Assessing training data quantity for optimum performance of LiDAR object classification models

Oorja Majgaonkar
Computer Science, NYUAD
ojm238@nyu.edu

Advised by: Debra Laefer

ABSTRACT

Classifying objects within aerial light detection and ranging (LiDAR) is an essential task that has recently been automated by machine learning. Conventional machine learning techniques cannot be used for 3-dimensional LiDAR data, prompting a recent growth in the development of neural networks specifically for 3D data. However, there has not been an examination into the quantity of data required for a high performing model that is reliable. Provisioning aerial LiDAR data requires significant manual labour, so there is an incentive in finding out how much data are required for a model to achieve good performance. This project assess the impact of the number of objects in the training data quantity on the classification accuracy of PointNet, a deep learning framework tailored for 3D point clouds. It is found that the underlying assumptions about training data for such networks remain largely uncovered and unaddressed. A PointNet implementation is trained on subsets of a large data set that contain an increasing number of objects, from 40 to 2000. It is seen that the accuracy of this implementation improves at a logarithmic rate within this data range and starts to even out at approximately 600,000 training points. This work provides a benchmark for subsequent development by establishing a learning curve for 3D-focused classification models and suggests how much data should be manually labelled and used for training when the model is used on new data.

Capstone Project 2, Spring 2021, Abu Dhabi, UAE

KEYWORDS

aerial LiDAR, object classification, learning curve, point cloud, training

Reference Format:
Oorja Majgaonkar. 2021. Assessing training data quantity for optimum performance of LiDAR object classification models. In *NYUAD Capstone Project 2 Reports, Spring 2021, Abu Dhabi, UAE.* 9 pages.

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 and planning, the classification of points in the point cloud is essential for object identification, and LiDAR data has been demonstrated to be more effective than imagery for urban object classification [6]. This classification is sometimes done by identifying features of the data, and using predictive models or machine learning-based models on these features, but these techniques involve significant human or computational resources [3]. With the development of deep learning models, much work has been done in applying and modifying these techniques to be used for 3-dimensional, particularly LiDAR, data, increasing the efficiency of the classification task.

1.1 Challenges with 3-dimensional LiDAR data

Certain properties of 3-dimensional data present a challenge to typical machine learning models used for object classification in 2-dimensional imagery, so these models cannot be directly applied to LiDAR data sets.

1.1.1 Greater information dimensionality. LiDAR data contain more information than images. Converting LiDAR data to 2D using projection, therefore, loses information in the third dimension and introduces false relations by collapsing the space between points [5]. For example, two points belonging to different objects have no contextual relation in the original data, but could be contiguous in its 2-dimensional 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. This approach this massively increases the space complexity by filling up the space between points and adding volume.

1.1.2 Unorderedness. A given dataset is typically collected over several flight paths that cover the total area from different angles, potentially going over a single point on the ground multiple times, or with paths overlapping each other. A point cloud differs from pixels in an image or voxels because the points are not ordered. The model that does classification must not depend on the input data being fed in a particular order.

1.1.3 Relation to other points. A given point, together with its neighboring points, usually forms a meaningful subset, therefore points cannot be considered in isolation. In other words, the 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.

1.1.4 Discontinuity. The geometric properties of the data set that is collected largely depends on the characteristics of the objects and surfaces in the area covered. For instance, a water covered surface, parts of the ground obstructed by trees, and vertical surfaces will have fewer LiDAR returns. The image in Figure 1 shows an example of this in the Vaihingen data set. As a result, the final data set is discontinuous, with sections of it that are completely empty when visualized in a 3-dimensional space. Points are also not uniformly distributed, and density of points can vary across the data set. The model used thus has to be tolerant of this discontinuity as well as non-uniform density.

With increased understanding of the benefits of LiDAR-based urban object classification, significant effort has recently been put into developing neural networks from the ground up that are more suited for point clouds because they fully consider their properties. The focus of this research has been on developing and adapting novel techniques for the development of new models, so they are mostly driven by performance results. In contrast, there is less focus on determining what training data are best suited for such models. Quantity of training data is one significant factor, amongst others such as balance of objects across class, motion, position, or orientation of objects, and density. While it is generally assumed that more training data will improve performance, there is no clearly established relationship between training data quantity and performance of a LiDAR-focused classification model.

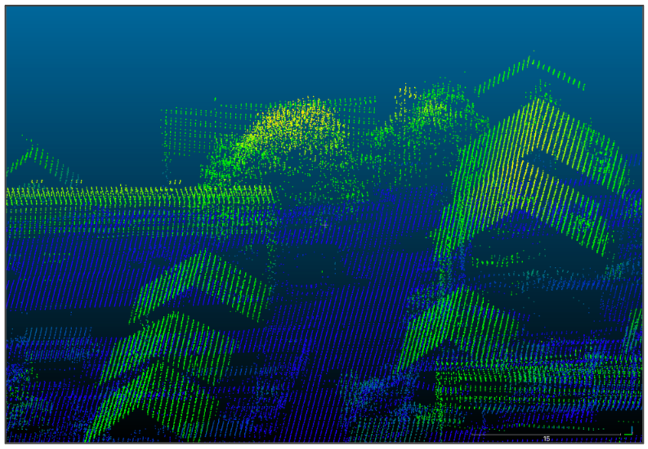
Figure 1: Non-uniform density in trees and roofs

Thus, the objective of this project is to assess the impact of training data quantity on performance of LiDAR-focused object classification models.

2 RELATED WORK

2.1 Feature-based learning

Other than aerial scanning, autonomous driving is another field in which LiDAR is a prominently used technology. Song et. al. use a feature-based approach to object classification for the application of self-driving vehicles. They derive these features by calculating the volume, density, and eigenvalues in three directions of each object. These statistical features are then used to divide the objects into categories. Then, a back-propagated neural network is trained on these features and then used to classify objects in a given scene. However, this methodology requires on the laborious task of crafting features by hand, as Song et. al. note that the model's accuracy can only be improved "by gathering more manually-labelled object feature datasets" [7]. For very large, dense data sets, such as the Sunset Park data, the amount of human and computational resources required for crafting such features would make it an inefficient task.

In addition, hand-crafted features consider points in isolation, giving each point a label based on the features calculated of that point alone with ensuring that it is consistent with the neighboring points [8]. This results in a noisy and inconsistent labelling, for instance within an object with points mostly classified as tree, there may be several "noisy" points classified as person.

2.2 PointNet

Qi et. al. develop 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 that convert the data into images or volumes have required O(N^2) or O(N^3) time and space. PointNet also achieved a state of the art performance at its time.

The development of PointNet uses the ModelNet data set, consisting of 12,311 CAD models. These are split into 9,843 samples for training and 2468 samples for testing [9]. This means that PointNet is designed for identifying smaller, mostly indoor objects such as chairs or tables, rather than for remote sensing.

Qi et. al. do describe an application for semantic segmentation, which is identifying semantic objects out of a scene containing several objects. Given an input data of a scene containing many objects, each point in the scene will be given one label corresponding to one of the categories it was trained on. In this case, the input data are divided into blocks of equal size. However, these are also from a close proximity and do not fully resolve the issues presented by large-scale aerial data. Large objects from a greater distance such as trees or buildings may have shapes that are not as densely defined and also feature more noise, occlusions, and clutter [11].

2.3 PointNet-inspired convolutional neural networks

While some studies have concluded that training separate neural networks for short-range versus long-range objects result in better performance [2], some such as [11] find that using an overlapping technique in the input blocks trains the model to recognize objects of different scales, preempting the need for separate networks for different scales. This suggests that a performance analysis of PointNet can still be beneficial and applicable to establishing the relationship between performance and data size with regards to aerial data.

Other models that are inspired by PointNet modify the approach in a way that is agnostic to the artefacts of aerial data. Wen et. al. produced a "directionally constrained" fully convolutional neural network (D-FCN) that performs convolutions considering the orientation of objects. As a directionally constrained network, information in the z-direction were not considered [8]. This is based on the idea that not every point requires information from its neighbors to be collected in the z direction, because many surfaces, such as rooftops, will

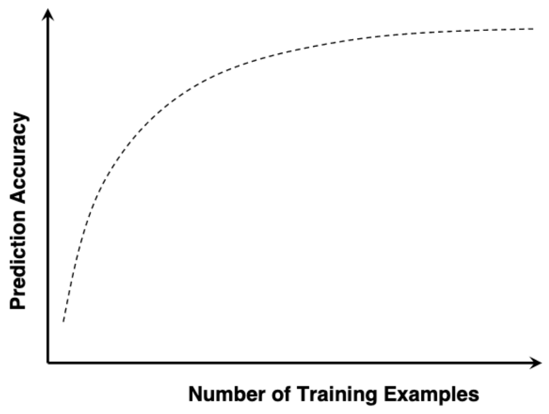

Figure 2: Learning curve showing accuracy versus number of training samples [4]

not have points above them, while surfaces such as roads will have no points below them. Unlike [11], they do not rely on hand-crafted features, which are tedious to derive, and achieve a high performance using only the 3D coordinates and intensity. Their improvement from is that it does not rely on hand-crafted features which are tedious. Only the 3D coordinates and intensity are sufficient for the high performance that it achieves.

Much of the literature uses the same data set with nearly the same training and validation split. For instance, [11] use 753,859 training points and 411,721 testing points while [8] use 753,876 training points and 411,722 testing points, both from the ISPRS Vaihingen data set.

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 [1]. The quantity of data can be either number of training iterations, or epochs, or number of training samples [4]. The model's performance on training set is also sometimes displayed to show the progress in learning. By illustrating the effect of training different amounts of data, one can determine at what point the amount of training data is considered sufficient, how much is redundant, and possibly at what point the amount of data causes overfitting. Simple examples can be seen in Figures 1 and 2.

[4] also reaffirms that in comparative analysis of machine learning models, results that are reported on a fixed-size training data set 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 discussed which train on the same

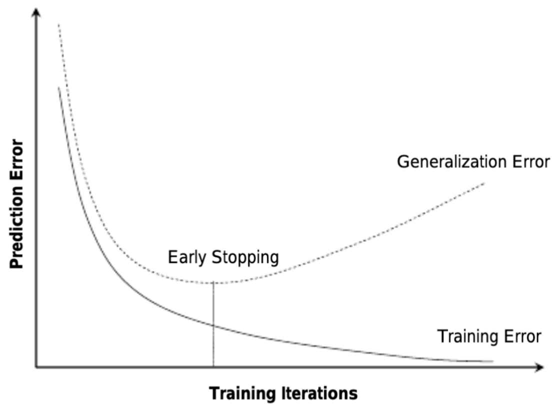

Figure 3: Learning curve showing generalization and training error versus training iterations [4]

data set. The insight on varying training data size is important to ascertain the reliability of the model in new applications, where the amount of training data available is not always the same as what the model was trained on during development. Evaluating performance using different sizes of data can therefore increase the robustness of the model, providing more holistic insights rather than only testing and reporting results achieved on fixed data sets.

The gap in theoretical foundation regarding learning curves has been investigated to some extent for neural networks in general. Cohen et. al. find that neural networks are often built for a specific application, focusing solely on performance and achieving this higher performance by modifying the parameters based on trial and error rather than on theory. They thus conduct a meta analysis of learning curves [1]. They use a physics-like approach attempting to model learning curves with Gaussian Processes, and suggest that this successful in predicting learning curves for the kind of neural networks they examine as seen in Figure 4.

Although there is a general preference for more training data, provisioning the training data by identifying and labelling new samples manually is a laborious task especially for LiDAR. Even with improvements in the time and space complexity of classification itself, existing methods involve high complexity preprocessing of input data. There is thus an incentive to examine the precise change in performance with incrementing input data examples.

3 METHODOLOGY

3.1 Scope

This research experiments on an implementation of PointNet, which has state of the art performance and is also accessible to run. Firstly, this work uses the number of objects in the

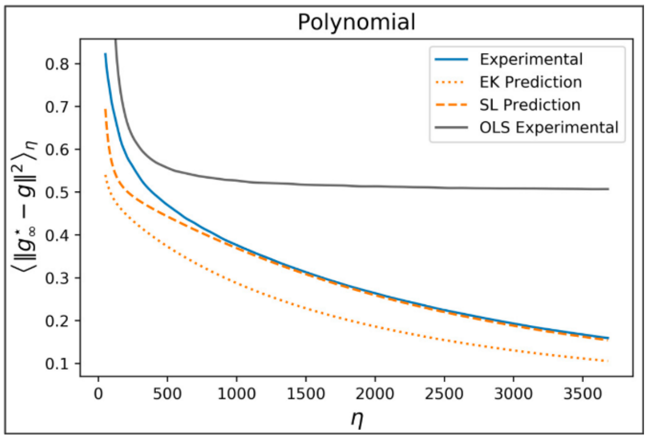

Figure 4: Experimental versus predicted learning curves [1]

training data as a metric of size, treating this as the independent variable in the study, and compares this number with accuracy, calculated as the proportion of true positives out of all the labels made by the model. It presents the results in terms of two relationships: between epoch and accuracy, and between size and accuracy. Finally, it discusses ways to improve robustness of the results gathered.

3.2 Data set

Exploratory observations are carried out on the visualization software CloudCompare on the Vaihingen data set as well as Sunset Park data set collected by the Urban Modelling Group (UMG) at the Center for Urban Science and Progress. The main performance analysis uses the ModelNet data set consisting of CAD scans of objects spread across 40 categories. The distribution is shown in Table 1.

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

Table 1: ModelNet object distribution

3.3 Model architecture and implementation

Figure 5 displays a diagram of the PointNet classification architecture.

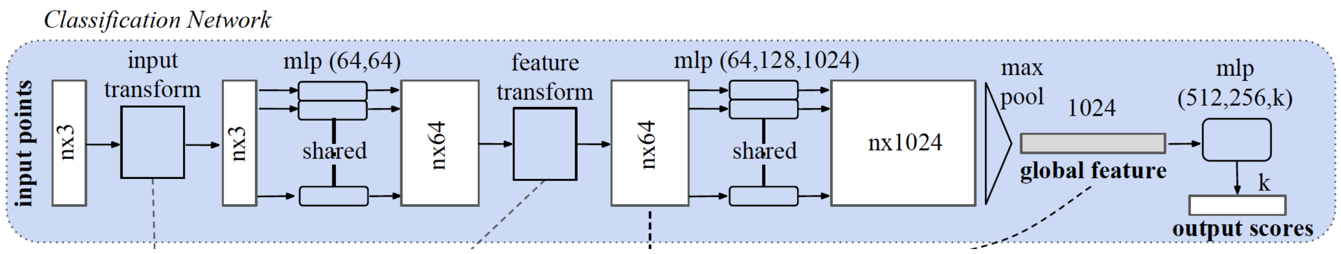

Figure 5: PointNet classification architecture [5]

PointNet's architecture for classification receives an input of 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, allowing for an informed classification [5] The output is a number of classification scores, one for each class that the label could be chosen from. Since 40 categories of objects are present, the model will output 40 scores for each of the 400 test objects, choosing the label with the highest score.

This project builds off a Pytorch implementation of PointNet developed by Yan et. al. [10].The implementation takes an input of a list of text files, where each file represents an object and contains the points that make up the object. It implements PointNet in Pytorch, and in every run of classification training, it outputs a pth file containing the configuration of the model and a logfile consisting the model's performance on training data, overall and class performance on test data, and time stamps.

3.4 Preprocessing

3.4.1 Preliminary analysis. Four size-related features in the training data were originally identified: number of points, number of overlapping points across objects, the variation of number of points in each object, and number of objects. The number of objects was chosen as the feature to focus on because of the semantic value it has. It also matched the format of data necessary for the PointNet implementation as described above, which trains on individual files containing a collection of points that each represent an object.

Manual analysis was performed on the Sunset Park data set by visualizing it in CloudCompare. This revealed that there are further elements to the number of objects that could be explored. For instance, cars included both moving and stationary cars, and stationary cars included both cars stopped at a traffic light or parked cars. An example of this difference can be seen in Figure 6.

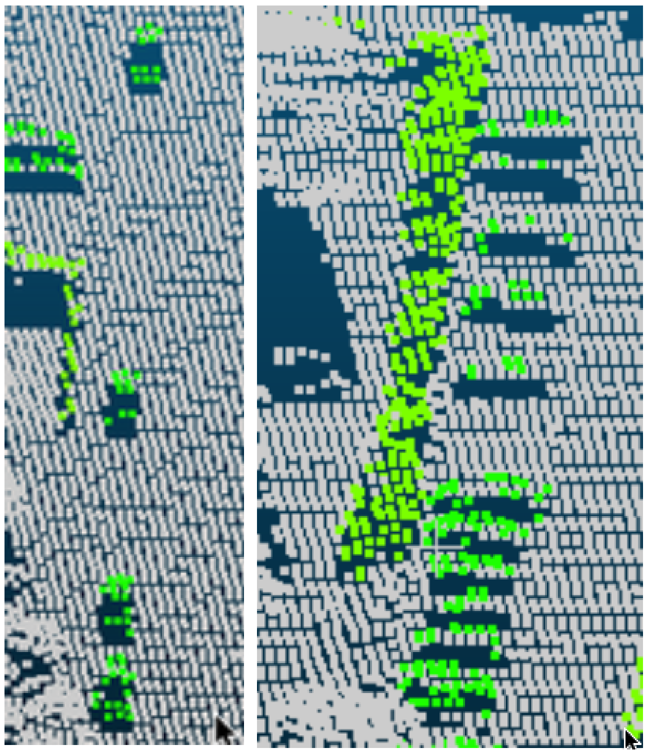

Figure 6: Moving cars (left) and parked cars (right)

3.4.2 Segmentation. A schedule was devised listing the progression of training sizes as follows: 40, 80, 120, 240, 320, 400, 600, 800, 1000, 2000. The distance between the larger training sizes is greater because as the accuracy results were plotted for each size, it was perceptible that the rate of improvement was slowing. Using larger data set allowed for acquisition of more insights on the shape of the learning curve without having to train on the intermediate sizes.

For each size n in the training size list, n/40 objects from each category were taken for training. The last 100 objects from each category were taken for testing for all the input sizes. Both the training and testing data sets were balanced across all 40 categories for every round. The data was segmented in this manner with Python scripts.

The code base of the PointNet implementation was modified to create the ability to use data of different sizes for training, instead of only a single default size.

3.5 Training

The runs with 40, 80, 120, 240, 400, and 600 objects were trained on 100 epochs. Following the visualization of these results, the subsequent runs with 800, 1000, and 1200 objects were trained on 120 epochs to see if increasing the number of epochs would affect the pattern of the learning curve over epochs.

At each epoch, the model's accuracy on classifying the training data and both the overall accuracy as well as category-specific accuracy on classifying the test data was recorded. If the accuracy on test data for a given epoch exceeded the highest saved accuracy so far, the results for that epoch were saved as the best instance accuracy.

PointNet was used with the parameter configurations shown in 2 for all runs.

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 Raw data

Table 3 displays a section of the raw results obtained, displaying the accuracy on each training data set size at every 10 epochs. These numbers are processed and analyzed in the following sections.

4.2 Accuracy versus training iterations

The graph in Figure 7 illustrates the results for each run plotted against epoch. The line for training data size 40 is significantly visually different than the others because of its flat start, indicating a slow start in model's learning. Indeed, the log file containing the precise values shows that the model consistently has an accuracy of 0.024510 on unseen data for the first 37 epochs, and only at the 38th epoch does it start increasing.

The general shape of the curves shows that as training size increases, there is a decelerating increase in accuracy. We can also estimate that rate of learning starts evening out

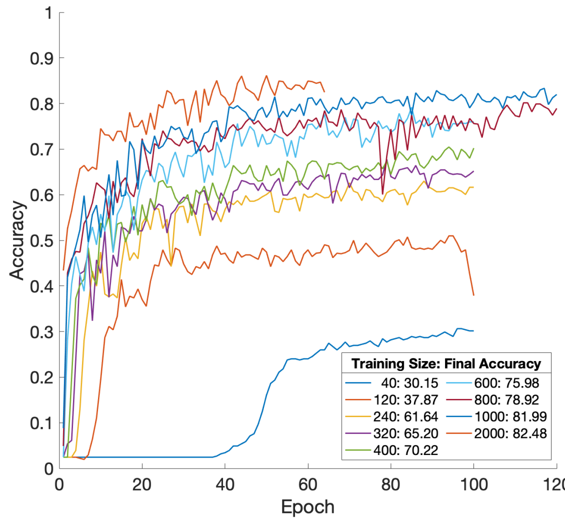

Figure 7: PointNet accuracy versus epoch

at approximately 20 epochs for all training sizes. For size 40, whose performance started improving only after epoch 37, the line starts to plateau near 57, corresponding to only 20 epochs of growth similar to the other plots.

The line for 800 features a sharp drop near epoch 79, which was caused by interruption of the training due to the machine crashing. The model, however, resumed training from a saved checkpoints and seems to reach the same peak accuracy value as before the interruption.

The plots of raw accuracies contain many overlaps especially in the earlier stages of training. To improve the visibility of the individual lines and allow for clearer comparison between them, the moving mean of the accuracy can be taken.

Figure 8 plots the same points but with a moving mean window of 10. With the smoothing of the curves, they more resemble the shape of the curve in the example of Figure 2. The gaps between the curves also reveals that for the most part, the model's performance has a greater jump when training data size is increased at lower values, and a smaller jump when training data size is increased at higher values. For example, the last data size of 2000 is double the value of the preceding size, and this interval is the largest. Yet the gap between this pair of curves is not significantly bigger than others. The training data sizes are not evenly spaced apart, so we need to account for this before making more detailed claims about the differences.

4.3 Learning curve

To account for the varying interval sizes in training data, the best instance accuracy for each run was then extracted and

Accuracy on Training Set Size									
Epoch	40	120	240	320	400	600	800	1000	2000
1	0.02451	0.02451	0.02451	0.02451	0.02451	0.02451	0.04902	0.088235	0.433824
10	0.02451	0.193627	0.474265	0.427696	0.53799	0.557598	0.617647	0.506127	0.745098
20	0.02451	0.370098	0.536765	0.520833	0.578431	0.632353	0.723039	0.719363	0.76348
30	0.02451	0.448529	0.574755	0.594363	0.585784	0.63848	0.724265	0.726716	0.751225
40	0.034314	0.471814	0.567402	0.590686	0.607843	0.659314	0.723039	0.732843	0.795343
50	0.160539	0.474265	0.60049	0.610294	0.664216	0.710784	0.740196	0.769608	0.834559
60	0.240196	0.479167	0.594363	0.61152	0.666667	0.753676	0.764706	0.811275	0.848039
70	0.267157	0.477941	0.626225	0.615196	0.654412	0.738971	0.740196	0.800245	
80	0.281863	0.485294	0.609069	0.639706	0.644608	0.73652	0.743873	0.801471	
90	0.291667	0.480392	0.612745	0.643382	0.685049	0.72549	0.757353	0.803922	
100	0.301471	0.378676	0.616422	0.651961	0.702206	0.759804	0.756127	0.803922	
110							0.773284	0.817402	
120							0.789216	0.819853	

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

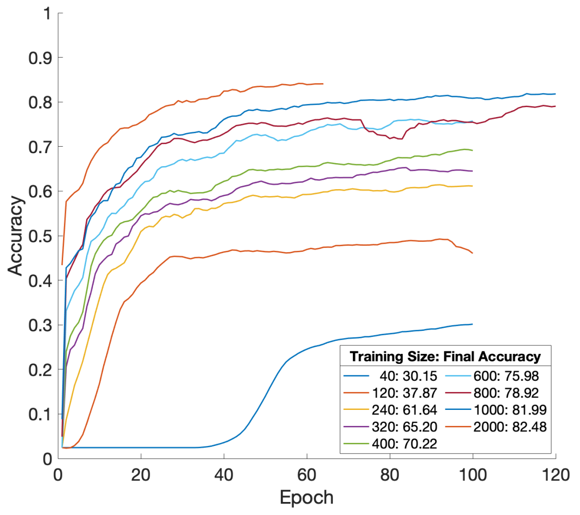

Figure 8: PointNet moving mean accuracy versus epoch

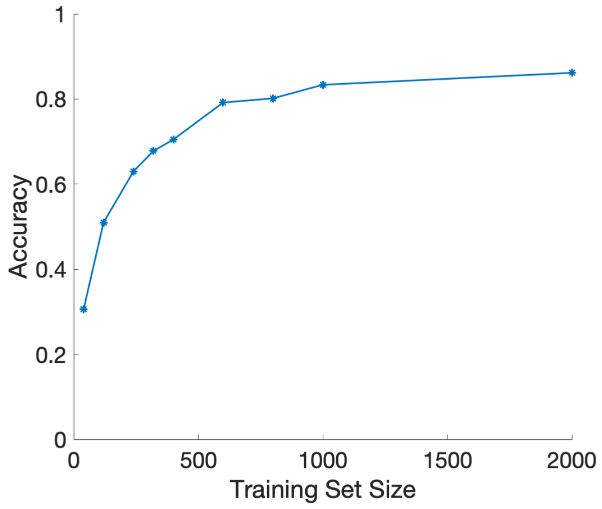

Figure 9: PointNet accuracy versus training dataset size

graphed. Figure 9 illustrates the relationship between training size and the best instance accuracy. This graph resembles a logarithmic growth curve. The accuracy starts plateauing after a training set size of 600. The 600 size was trained on with 100 epochs while the 800 size was trained on with 120 epochs. Looking at Figure 8, it is apparent that performance on the 600 size and 800 size plateau to almost the same value at 100 epochs, and the performance for 800 size increases shortly after. Even before the outlier caused by interruption in size 800, the curve for size 600 approaches it very closely. This suggests that the improvement in classification accuracy after 600 is small compared to the increase in training size.

This curve fits a logarithmic function as shown in Figure 10, quantifying a decaying rate of learning.

5 DISCUSSION

These results are in accordance with the typical shape of learning curves found in [4] and [1]. However, this pattern needs to be ascertained by training on larger sizes, using different perturbations of the objects in the training and test set, and applying it to different data sets altogether. Since

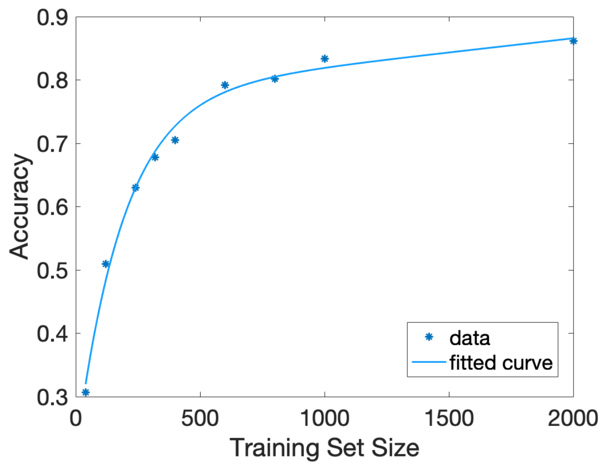

Figure 10: PointNet accuracy versus training dataset size with logarithmic curve fitting

the likelihood of overfitting increases with larger training sets, it is worthwhile to see if this point is reached with these other methods.

The results show a relationship of logarithmic accuracy growth as the number of objects in the the training set increase. This relationship can be examined in more complex detail in the manner undertaken in [1] with more data points. One such way is to use a Poisson process to understand the distribution of points and predict the learning curve.

The literature also discussed the differences in aerial and close range LiDAR data in [8] and [11], such as aerial data having lower and more inconsistent density per volume of an object. Although the data used in this work was close range, the techniques may be applied to aerial data with further experimentation. Aside from extracting objects from aerial data and testing how well the model classifies the, the training data can be manipulated to resemble aerial data and examine the accuracy in the same way.

6 CONCLUSION

The novelty of neural networks dedicated to LiDAR means that to date, most effort has been spent on actually developing new networks rather than evaluating the characteristics of their training. As such, many fundamental questions on how performance changes with changes in input data remain unexplored, a major one being quantity of data. This research project addresses this gap by assessing the impact of data quantity on the classification accuracy of PointNet, and is thus a theoretical foundation for further analysis of input data changes and their impact on model performance. We find that the model's rate of learning in terms of accuracy improvement per epoch is not affected by the size of the training data set. All sizes exhibited a slow in accuracy improvement following epoch 20. The objects in the training data were found to each contain 10,000 points. The learning curve starts plateauing at a training set size of 600, with a very small increase from 1000 to 2000, meaning the corresponding number of points at which improvement slows drastically is 600,000.

These results should be considered a preliminary step in understanding the nature of training data best suited for optimizing LiDAR-focused classification models, based upon which further work can make the current landscape in machine learning for LiDAR more cohesive. As future work, the model should continue being trained on larger sizes of data to see how the curve moves over more training samples, as well as on different combinations of object samples in the training and test data sets. This is crucial in knowing whether the fit applied on the results so far is true to the data.

The classification errors at the end of every stage can also be analyzed to identify specific characteristics of objects that are misclassified, for example being stationary or moving, partial obstruction, or orientation. Additionally, if patterns of misclassification are revealed to exist, they can be used to test and correct imbalance in the training data.

Additionally, the implementation's performance with direct aerial data should also be further thoroughly examined, since its simplicity and accessibility make it a convenient tool for classification. Each of the objects used in training contained 10,000 points, so the training data was balanced unlike most aerial data. It is worth examining the effect of uneven distributions in both the number of points per object and number of objects per category. Another way that the performance with aerial data can be evaluated is using the density variation technique from [11]. Since indoor, near objects have a more defined shape and a higher number of points per unit volume compared to aerial data, the model would be less accurate on sparse data sets if it were trained on denser ones. Thus, it may be worth investigating the effect of deprecating density in training data on the model's accuracy on aerial data sets.

7 ACKNOWLEDGEMENTS

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. The code in this project modified and appended to the original implementation in [10] and can be found at https://github.com/0orja/Pointnet_Pointnet2_pytorch along with original logs.

REFERENCES

[1] Omry Cohen, Or Malka, and Zohar Ringel. 2021. Learning curves for overparametrized deep neural networks: A field theory perspective. *Physical Review Research* 3, 2 (Apr 2021). https://doi.org/10.1103/physrevresearch.3.023034

[2] Guus Engels, Nerea Aranjuelo, I. Arganda-Carreras, M. Nieto, and O. Otaegui. 2020. 3D Object Detection From LiDAR Data Using Distance Dependent Feature Extraction. In *VEHITS*.

[3] Zhizhong Kang and Juntao Yang. 2018. A probabilistic graphical model for the classification of mobile LiDAR point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing* 143 (05 2018), 16. https://doi.org/10.1016/j.isprsjprs.2018.04.018

[4] Claudia Perlich. 2011. Learning Curves in Machine Learning. (01 2011). https://doi.org/10.1007/978-0-387-30164-8_452

[5] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. arXiv:cs.CV/1612.00593

[6] Marco Scaioni, Bernhard Höfle, A. Kersting, Luigi Barazzetti, M. Previtali, and Daniel Wujanz. 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 (04 2018), 1503–1510. https://doi.org/10.5194/isprs-archives-XLII-3-1503-2018

[7] Wei Song, Shuanghui Zou, Tian Yifei, Simon Fong, and Kyungeun Cho. 2018. Classifying 3D objects in LiDAR point clouds with a back-propagation neural network. *Human-centric Computing and Information Sciences* 8 (12 2018). https://doi.org/10.1186/s13673-018-0152-7

[8] Cc Wen, Yang Lina, Xiang Li, Ling Peng, and Tianhe Chi. 2020. Directionally constrained fully convolutional neural network for airborne LiDAR point cloud classification. *ISPRS Journal of Photogrammetry and Remote Sensing* 162 (04 2020), 50–62. https://doi.org/10.1016/j.isprsjprs.2020.02.004

[9] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3D ShapeNets: A deep representation for volumetric shapes. 1912–1920. https://doi.org/10.1109/CVPR.2015.7298801

[10] Xu Yan. 2019. Pointnet/Pointnet++ Pytorch. *https://github.com/yanx27/Pointnet_Pointnet2_pytorch* (2019).

[11] Mohammed Yousefhussien, David J. Kelbe, Emmett J. Ientilucci, and Carl Salvaggio. 2018. A multi-scale fully convolutional network for semantic labeling of 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing* 143 (2018), 191 – 204. https://doi.org/10.1016/j.isprsjprs.2018.03.018 ISPRS Journal of Photogrammetry and Remote Sensing Theme Issue “Point Cloud Processing”.