# ASSESSING LIDAR TRAINING DATA QUANTITIES FOR CLASSIFICATION MODELS

O. Majgaonkar[1], K. Panchal[2] D. Laefer[3], M. Stanley[2], Y. Zaki[1]

[1] New York University Abu Dhabi, Abu Dhabi, UAE - (oorja.majgaonkar, yasir.zaki)@nyu.edu
[2] New York University, New York, NY, USA - (kp2670, michael.stanley)@nyu.edu
[3] Center for Urban Science and Progress; Department of Civil and Urban Engineering, New York University, New York, NY, USA - debra.laefer@nyu.edu

**KEY WORDS:** LiDAR, Object Classification, Learning Curve, Point Cloud, Training.

**ABSTRACT:**

Classifying objects within aerial Light Detection and Ranging (LiDAR) data is an essential task to which machine learning (ML) is applied increasingly. ML has been shown to be more effective on LiDAR than imagery for classification, but most efforts have focused on imagery because of the challenges presented by LiDAR data. LiDAR datasets are of higher dimensionality, discontinuous, heterogenous, spatially incomplete, and often scarce. As such, there has been little examination into the fundamental properties of the training data required for acceptable performance of classification models tailored for LiDAR data. The quantity of training data is one such crucial property, because training on different sizes of data provides insight into a model's performance with differing data sets. This paper assesses the impact of training data size on the accuracy of PointNet, a widely used ML approach for point cloud classification. Subsets of ModelNet ranging from 40 to 9,843 objects were validated on a test set of 400 objects. Accuracy improved logarithmically; decelerating from 45 objects onwards, it slowed significantly at a training size of 2,000 objects, corresponding to 20,000,000 points. This work contributes to the theoretical foundation for development of LiDAR-focused models by establishing a learning curve, suggesting the minimum quantity of manually labelled data necessary for satisfactory classification performance and providing a path for further analysis of the effects of modifying training data characteristics.

## 1. INTRODUCTION

Aerial light detection and ranging (LiDAR) data are useful for various mapping, surveying, and planning purposes in urban and natural areas. The data are most often found as a collection of points known as a point cloud, in which each point contains x, y, and z coordinates, return intensity, the number of returns from that particular coordinate, a timestamp, and source or flight line information. Specifically in urban modelling , the classification of points is essential for object identification, and LiDAR data has been demonstrated to be more effective than imagery for urban object classification (Scaioni et al., 2018). This classification is sometimes done by identifying features of the data and using predictive models or machine learning (ML) based models on these features, but these techniques involve significant human and computational resources (Kang and Yang, 2018). With the development of deep learning models, much work has been done in applying and modifying these techniques to be used for three-dimensional (3D) data to increase efficiency in classification. However, certain properties of 3D data present a challenge to typical machine learning models that were originally designed for object classification in two-dimensional (2D) imagery, so these models cannot be directly applied to LiDAR data sets

### 1.1 Greater dimensionality, unorderedness, and other traits

LiDAR data contain more information than images. Converting LiDAR data into 2D using projections reduces the richness of the data and introduces false relations by collapsing the space between points (Qi et al., 2017). For example, two points belonging to different objects have no contextual relation in the original data but could be contiguous in its 2D data projection. The assumption of continuity is unnecessary and negatively impacts the accuracy of the classification. On the other hand, some classification approaches turn the point cloud into voxels, which increases complexity by generating index information that must also be stored.

A given aerial LiDAR dataset is typically collected over several flight paths that cover the total area from different angles, potentially recording a location multiple times with separate flight paths. Thus, the resulting data are unordered. Consequently, classification algorithms or ML approaches must either forgo relying upon an inherent ordering or must impose that order as part of the pre-processing.

Furthermore a given point, together with its neighboring points, usually forms a meaningful subset. Therefore, points cannot easily be considered in isolation. In other words, classification of a point must be consistent with its neighbors. Moreover, the spatial relation between points must be preserved when the points undergo transformations or convolutions.

Additionally, the geometric properties of the collected data set largely depends on the characteristics of the objects and surfaces in the scene. For instance, parts of the ground obstructed by trees, vertical surfaces, and water will have fewer LiDAR returns than solid, horizontal surfaces (Stanley and Laefer, 2021). Figure 1 shows an example. Additionally, the data sets are discontinuous, with sections that are completely empty when visualized in a 3D space. Finally, points are not uniformly distributed, and point density can vary across the data set. Consequently, models must be insensitive to these traits.

With increased understanding of the benefits of LiDAR-based urban object classification, significant efforts have been put into developing neural networks more suited for point clouds by

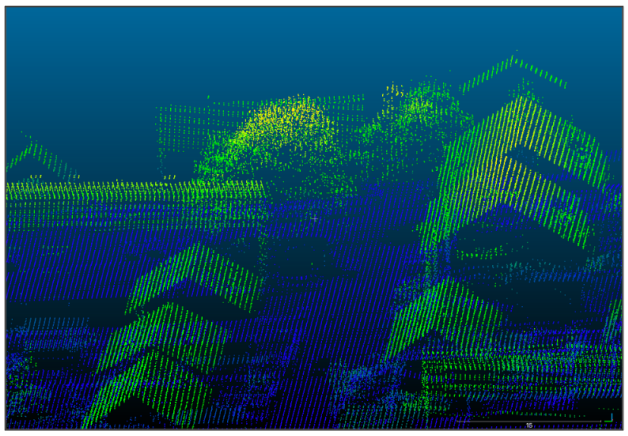

Figure 1. Non-uniform density in trees and roofs in Vaihingen data set (Rottensteiner et al., 2013)

considering their attributes more fully. The focus of that research has been on developing novel techniques that are mostly performance driven with little investigation into training set optimization. Potential optimization factors include the size of the training set and the balance of objects across class, motion, position, and orientation of objects, as well as density. While more training data are often assumed to improve performance, there is no clearly established relationship published to date between LiDAR training data quantity and performance of classification models. Thus, the objective of this project is to assess the impact of training data quantity on performance of a LiDAR-focused object classification model.

## 2. RELATED WORK

Object identification is a long standing and widely undertaken activity in LiDAR processing (e.g. Aljumaily et al. 2017, Nurunnabi et al. 2018, Soilan et al. 2018, Zolanvari et al., 2018). In the ML sphere, there has been a gradual trajectory of moving away from feature-based learning to that which is more contextualized.

### 2.1 Feature-based learning

A prominent example of feature-based object classification is that by Song et al. (2018). They derived these features by calculating the volume, density, and eigenvalues in three directions of each object. These statistical features were then used to divide the objects into categories. Then, a back-propagated neural network was trained on these features and used to classify objects in a given scene. However, this methodology requires the laborious task of crafting features by hand. Furthermore, the authors noted that the model's accuracy could only be improved through additional manual labeling, which complicates this type of approach being applied to very large, heterogeneous data sets (Song et al., 2018). In addition, hand-crafted features consider points in isolation by giving each point a label based on the features calculated on that point alone, without ensuring consistency with neighboring points (Wen et al., 2020). This can result in noisy and inconsistent labelling.

### 2.2 PointNet

To overcome some of the problems with feature-based classification, Qi et. al. (2017) developed PointNet, a neural network that directly acts on point clouds rather than converting data into 3D voxels or relying on hand-crafted, statistical features. The result is the ability to process points in O(N) time and space. In contrast, other methods have required $O(N^2)$ or $O(N^3)$ time and space.

The development of PointNet used ModelNet, a data set consisting of 12,311 LiDAR scans. These are split into 9,843 samples for training and 2,468 samples for testing (Wu et al., 2015). Importantly, PointNet was designed for identifying small, mostly indoor objects such as chairs or tables, rather than for outdoor objects. Qi et. al. (2017) described an application for semantic segmentation from a scene containing several objects. In this case, the input data were divided into blocks of equal size. However, these were also near-range objects. Thus, the problem of identifying objects with fewer features, occlusions, and significant noise was not addressed (Yousefhussien et al., 2018).

### 2.3 PointNet-inspired convolutional neural networks

While some studies have concluded that training separate neural networks for short-range versus long-range objects results in better performance (Engels et al., 2020), others (Yousefhussien et al., 2018) found that using an overlapping technique in the input blocks trained the model to recognize objects of different scales, thereby preempting the need for separate networks for different scales. This suggests that a performance analysis of PointNet can be beneficial and applicable to establishing the relationship between performance, data size, and scene size with regards to aerial data.

Other models inspired by PointNet modify the approach in a way that is agnostic to the artefacts of aerial data. For example, Wen et al. (2020) produced a "directionally constrained" fully convolutional neural network that performed convolutions considering the orientation of objects. As a directionally constrained network, information in the z-direction was not considered (Wen et al., 2020). This was based on the idea that not every point requires information from its neighbors in the z direction, because many surfaces, such as rooftops have no points above them, and surfaces such as roads have no points below them. These types of approaches do not rely on hand-crafted features, which are laborious to derive.

To date, many of the studies in this area have used the same data sets with nearly the same split between the training and validation. For instance, (Yousefhussien et al., 2018) used 753,859 training points and 411,721 testing points while (Wen et al., 2020) used 753,876 training points and 411,722 testing points, both from the ISPRS Vaihingen data set (ISPRS, 2013).

### 2.4 Learning curves

The performance of any machine learning algorithm can be quantified by a learning curve, which benchmarks a generalization performance metric, such as accuracy or error, against the quantity of training data (Cohen et al., 2021). By illustrating the effect of training different amounts of data, one can determine the amount of training data needed. For instance, Figure 2 shows diminishing returns in performance with more training samples, and Figure 3 shows the generalization error increasing after a certain point, indicating overfitting. This paper and most others in this field have not investigated the impact of the size of the training data set on the final performance.

Perlich (2010) reaffirms that in comparing machine learning models, results that are reported on a fixed-size training data set

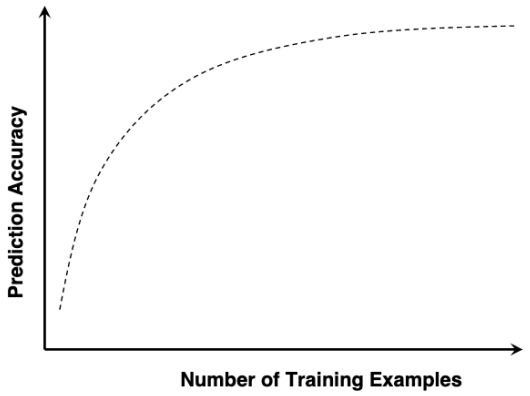


Figure 2. Learning curve showing accuracy versus number of training samples (Perlich, 2010)

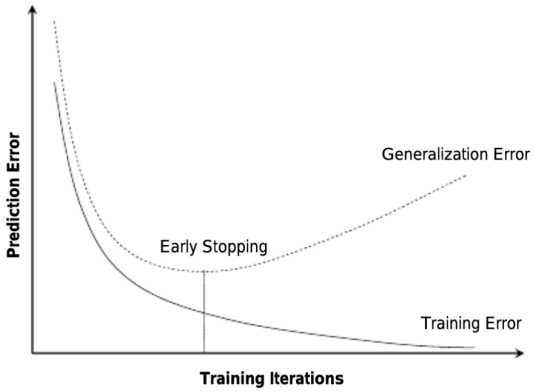


Figure 3. Learning curve showing generalization and training error versus training iterations (Perlich, 2010)

do not provide any information on how the model would fare with differing training data sizes. Most papers report results in this way, as seen in the PointNet-derived neural networks, which discussed training on the same data set. The insight on varying training data size is important to ascertain the reliability of the model in new applications, where the same amount of training data is not always available as that which was used to originally train the model. Evaluating performance using different sizes of data can increase the robustness of the model and provide more holistic insights, rather than only testing and reporting results achieved on fixed-size data sets.

Additionally, there has been some investigation into theoretical foundation regarding learning curves in general. Cohen et al. (2021) noted that neural networks are often built for a specific application, focusing solely on achieving greater accuracy by modifying the parameters based on trial and error rather than on theory or conducting some form meta analysis of the learning curves. Those authors attempted to model learning curves with Gaussian Processes using techniques from physics.

While there is a general preference for more training data, this is a laborious task especially for LiDAR, and although the time and space complexity of the classification task itself has improved, existing methods involve high complexity pre-processing. There is, thus, an incentive to examine the precise change in performance with increasing input data.

## 3. METHODOLOGY

### 3.1 Scope

This research employed an implementation of PointNet with state-of-the-art performance to test the impact of the number of objects in the training data and the number of epochs as metrics to consider accuracy performance (percentage of true positives). While the final performance analysis used the ModelNet data set consisting of scans of 40 categories of objects (Table 1), the study was informed with visualization investigations by two publicly available datasets Vaihingen (Rottensteiner et al., 2013) and Sunset Park (Laefer and Vo, 2020).

| | | | | | |
|---|---|---|---|---|---|
| Airplane | 727 | Dresser | 287 | Range hood | 216 |
| Bathtub | 157 | Flower pot | 170 | Sink | 149 |
| Bed | 616 | Glass box | 272 | Sofa | 781 |
| Bench | 194 | Guitar | 256 | Stairs | 145 |
| Bookshelf | 673 | Keyboard | 166 | Stool | 111 |
| Bottle | 436 | Lamp | 145 | Table | 493 |
| Bowl | 85 | Laptop | 170 | Tent | 184 |
| Car | 298 | Mantel | 385 | Toilet | 445 |
| Chair | 990 | Monitor | 566 | TV stand | 368 |
| Cone | 188 | Night stand | 287 | Vase | 576 |
| Cup | 100 | Person | 109 | Wardrobe | 108 |
| Curtain | 159 | Piano | 332 | XBox | 124 |
| Desk | 287 | Plant | 341 | | |
| Door | 130 | Radio | 125 | | |

Table 1. ModelNet object distribution

### 3.2 Model architecture and implementation

PointNet's classification architecture, displayed in Figure 4, receives a number of points as input. It applies transformations between two multi-layer perceptrons (mlp) and then aggregates point features by max pooling. Finally, to leverage both global knowledge of points in the whole cloud and local knowledge of neighboring points in the classification, a global feature vector is calculated and applied to local feature vectors. The features on each point, thus, contain both local and global information, thereby allowing for an informed classification (Qi et al., 2017). The model outputs a score for each of the 40 categories for every test object and chooses the label with the highest score.

This project builds off a Pytorch implementation of PointNet (Yan, 2019). The program takes an input of a list of text files, where each file represents an object and contains the points that comprise the object. In every run of the classification training, a pth file is output containing: (1) the configuration of the model; (2) a log file consisting the model's performance on training data; (3) overall and class performance on test data; and (4) time stamps.

### 3.3 Pre-processing

**3.3.1 Preliminary analysis** Manual analysis was performed on the Sunset Park data (Laefer and Vo, 2020) in CloudCompare to develop an understanding of visual features in the data and determine what relevant variables were worth examining. Four size-related variables in the training data were originally identified: number of points, number of overlapping points across objects, variation of number of points in each object, and number of objects. Ultimately, the number of objects was chosen as the focus because of its semantic value and that it matched the format of data necessary for the PointNet implementation (i.e. training on individual files containing a collection of points that each represent an object).

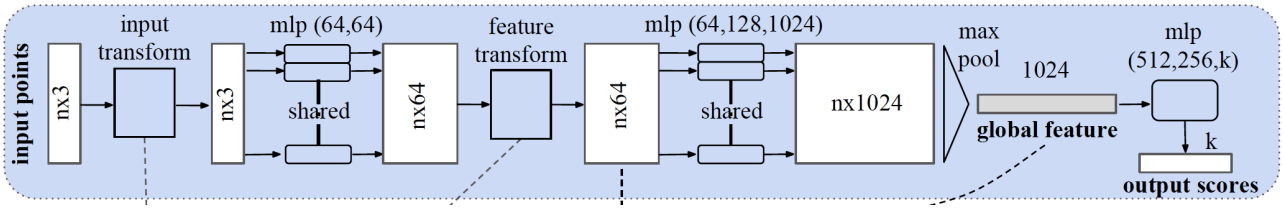


Figure 4. PointNet classification architecture (Qi et al., 2017)

**3.3.2 Segmentation** The model was trained on the following progression of training sizes: 40, 80, 120, 240, 320, 400, 600, 800, 1,000, 2,000, 4,000, and 9,843 (full). As accuracy results were plotted for each size to investigate the rate of improvement, having training sizes with increasing levels of separation allowed for more insights into the shape of the learning curve without having to train on equi-spaced, intermediate sizes. For each size n between 40 to 4,000 in the training size list, n/40 objects from each category were taken for testing. For the size 9,843, all the objects were used. The last 100 objects from each category were taken for validating all the input sizes. The PointNet implementation code was modified to create the ability to use data of different sizes for training, instead of only a single default size.

### 3.4 Training

Every size was trained with 200 epochs. At each epoch, if the accuracy exceeded the highest saved accuracy so far, the results for that epoch were saved as the best instance of achieved accuracy. PointNet was used with the parameter configurations shown in Table 2 for all runs.

| Parameter name | Value |
|---|---|
| Batch size | 24 |
| Learning rate | 0.001 |
| Optimizing algorithm | Adam |
| Decay rate | 0.0001 |

Table 2. PointNet parameter configurations

## 4. EVALUATION

### 4.1 Accuracy versus training iterations

Table 3 displays the raw accuracy obtained on each training data set size at every 20 epochs. Figure 5 illustrates the results for each run plotted against epoch. The line for training data size 40 visually differs from the others because of its flat start – indicating a slow start in the model's learning. In this case, the model consistently classified with an accuracy of 0.024510 on unseen data for the first 12 epochs. Only after that did it start increasing and did not reach 0.1 until around the 45th epoch.

The general shape of the curves illustrated that as training size increased, the improvement rate declined. This began around 45 epochs for all sizes, except for size 40, whose performance started improving only after epoch 45. With that case, slowing began near 90 (corresponding to only 45 epochs of fast growth).

The plots of raw accuracies contained many areas of overlap, especially in the earlier stages of training. To improve the interpretability of the individual lines and allow for clearer comparison between them, the moving mean of the accuracy was taken with a moving mean window of 10 (Figure 6).

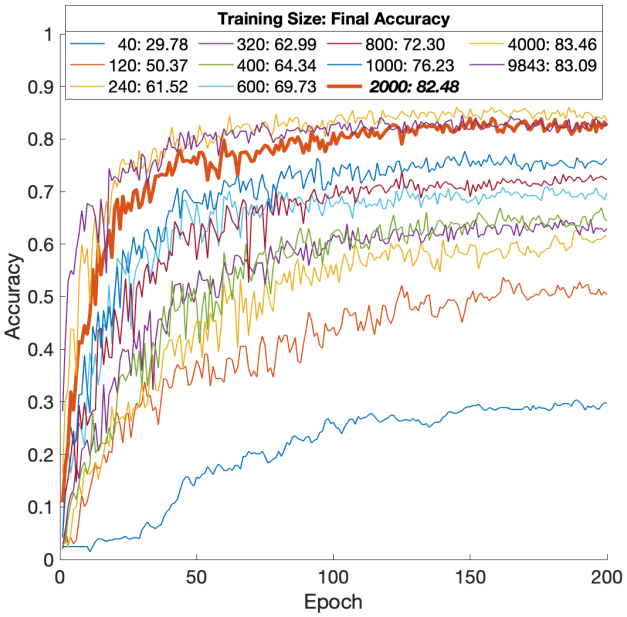


Figure 5. PointNet accuracy versus epoch

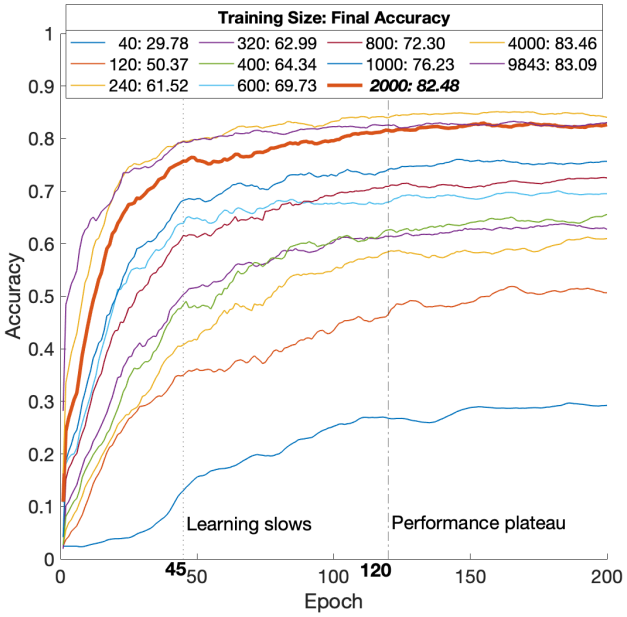


Figure 6. PointNet moving mean accuracy versus epoch

With the smoothing of the curves, they more resembled those in Figure 2. The gaps between the curves also revealed that for the most part, the model's performance was more acutely impacted by increases in the training data size at the smaller sizes. For example, the data size of 4,000 was double the value of the preceding size, yet the gap between that pair of curves was not significantly bigger than others. Furthermore, the full

| | Training set size | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 40 | 120 | 240 | 320 | 400 | 600 | 800 | 1000 | 2000 | 4000 | 9843 |
| Epoch | Accuracy | | | | | | | | | | |
| 20 | 0.0392 | 0.2659 | 0.2549 | 0.3468 | 0.2426 | 0.4926 | 0.4767 | 0.5049 | 0.6679 | 0.7218 | 0.7292 |
| 40 | 0.0956 | 0.3260 | 0.4179 | 0.4743 | 0.4228 | 0.5699 | 0.5870 | 0.6642 | 0.7549 | 0.7966 | 0.8039 |
| 60 | 0.1691 | 0.3370 | 0.5257 | 0.5662 | 0.5846 | 0.6912 | 0.6238 | 0.7218 | 0.7978 | 0.8088 | 0.8199 |
| 80 | 0.2059 | 0.4265 | 0.5233 | 0.5625 | 0.5870 | 0.6998 | 0.6544 | 0.7230 | 0.7794 | 0.8235 | 0.8015 |
| 100 | 0.2610 | 0.4424 | 0.5404 | 0.6189 | 0.6225 | 0.6728 | 0.6912 | 0.7463 | 0.8113 | 0.8297 | 0.8272 |
| 120 | 0.2696 | 0.4583 | 0.5686 | 0.6017 | 0.6103 | 0.6850 | 0.7010 | 0.7488 | 0.8199 | 0.8456 | 0.8297 |
| 140 | 0.2708 | 0.4779 | 0.6091 | 0.6164 | 0.6385 | 0.6936 | 0.7145 | 0.7475 | 0.8125 | **0.8529** | 0.8248 |
| 160 | 0.2868 | **0.5037** | 0.5919 | 0.6176 | **0.6556** | 0.6985 | 0.7194 | 0.7561 | 0.8137 | 0.8529 | 0.8211 |
| 180 | 0.2880 | 0.5037 | 0.6054 | 0.6275 | 0.6471 | **0.7071** | 0.7132 | 0.7488 | **0.8248** | 0.8505 | 0.8211 |
| 200 | **0.2978** | 0.5037 | **0.6152** | **0.6299** | 0.6434 | 0.6973 | **0.7230** | **0.7623** | 0.8248 | 0.8346 | **0.8309** |

Table 3. Raw accuracy values every 10 epochs on training set size

data set with 9,843 objects under-performed compared to size 4,000, thereby showing that the model was overfitting at this point.

### 4.2 Learning curve

To account for the varying interval sizes in training data, the best instance accuracy for each run was then extracted and graphed. Figure 7 illustrates the relationship between training size and the best instance accuracy. This graph resembles a logarithmic growth curve. The accuracy appears to start plateauing with a training set size of 2,000, as highlighted in Figures 5 and 6, with only a small increase in accuracy from 2,000 to 4,000. Finally, training on the full set of 9,843 objects appeared harmful to the model's performance.

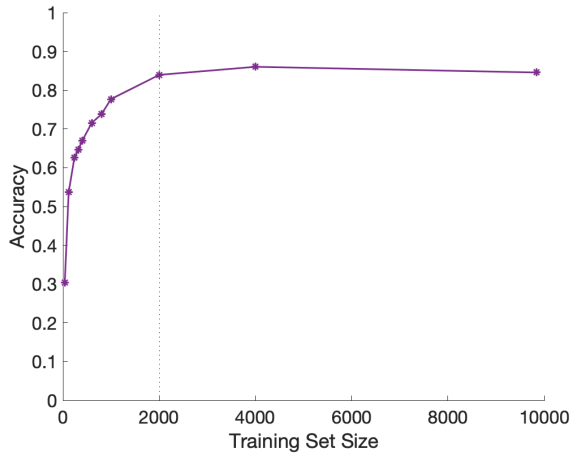


Figure 7. PointNet learning curve: maximum accuracy per training dataset size

## 5. DISCUSSION

Critically, the results produced herein were based on a near perfect training data set in which all objects were small scale and scanned from close range. The final output was a collection of point clouds each containing 10,000 points and each complete, dense, clean, and without abnormalities. These produced clearly defined shapes that differ from readily-achievable, real-world aerial LiDAR data, which contain flaws like density differences, noise, and incompleteness due to self-shadowing and street shadowing, as defined by Hinks et al. (2009). The reasons are multi-fold. For example, aerial LiDAR scans include both stationary and moving cars (Figure 8), and the latter introduces latency into the object representation.

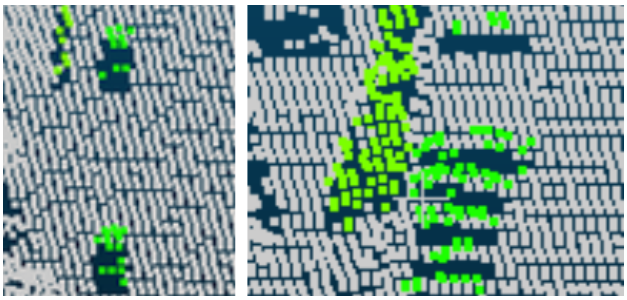

Figure 8. Moving cars (left) and parked cars (right) in Vaihingen data set (Rottensteiner et al., 2013)

In commercial aerial scans, surface composition, obstructions, object motion and orientation, and incident angle all affect the consistency of the returned LiDAR points (Figure 1). The properties of the flight mission including elevation, LiDAR sensor orientation, sensor field of view, flight speed, and pulse rate also influence the final point cloud quality (Stanley and Laefer, 2021). Wen et al. (2020) and Yousefhusssien et al. (2018) also noted that aerial data has lower and more inconsistent density per volume of an object. Examining the effects of uneven distributions in both the number of points per object and number of objects per category will be important to establishing a more fundamental understanding of ML optimization for use with LiDAR point cloud data. Since indoor, close-range objects have more defined shapes and higher point densities compared to aerial data, the model would be expected to be less accurate on sparse data sets. The relationship between the relative perfection of the training set versus the quality of the actual data and its impact on performance is an area ripe for investigation. Furthermore, training data can be manipulated to resemble aerial data by adding noise artificially and removing vertical surfaces. While the reported results showed a relationship of logarithmic growth in accuracy that largely mirrored those reported by Perlich (2011) and Cohen et al. (2021), this pattern needs to be further tested on larger and less perfect datasets to determine the generalizability of the observations herein.

## 6. CONCLUSIONS

To date, the focus of neural networks dedicated to LiDAR has predominantly concentrated on developing new networks rather than evaluating the characteristics of their training protocols. As such, many fundamental questions on how performance changes with modifications in the input data remain unexplored. One major component is establishing the ideal quantity of training data as a balance of resources versus performance. This research begins to address this gap by assessing the impact of

training data quantity on the classification accuracy of PointNet, and, thus, lays the ground work for further evaluation of input data changes with respect to their impact on model performance. This paper demonstrated that the model's learning rate was unaffected by the training data set size with respect to the number of epochs. Specifically, all sizes exhibited a slowing in accuracy improvement following epoch 45. The learning curve started plateauing with a training set size of 2,000 objects, with a very small increase observed from 2,000 to 4,000 and a slight decrease from 4,000 to 9,843. The objects in the training data each contained 10,000 points. Therefore, the corresponding number of points after which improvement was only incremental was 20 million. Furthermore, the slight decrease in the model's accuracy between 4,000 to 9,843 showed overfitting. Consequently, more investigation is needed as to whether a training set size of approximately 20,000,000 points can be generalized for the use of the "ModelNet" data set.

## 7. ACKNOWLEDGMENTS

Funding for this project was provided by the National Science Foundation awards 1826134 and 1940145. The ModelNet, Vaihingen, and Sunset Park data sets were provided by the Princeton ModelNet Project, German Society for Photogrammetry, Remote Sensing and Geoinformation, and the NYU Center for Urban Science and Progress, respectively.

## REFERENCES

Aljumaily, H., Laefer, D., Cuadra, D., 2017. Urban Point Cloud Mining Based on Density Clustering and MapReduce. *Journal of Computing in Civil Engineering*, 31, 04017021.

Cohen, O., Malka, O., Ringel, Z., 2021. Learning curves for overparametrized deep neural networks: A field theory perspective. *Physical Review Research*, 3, 023034.

Engels, G., Aranjuelo, N., Arganda-Carreras, I., Nieto, M., Otaegui, O., 2020. 3d object detection from lidar data using distance dependent feature extraction. *Proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems - VEHITS*, 289–300.

Hinks, T., Carr, H., Laefer, D. F., 2009. Flight Optimization Algorithms for Aerial LiDAR Capture for Urban Infrastructure Model Generation. *Journal of Computing in Civil Engineering*, 23(6), 330-339.

Kang, Z., Yang, J., 2018. A probabilistic graphical model for the classification of mobile LiDAR point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 143, 16.

Laefer, D., Vo, A.-V., 2020. 2019 LiDAR Data Collection for Sunset Park, Brooklyn, NY. *NYU Spatial Data Repository*. Available at *http://hdl.handle.net/2451/60458*.

Nurunnabi, A., Sadahiro, Y., Laefer, D., 2018. Robust statistical approaches for circle fitting in laser scanning three-dimensional point cloud data. *Pattern Recognition*, 81, 417-431.

Perlich, C., 2010. *Learning Curves in Machine Learning*. Springer US, Boston, MA, 577–580.

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

Rottensteiner, F., Sohn, G., Gerke, M., Wegner, J. D., 2013. ISPRS Vaihingen dataset. *https://www2.isprs.org/media/komfssn5/complexscenes_revision_v4.pdf*.

Scaioni, M., Höfle, B., Kersting, A., Barazzetti, L., Previtali, M., Wujanz, D., 2018. Methods From Information Extraction From LiDAR Intensity Data and Multispectral LiDAR Technology. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-3, 1503-1510.

Soilán, M., Truong-Hong, L., Riveiro, B., Laefer, D., 2018. Automatic extraction of road features in urban environments using dense ALS data. *International Journal of Applied Earth Observation and Geoinformation*, 64, 226-236.

Song, W., Zou, S., Yifei, T., Fong, S., Cho, K., 2018. Classifying 3D objects in LiDAR point clouds with a back-propagation neural network. *Human-centric Computing and Information Sciences*, 8.

Stanley, M. H., Laefer, D. F., 2021. Metrics for aerial, urban lidar point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 175, 268-281.

Wen, C., Lina, Y., Li, X., Peng, L., Chi, T., 2020. Directionally constrained fully convolutional neural network for airborne LiDAR point cloud classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 162, 50-62.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3d shapenets: A deep representation for volumetric shapes. 1912–1920.

Yan, X., 2019. Pointnet/Pointnet++ Pytorch. *https://github.com/yanx27/Pointnet*$_P ointnet2_p ytorch$.

Yousefhussien, M., Kelbe, D., Ientilucci, E., Salvaggio, C., 2018. A multi-scale fully convolutional network for semantic labeling of 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 143, 191-204. Theme Issue "Point Cloud Processing".

Zolanvari, S. I., Laefer, D. F., Natanzi, A. S., 2018. Three-dimensional building façade segmentation and opening area detection from point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 143, 134-149. Theme Issue "Point Cloud Processing".