 Step Back. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates, Inc.: New York, NY, USA, 2019; Volume 32.
36. Zaboli, M.; Rastiveis, H.; Shams, A.; Hosseiny, B.; Sarasua, W. Classification Of Mobile Terrestrial Lidar Point Cloud in Urban Area Using Local Descriptors. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.* **2019**, *XLII-4/W18*, 1117–1122. [CrossRef]
37. Wu, W.; Qi, Z.; Fuxin, L. PointConv: Deep Convolutional Networks on 3D Point Clouds. *arXiv* **2019**, arXiv:1811.07246v3.
38. Wang, Y.; Shi, T.; Yun, P.; Tai, L.; Liu, M. PointSeg: Real-Time Semantic Segmentation Based on 3D LiDAR Point Cloud. *arXiv* **2018**, arXiv:1807.06288.
39. Tan, W.; Qin, N.; Ma, L.; Li, Y.; Du, J.; Cai, G.; Yang, K.; Li, J. Toronto-3D: A Large-Scale Mobile LiDAR Dataset for Semantic Segmentation of Urban Roadways. *arXiv* **2020**, arXiv:2003.08284v3.
40. Milioto, A.; Vizzo, I.; Behley, J.; Stachniss, C. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), The Venetian Macao, Macau, 4–8 November 2019.
41. Xiao, A.; Yang, X.; Lu, S.; Guan, D.; Huang, J. FPS-Net: A convolutional fusion network for large-scale LiDAR point cloud segmentation. *ISPRS J. Photogramm. Remote Sens.* **2021**, *176*, 237–249. [CrossRef]