GROUND POINT FILTERING FROM AIRBORNE LIDAR POINT CLOUDS USING DEEP LEARNING: A PRELIMINARY STUDY

Eric Janssens-Coron[1,2]*, Eric Guilbert[1]

[1] Dept. of Geomatics Sciences, Laval University, Québec, G1V 0A6 (QC) Canada
[2] Xeos Imaging Inc. Québec, G1P 4R1 (QC) Canada
eric.janssens-coron.1@ulaval.ca, eric.guilbert@scg.ulaval.ca

Commission IV, WG IV/1

KEY WORDS: lidar, deep learning, ground point, classification

ABSTRACT:

Airborne lidar data is commonly used to generate point clouds over large areas. These points can be classified into different categories such as ground, building, vegetation, etc. The first step for this is to separate ground points from non-ground points. Existing methods rely mainly on TIN densification but there performance varies with the type of terrain and relies on the user's experience who adjusts parameters accordingly. An alternative may be on the use of a deep learning approach that would limit user's intervention. Hence, in this paper, we assess a deep learning architecture, PointNet, that applies directly to point clouds. Our preliminary results show mitigating classification rates and further investigation is required to properly train the system and improve the robustness, showing issues with the choices we made in the preprocessing. Nonetheless, our analysis suggests that it is necessary to enrich the architecture of the network to integrate the notion of neighbourhood at different scales in order to increase the accuracy and the robustness of the treatment as well as its capacity to treat data from different geographical contexts.

1. INTRODUCTION

Airborne lidar data is now a common technology used to generate digital surface models over large areas with a high density of points. These points can be classified into different categories such as ground, building, vegetation, etc to produce 3D models. The first step in this process is the generation of a digital terrain model supporting 3D objects.

The DTM is built from ground points obtained from the whole point cloud. Hence the quality of the DTM directly depends on the quality of the algorithm selecting the points. Oldest approaches rely on geometrical and statistical indicators but ground filtering methods perform differently on different types of terrain. Methods based on the construction and the densification of a triangulated irregular network (Axelsson, 2000) appear to be the most robust to all environments and are the most commonly used. Overall, these methods can give good classification rates in benchmarks but, considering the large amounts of data to process, researchers looked for different approaches based on machine learning.

Machine learning approaches can integrate and handle more patterns, leading to more sophisticated and more efficient classifiers. Although they can provide better results, their improvements remain marginal. The reason is that patterns still rely on statistical and geometrical indicators, facing the same difficulties regarding the different types of surface.

In these last few years, deep learning methods became popular approaches for semantic segmentation of datasets. Most methods apply to images. These methods can also apply to point clouds but many rely on voxelisation: space is divided in voxels and each voxel is assigned a value according to the points it contains. Hence such methods do not work on a point cloud but on an approximation. They also demand a lot of resources since they require a complete voxelisation of the 3D space, leading to many empty voxels.

Recently, a new model, PointNet, was developed that can handle directly point clouds (Qi et al., 2017a). The authors applied PointNet to classify or segment 3D objects or indoor scenes. In these examples, point clouds remain more or less structured and can form regular clusters of points and geometrical shapes. On the opposite, airborne lidar points are highly unstructured and, in natural environments, present little regularity. Hence, this paper introduces preliminary results in classifying airborne lidar point clouds. Our objective is to assess PointNet ability in selecting ground points. Since existing methods perform differently in different environments, we assess PointNet in both urban and forest environments.

The remaining of the paper is organised as follows: the next section presents existing approaches on ground point selection. Section 3 presents PointNet and the set of experiments that were conducted. Section 4 presents the results with a discussion on the limitations of PointNet. Last section presents concluding remarks and directions for future works.

2. EXISTING WORKS

2.1 Geometrical approaches

The first methods for ground point selection were based on geometrical indicators. They would look at indicators such as the elevation (lowest points are most likely on the ground), slope or elevation difference (neighbouring points on and above grounds may show a rapid change of elevation) and curvature. (Meng et al., 2010) provides a classification of these methods.

Among these methods, the most commonly used are based on TIN densification (Axelsson, 2000). They consist first in selecting the lowest point in each cell of a grid covering the area of in-

*Corresponding author

terest. These points form an initial triangulation and other points close enough to the TIN are added iteratively.

Interpolation methods and contour methods (based on active contours) build a surface by interpolation or approximation of the lidar points and iterate to reduce the residue. Interpolation can be linear or based on splines. The method can be refined to consider different scales: (Evans and Hudak, 2007) makes use of thin plate splines to compute a curvature at different scales.

Segmentation methods cluster points into regions with homogeneous slope or elevation. These methods are mainly used in urban environments where regions are easier to characterise. Morphological methods apply opening operations (a combination of erosion and dilation operations) on the points. Directional scanning filters process points along a scan line, looking at only one direction. Methods based on statistical analysis were also developed (Chen et al., 2017). They mainly rely on the computation of skewness.

Among these approaches, some, such as TIN densification, directly identify ground points while others, mainly belonging to interpolation and morphological methods, generating a raster or grid DTM and do not always yield a classification. Overall, they do not perform equally according to different environments. While best scores are obtained in urban areas, they are usually less efficient in natural environments such as: rough terrain with steep slopes and terrain with low vegetation where the difference with the ground is difficult to make and forested areas where few points reach the ground (Meng et al., 2010). However, citing (Chen et al., 2017), "filters that work well in forest (urban) areas may struggle in urban (forest) areas."

Methods in the literature are usually benchmarked on ISPRS reference datasets. These datasets are raw lidar point clouds obtained in different environments. Overall, methods that give the best results are based on TIN densification, interpolation and morphology. However, details of the experiments and parameter settings are not always given and results for a same method on a same dataset can greatly vary. For example, for progressive triangulation on ISPRS dataset 1, (Bigdeli et al., 2018) found an error of 10.21% while (Hui et al., 2019) obtained an error of 28.21%. Indeed, results are sensitive to parameter settings and parameters are usually adjusted according to different variables such as the point density and the type of terrain.

2.2 Machine learning approaches

While ISPRS datasets are low density point clouds (below 1 point per square metre) over a few square kilometres, current ALS systems can acquire point clouds at a much higher density, above 20 points per square metre on much larger areas. Hence, in the last decade, some researchers explored machine learning approaches to define new descriptors and automatically adjust the parameters. Machine learning is already commonly used for semantic classification of points. Much of this work applies to indoor environment and terrestrial laser scanning for classifying urban features or buildings. Among the methods developed for aerial lidar data, some resample lidar points into a regular grid (Lodha et al., 2006) or into voxels (Wang et al., 2018) or make use of an orthoimage providing extra information (Chehata et al., 2009).

Machine learning relies on the definition of geometrical features on each point computed in a small neighbourhood such as the slope, the planarity or the difference of elevation. Some methods also include the intensity. On top of that, machine learning approaches also apply clustering approaches to define planar segments and further features on these planar segments (Lu et al., 2009). Most common techniques are conditional random field (Lu et al., 2009, Niemeyer et al., 2012), support vector machines (Mewhort, 2013) and random forest (Ni et al., 2017) but also adaBoost and bagging (Nourzad and Pradhan, 2014).

In most cases, these techniques are applied to urban environment with the objective of classifying ground, vegetation and buildings. Hence, point clusters are mainly defined by plane regions obtained by graph-based segmentation, growing regions or RANSAC. (Lu et al., 2009) and (Ni et al., 2017) also consider natural environments. (Lu et al., 2009) define disk-based feature regions while (Ni et al., 2017) define rough surfaces and scattered points on which features are also computed.

Overall, machine learning brings an improvement to previous geometrical approaches since features defined on points and clusters combine the different characteristics computed with earlier approaches. Performance of the method depends on the choice of features and most works focus on urban areas. In the remaining of the paper, we assess a deep learning approach which would have the ability to learn features during the training process.

3. AIRBORNE LIDAR POINT CLOUD CLASSIFICATION WITH DEEP LEARNING

3.1 The PointNet architecture

In recent years, new classification techniques based on deep learning have emerged for recognition of discrete 3D objects and terrestrial / mobile lidar data. The first methods were inspired by methods used on 2D images and require a voxelisation of space (Maturana and Scherer, 2015). Voxel values are defined according to the points found in each voxel. As such, input data are downgraded and, since points do not occupy the whole space, this approach leads to generating many empty voxels adding extra computation load. An alternative was proposed by (Qi et al., 2017a) with PointNet.

PointNet is a convolutional neural network (CNN). It uses convolutions to generate new features for each point and highlight patterns in the data. The main specificity of PointNet is to deal directly with each point of the point cloud without resorting to voxelisation. The second interest of PointNet is to propose solutions to two fundamental problems related to point clouds: the absence of ordering between the points and the dependence on rigid transformations. A third feature of PointNet is the division of the point cloud into blocks along a regular grid. These blocks are the expression by PointNet of the notion of neighbourhood. Although PointNet divides the cloud of points into blocks, this division is to be distinguished from that of point cloud voxelisation approaches that use it to generalise the information in each of these blocks.

The PointNet architecture consists of 3 steps (Figure 1). The first step takes a set of points defined by their xyz coordinates and computes local features to each points. Features are added through a multilayer perceptron (MLP). The first step can also include mini-nets of neurons (net transforms or T-nets) that make the points independent from rigid transformations (translations and rotations). At the end of this step, each point is described by a vector of 64 features.

The second step computes global features over all the points. It uses for this a multilayer perceptron followed by a max pooling. The max pooling makes the point cloud invariant to point order. The information is added to each of the local features in the block forming a total of 1088 features. The final step of PointNet effectively performs the classification, producing a tensor of probabilities of belonging to each class for each point.

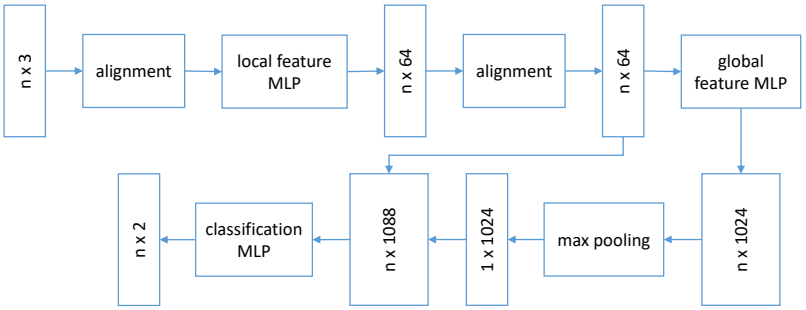

Figure 1: PointNet architecture. MLP stands for multi layer perceptron.Alignment blocks are optional

3.2 Aerial lidar point classification

An aerial LiDAR data acquisition project does not necessarily or even rarely includes images. Therefore, the main information available is the xyz coordinates of the points to which the intensity can be added, even though this characteristic is still difficult to apprehend. The areas covered in a survey can reach tens of thousands of square kilometres, yielding huge volumes of data (hundreds of GB for billions of points).

To facilitate storage and because of current software limitations, point clouds are usually cut into tiles of reasonable size, in our case, 1 km^2 tiles. The amount of points per tile depends on the density of the point cloud. This density varies greatly even within a project depending on the sensor but also on variations in the speed of the aircraft, morphology of the territory, etc. Swath overlaps also artificially change the density. For example, in our case studies, the advertised density is conservative at 4 points/m^2 but the actual density in a tile can vary from 4 to 50 pts/m^2 since a pulse can have several returns. Hence, the number of points per tile can vary between 10 and 22 million points.

We used two datasets:

- A point cloud of eastern Montreal collected by Xeos Imaging Inc. in 2015 openly available online[1] with a calculated average density of 16pts/m^2 divided in 12 tiles. Coordinates are in MTM7. In this dataset, 43.78% of the points are classified as ground, meaning that the ratio of ground points is rather balanced.
- A point cloud located in a dense forest area in southern Quebec province with an advertised density of 4pts/m^2, representing 195 million points provided by Xeos Imaging, Inc. Coordinates are in MTM8 grid system. The point ratio is very unbalanced with 79.11% as ground.

3.3 Data preparation

PointNet code is available online[2]. The code is in Python and runs with TensorFlow. PointNet takes in input a set of points arranged in batches. These batches are further divided in blocks. Each block contains 4096 points.

Batches were prepared by dividing each tile in blocks and then by grouping blocks in batches. Three approaches were considered to define the blocks. The first approach consists in simply taking 4096 consecutive points of the tile as they are recorded in the file. It means that there is no spatial arrangement between the points and that points in a block are not necessarily close to each other.

[1] http://donnees.ville.montreal.qc.ca/dataset/lidar-aerien-2015
[2] https://github.com/charlesq34/pointnet

The second and the third approach consist in partitioning the tile in blocks of approximately 4096 points. Block size is computed based on the point density, e.g. for a density of 16 pts/m^2, each block must be 16 m side. This represents 3969 blocks for one tile. Since the density varies inside a tile, blocks do not contain exactly 4096 points. In the second approach, points are duplicated or deleted randomly to get the right number. In the third approach, we take the first 4096 points and if some are missing, the first ones are duplicated. While there is no explicit spatial relations between the points, they are still arranged so that points in a block are close to each other.

Points in input can be defined by their xyz coordinates only or by adding other features computed during the preparation stage. Hence, we considered the three coordinates x, y, z and the intensity I. Coordinates can be simply translated to a common origin within the tile or normalised by using maximum and minimum coordinates to express all coordinates in a block on an interval $[0, 1]$. In order to avoid excessive deformations due to height variations between tiles, normalisation is done over the whole tile. Normalisation of the intensity was done by dividing each value by the maximum possible intensity on the whole dataset in order to bring it below 1.

The urban point cloud is composed of 12 tiles. Among these, 9 were used for the learning phase and 3 for the evaluation phase. The forest point cloud contains 20 tiles. We used 12 tiles for the training phase and 8 for the evaluation phase. Training was done independently on each dataset.

4. RESULTS AND DISCUSSION

The configuration used to perform our tests is composed of a 7th generation 3.8GHz I7 processor, 64 GB of RAM and a Quadro P4000 graphics card. The processing times with this configuration are about 16.5 hours for the Vanilla version of PointNet (without the T-nets) and 30.5 hours for the version with the T-nets.

The following hyperparameters were considered: the number of epochs, the batch size and the learning rate. While a batch size of 32 is more commonly found in the literature, we took a batch size of 16 mainly for technical reasons. Tests have been conducted on the learning rate value and best results were obtained for 0.001. Nonetheless, at this stage, We did not observe big variations with other values. In all cases, accuracy converge in less than 75 epochs. Hence all the tests were performed with this configuration.

We present results obtained for several tests assessing the influence on the number of features in input and the definition of blocks in each tiles in Table 1. Last column measures the total

Test	Attributes	Returns	Block	T-nets	Accuracy
1	x_n, y_n, z_n, I_n	Last	First 4096	No	76.96%
2	$x_t, y_t, z_t, x_n, y_n, z_n, I_n$	Last	No partition	No	55.72%
3	$x_t, y_t, z_t, x_n, y_n, z_n, I_n$, R#, NbR	All	First 4096	No	81.4%
4	$x_t, y_t, z_t, x_n, y_n, z_n, I_n$, R#, NbR	All	Random	No	78.01%
5	x_n, y_n, z_n, I_n	Last	Random	No	76.15%
6	$x_t, y_t, z_t, x_n, y_n, z_n, I_n$	Last	First 4096	No	75.65%
7	x_n, y_n, z_n, I_n	Last	First 4096	Yes	77.99%

Table 1: Results obtained on the urban dataset. Column block and alignment indicate how blocks were built and whether alignment using the T-Nets was done. Point selection indicates how points were selected to build blocks.

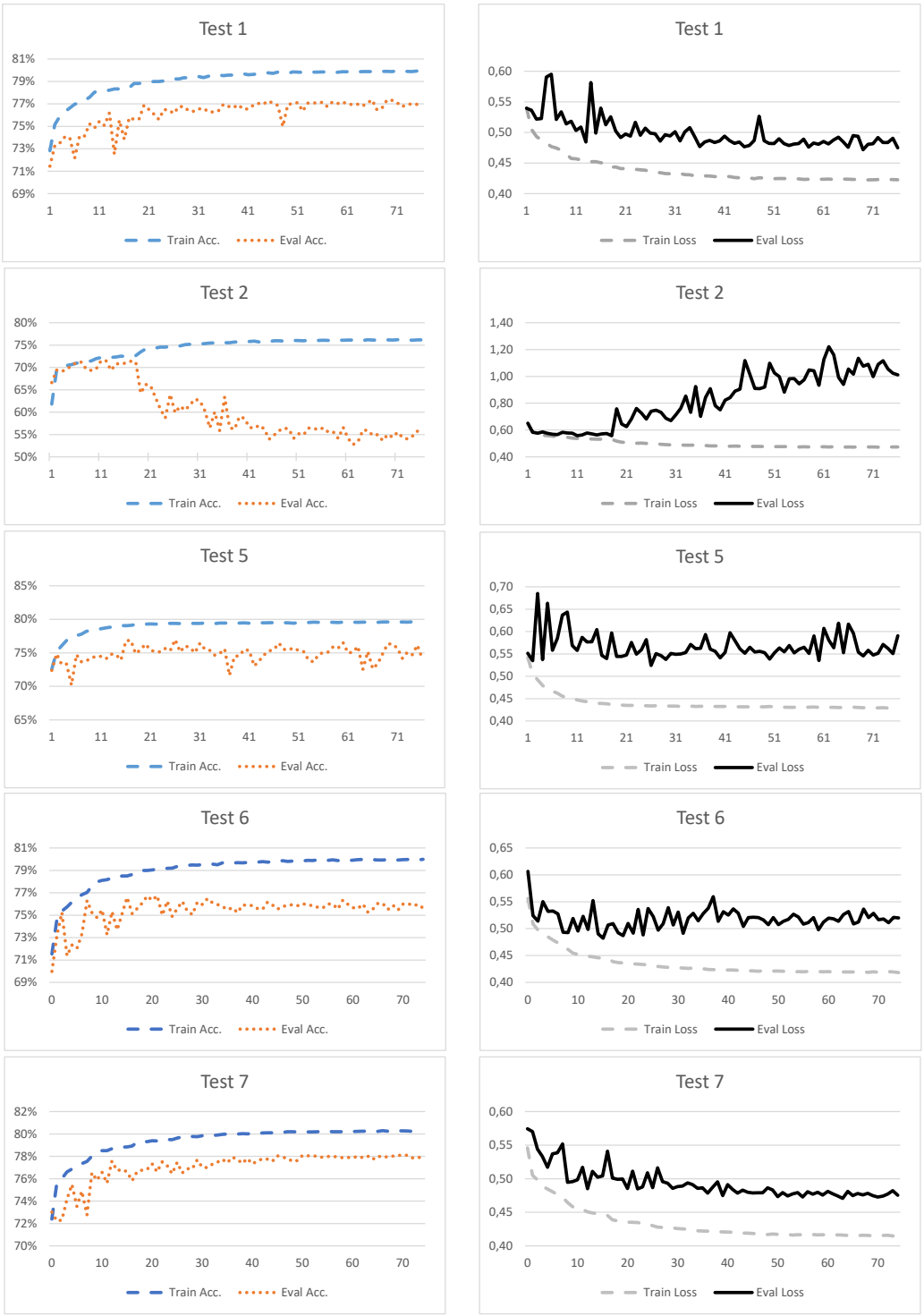

Figure 2: Accuracy and loss for training and evaluation of the urban point cloud with different settings

accuracy defined by the number of points correctly classified divided by the total number of points.

We also present the learning curves and loss curves on Figure 2. We represent for each iteration the accuracy and loss computed on training data and on evaluation data. The loss is used to measure the inconsistency between a predicted value and the actual label. It is a non-negative value, such that the robustness of the model increases as the loss decreases. We used the sigmoid cross-entropy method as defined in PointNet. It is commonly used for multi-label classification since it is independent for each label, meaning that the loss computed for every class is not affected by other component values.

All tests included normalised coordinates and intensity. Tests were first conducted on the urban dataset. They were done using both last returns only and all returns since both can be done in urban areas. In the second case, we added the number of returns and the number of the return to each point. We also considered both translated coordinates and normalised coordinates as done in existing tests for PointNet conducted in indoor environments (Qi et al., 2017a).

At first sight, we can notice significant differences in the prediction accuracy between tests with a difference of 23.31 points of percentage between the highest and lowest classifications. It appears immediately that the method without partitioning (test 2) is not appropriate. As seen on Figure 2, the network is not able to learn properly. Indeed, in test 2, points are arranged along swathes and points in the same tile can come from different swathes. While classification is supposed not to be affected by point ordering, points still need to be spatially organised to compute significant global features. Completing blocks with points taken randomly also yields lower results than taking the first 4096 as shown by tests 1 and 5 and tests 3 and 4 where the difference is around 4 points.

We note that best results were obtained when considering all returns (tests 3 and 4). However, it does not lead to better classifying points on the ground since all points that are not last returns are correctly classified as non ground. Increasing the number of attributes in input points did not have much impact and results in test 6 are similar, with even slightly inferior results.

Finally, we compared the classification with and without the T-nets (test 7). As mentioned by (Qi et al., 2017a), T-nets are used to align attributes to make them invariant to affine transformations, having a more regular learning curve. The use of T-nets increased the classification of 1 point but significantly increasing computation time. It also slightly smooth the training and loss curves (Figure 2). Overall, the accuracy obtained on the data is slightly lower than that obtained by (Qi et al., 2017a) when segmenting indoor scenes where the average accuracy was of 78.62%.

When studying the classification produced by PointNet, we see that the network tends to commit many points that were not classified on the ground in the reference data. As shown in Figure 3, most points located close to the ground were classified as ground. We see two issues for this: first, referenced data were classified in order to produce a DTM with the objective of minimising commissions. This may introduce a bias in the learning process. However, the network still tends to classify too many points on the ground, relying too much of the elevation attribute. In Figure 3 on the right, some points above the ground have been misclassified while the reference was correct.

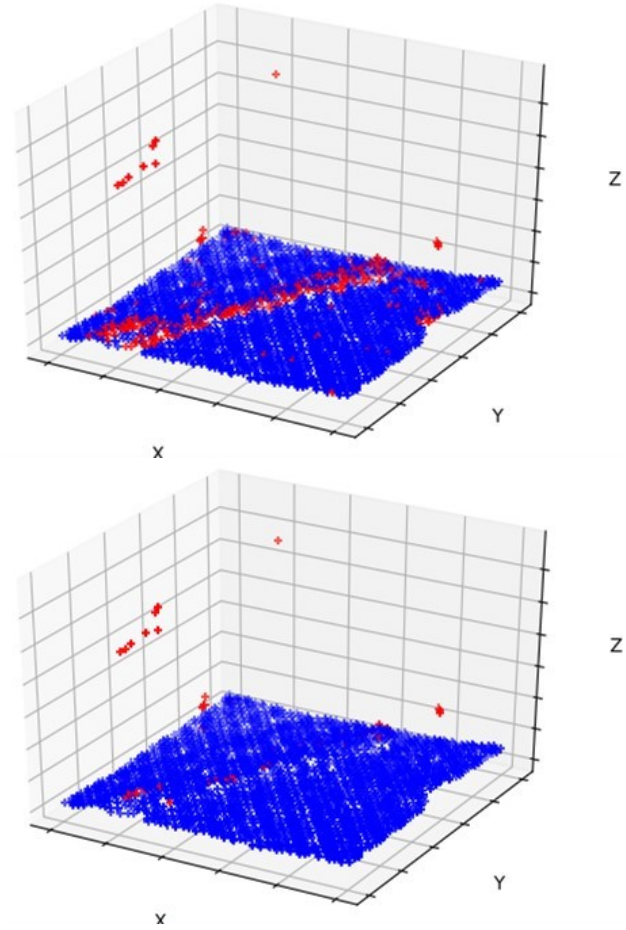

Figure 3: Comparison between reference points (above) and classified points (below). Blue represents ground points

Test	Attributes	Alignment	Accuracy
8	x_n, y_n, z_n, I_n	No	82.76%
9	x_n, y_n, z_n, I_n	Yes	82.01%

Table 2: Results obtained on the forest dataset. Only last returns were considered.

PointNet was also executed on a second dataset in a forest environment. Results are presented in Table 2 with learning and evaluation curves in Figure 4. Blocks were defined with the third approach since it yields better classification. Accuracy seems much higher than in urban area however, curves in Figure 4 clearly show that the algorithm was not able to learn from the data. It starts close to 80% and does not evolve where a normal learning curve should increase regularly before it plateaus. We see again that the alignment did not have much influence on the result but it clearly smoothed the evaluation loss curve.

An example of classification is shown on Figure 5. In that case, it appears that points were all classified as non ground. Like many ground points were missing in the reference, it seems that the network had difficulties to learn between ground and non-ground, because of the confusion with the low vegetation.

These tests show that the network was not yet able to learn properly to classify points. While in urban area, it tends to classify too many points as ground points, in forest area, it omits too many points. The forest area contained more points on the ground but they were concentrated in open areas. Point classification depends on the size of the blocks and blocks were probably too small for the network to identify a clear pattern for points on the ground since they are often mixed with low vegetation.

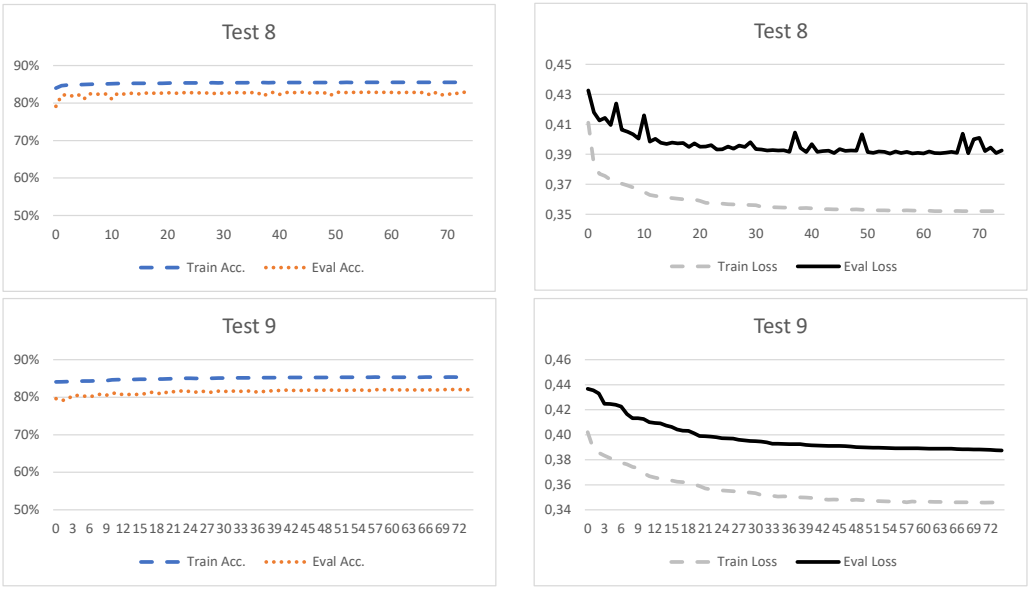

Figure 4: Accuracy and loss for training and evaluation of a forest point cloud with and without alignment

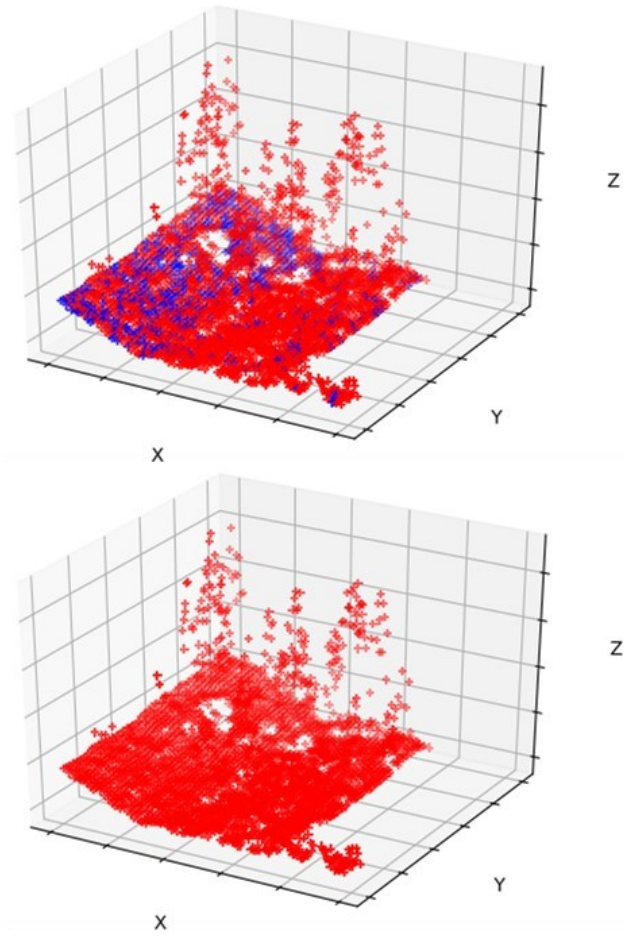

Figure 5: Comparison between reference points (above) and classified points (below). Blue represents ground points

5. CONCLUSION AND FUTURE WORKS

In this paper, we applied PointNet, a deep learning model, to classify aerial lidar point clouds. PointNet has the advantage of processing directly the points without voxelisation. Our current tests were able to provide results in line with existing approaches however they are not satisfactory. Loss curves show a lack of robustness. As mentioned, one reason was that, in datasets provided for training, quality was defined by the lack of commissions and many ground points had been omitted.

In order to improve the robustness of the method, the sigmoid cross-entropy method loss calculation should be modified to increase the loss when the network does commissions. In other words, we want to further penalise the neural network when it classifies a non-ground point as ground than when it classifies a ground point as non-ground.

Nonetheless, further investigation regarding data preparation (mainly normalisation and batch definition) need to be conducted since the approach so far lacks of robustness, especially in forest. Existing methods are applied for object classification or for segmentation of an indoor scene where normalisation is applied to the whole scene. In an open environment, different kinds of normalisation, along the whole point cloud or per tiles, rather than in each block or batch may be tested since the scene does not have natural boundary. Furthermore, features considered in the classification can vary depending on the context. Hence, data preparation may be conducted and training may be performed separately for different types of terrain (agricultural and forest areas, step terrain, etc.).

Finally, our analysis suggests that it is necessary to enrich the architecture of the convolutional network to integrate the notion of neighbourhood. Currently, neighbourhood is only implicitly considered within blocks. Blocks group together points in a batch that are spatially close to favour computation of global features from points which are close. However, PointNet is not yet able to handle notions of scale to compute features at different resolutions and define some clusters as existing geometrical and machine learning approaches are able to do. Some authors have considered these elements by explicitly defining a neighbourhood for points (Engelmann et al., 2017, Qi et al., 2017b) at different scales. Neighbourhoods are defined by k-nearest neighbours or

by a distance from a point. These approaches increase further the computation and storage cost and specific spatial access methods will have to be implemented, especially in natural environments where point distribution is more unstructured and clusters are more difficult to define.

ACKNOWLEDGEMENTS

This project was supported by NSERC grant EGP 531444-18 and by Xeos Imagerie Inc. The authors also thank Tony St-Pierre and Xeos Imagerie Inc for giving access to their data and for their assistance during the project.

REFERENCES

Axelsson, P., 2000. DEM generation from laser scanner data using adaptive TIN models. In: D. Fritsch and M. Molenaar (eds), International archives of photogrammetry and remote sensing, Vol. XXXIII, Part B4, pp. 110–117.

Bigdeli, B., Amirkolaee, H. A. and Pahlavani, P., 2018. DTM extraction under forest camopy using lidar data and a modified invasive weed optimization algorithm. Remote Sensing of Environment 216, pp. 289–300.

Chehata, N., Guo, L. and Mallet, C., 2009. Contribution of airborne full-waveform lidar and image data for urban scene classification. In: 16th IEEE International Conference on Image Processing, pp. 1669–1672.

Chen, Z., Gao, B. and Devereux, B., 2017. State-of-the-art: DTM generation using airborne lidar data. Sensors 17(150), pp. 24 pages.

Engelmann, F., Kontogianni, T., Hermans, A. and Leibe, B., 2017. Exploring spatial context for 3d semantic segmentation of point clouds. In: IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 716–724.

Evans, J. S. and Hudak, A. T., 2007. A multiscale curvature algorithm for classifying discrete return lidar in forested environments. IEEE transactions on geoscience and remote sensing 45(4), pp. 1029–1038.

Hui, Z., Li, D., Ziggah, Y. Y. and Wang, L., 2019. Automatic DTM extraction form airborne lidar based on expectation-maximization. Optics and Laser Technology 112, pp. 43–55.

Lodha, S. K., Kreps, E. J., Helmbold, D. P. and Fitzpatrick, D., 2006. 'aerial lidar data classification using support vector machines (svm). In: Proc., 3rd Int. Symp. on 3D Data Processing, Visualization and Transmission (3DPVT06), IEEE Computer Society, pp. 567–574.

Lu, W.-L., Murphy, K. P., Little, J. J., Sheffer, A. and Fu, H., 2009. A hybrid conditional random field for estimating the underlying ground surface from airborne lidar data. IEEE Transactions on Geoscience and Remote Sensing.

Maturana, D. and Scherer, S., 2015. Voxnet: A 3D convolutional neural network for real-time object recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 922–928.

Meng, X., Currit, N. and Zhao, K., 2010. Ground filtering algorithms for airborne lidar data: A review of critical issues. Remote sensing 2, pp. 833–860.

Mewhort, A., 2013. Active learning for semantic labelling of airborne lidar data. Master's thesis, Simon Fraser University.

Ni, H., Lin, X. and Zhang, J., 2017. Classification of ALS point cloud with improved point cloud segmentation and random forests. Remote Sensing 9(288), pp. 34 pages.

Niemeyer, J., Rottensteiner, F. and U, S., 2012. Conditional random fields for lidar point cloud classification in complex urban areas. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. I-3, pp. 263–268.

Nourzad, S. H. H. and Pradhan, A., 2014. Ensemble methods for binary classifications of airborne lidar data. Journal of Computing in Civil Engineering 28(6), pp. 04014021, 11 pages.

Qi, C. R., Su, H., Mo, K. and Guibas, L. J., 2017a. Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR), pp. 652–660.

Qi, C. R., Yi, L., Su, H. and Guibas, L. J., 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett (eds), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., pp. 5099–5108.

Wang, L., Huang, Y., Shan, J. and He, L., 2018. MSNet: multi-scale convolutional network for point cloud classification. Remote sensing.