Point Cloud Subsamping Optimization for Minor Class Retention

Rajarshi Bhattacharjee, 20349, Economic Sciences

Abstract: Our project proposes an approach to optimize point subsampling for minor class retention by combining subsampling techniques through ML techniques, such as Decision Trees and Genetic Algorithms. Our approach attempts to predict the required subsampling values under various subsampling techniques to achieve a desired value of point cloud retained, outperforming existing techniques. This enables the processing and storage of point cloud data with reduced computational and storage requirements without sacrificing information from minor classes.

Keywords: Minor Classes Retention, Cloud Subsampling, Machine Learning, CloudCompare

1. Introduction

Point clouds are an essential representation of 3D environments, and they have become increasingly popular in various applications, such as autonomous vehicles, robotics, and augmented reality. Point clouds can contain millions or billions of points, making them challenging to process efficiently. Therefore, subsampling algorithms are used to reduce the size of point cloud data by selecting a subset of points while retaining important geometric features. However, subsampling algorithms can introduce biases in the data, particularly towards certain classes, which can impact the performance of downstream tasks. Therefore, this report proposes a novel approach that uses machine learning algorithms to predict the effective class bias of a combination of subsampling algorithms used in sequence, investigating the impact of different ratios of sequential point removals using multiple subsampling algorithms on the class distribution of the data. Specifically, we investigate the impact of different ratios of sequential point removals using multiple subsampling algorithms on the class distribution of the data.

Subsampling algorithms for point clouds can be classified into two types: random sampling and non-random sampling. Random sampling involves selecting points randomly from the original dataset, while non-random sampling involves selecting points based on some criteria, such as distance or density. Non-random subsampling algorithms can be further divided into two categories: voxel-based subsampling and point-based subsampling.

Voxel-based subsampling or space-based subsampling involves dividing the point cloud into a grid of voxels and selecting one point from each voxel. This method can introduce biases towards major classes, as voxels containing a large number of points are more likely to be selected. Point-based subsampling involves selecting points based on their distance to a reference point or set of points. This method can introduce biases towards minor classes, as points that are far away from the reference points are more likely to be selected.

An issue with these subsampling techniques is that the subsampled data is generally biased towards major classes as it is easier for minor classes to get sampled out due to the lower number of minor classes present in the cloud sample. This paper attempts to solve this problem by developing a machine learning algorithm that, when fed in with the required test data, can generate the required subsampling levels to obtain the desired number of minor classes in the subsampled cloud.

1.1 Literature review

Point cloud subsampling has been an active area of research in recent years, with several studies proposing different algorithms to overcome the challenges of processing large point cloud datasets efficiently. Several approaches have been proposed to address these biases, including using machine learning algorithms. [1] aims to extend a machine learning (ML) classification method with a multi-level and multi-resolution (MLMR) approach to subsample 3D point cloud data through a hierarchical approach depending on geometric information.

Recent work by [2] reviews point cloud semantic segmentation algorithms of region growing, model fitting, unsupervised clustering, supervised machine learning, and deep learning with their advantages, disadvantages, and specific applications in the cultural heritage field.

[3] proposes an approach that involves removing between-class data point imbalances and guiding the network to pay more attention to majority classes. Data imbalance is alleviated using a hybrid-sampling method involving oversampling, as well as undersampling, respectively, to decrease the amount of data in majority classes and increase the amount of data in minority classes.

Overall, there have been several approaches proposed for point cloud subsampling, with machine learning algorithms playing an increasingly important role in addressing biases in the data.

2. Setup

The project involves the use of three subsampling scripts: random_sampler.py, space-based_sampler.py and voxel-based_sampler.py along with a file named point_cloud_utils.py to read the cloud files. All the scripts are in Python language. The data provided is described in a dedicated section below. CloudCompare and a code editor are the requirements for running the given files. Along with the softwares,, certain Python libraries must be installed in the system: NumPy, pandas, tqdm, random, open3d, multiprocessing, subprocess, CSV, argparse and sklearn. These will be provided in a requirements.txt file.

2.1 Data

The data provided is the LiDAR Dayton Annotated Laser Earth Scan (DALES) data set provided by the University of Daytona, Ohio. 20 clouds are provided visualizing different geographic locations. The dataset is in CSV format and is a collection of point cloud files with the x, y, z, coordinates and scalar features of intensity and a label (0-9) that is the class to which the point belongs. The data provided contains points from varying classes. Labelled point clouds have a class label assigned to every point in the cloud. Some of these classes have a lot of points ("ground", "vegetation"), while others can have a very low number of points ("wire", "pole", "person"). Classes that have more points are termed major classes, while those with a low number of points are termed minor classes.

For the purpose of the project, 5 datasets are chosen at random from the given 20 datasets to generate the test data.

2.2 Training-set

The training set is generated by running multiple iterations for different sampling levels for all three scripts, one after the other, in a process described in the methodology. The training set is developed through the use of 5 of the cloud samples through 5000 iterations using the following constraints:

> Random_sampler: Can take any value of loss level and subsample the data
> Space-based and Voxel-based sampler: Can take loss levels only in the range of (12.5, 22.5, 32.5, …, 92.5) due to compilation limitations.

A sample of the training dataset

random_sampler_compression	space_based_compression	voxel_based_compression	percentage_of_minor_classes_retained
0.74	0.925	0.125	0.59
0.39	0.725	0.725	0.56
0.34	0.425	0.825	0.5
0.42	0.225	0.425	0.38
0.63	0.425	0.625	0.59
0.39	0.925	0.725	0.6
0.33	0.125	0.325	0.29
0.76	0.125	0.225	0.47
0.76	0.325	0.725	0.66
0.37	0.625	0.125	0.35

2.3 Additional Programs

An analyzer program is designed to run on the unprocessed and then the final processed cloud to find the percentage of minor classes retained, denoted by m, using the labels provided. This

calculates the number of points with labels of (5, 6, 7, 8) in both the processed and unprocessed files and returns a value m = no of points in processed/ no of points in the unprocessed file.
An automation program is also designed to run the codes, as described below, and then divided into 64 processes to speed up the training data generation.
A program is also built to run on the clouds and return an ascending order of points pertaining to all label values to determine which labels are used for the minor classes (labels with lowest number of points are for minor classes).

3. Methodology

The methodology proposed in this research paper aims to develop a method for point cloud data compression and subsampling while retaining the minor classes for further analysis. The proposed method involves a combination of various subsampling techniques and machine learning algorithms to achieve the desired results. The following steps describe the proposed methodology in detail.

3.1 Visualization and Identification of Minor Classes

The first step involves visualizing the given point cloud dataset using CloudCompare, a point cloud processing software. The goal is to identify the minor classes within the dataset, which are usually less prevalent but may still be of significant importance. These classes will be retained during the compression and subsampling process.

3.2 Script Modification and Execution

The next step involves modifying the provided scripts to suit the environment and creating the necessary directories. Two directories, "Sample" and "Output," need to be created. The scripts need to be run, considering the desired compression level, which represents the percentage of points to be removed.

3.3 Subsampling Techniques

After obtaining a compressed point cloud file, various subsampling techniques are employed in the following order: Random Sampler, Space-Based, and Voxel-Based. The resulting point cloud file after the third subsampling technique will be used for analysis.

3.4 Analysis and Generation of Combinations

The compressed file obtained after the subsampling techniques is analyzed using point cloud analytics to determine the number of minor classes retained in the file. The goal is to determine the percentage of minor classes retained after all three subsampling techniques. The analysis results in a string of (x, y, z, m), where x represents the percentage subsampled for the random

sampler, y represents the percentage subsampled for the space-based sampler, z represents the percentage subsampled for the voxel-based sampler, and m represents the percentage of minor classes retained after all three subsampling techniques. Multiple combinations of (x, y, z, m) are generated by running the scripts on multiple percentage values, with the range of (30, 80) suggested.

3.5 Python Script Development

To expedite the process of generating combinations, a Python script is developed that runs the random sampler, space-based, and voxel-based scripts on a file for multiple values of (x, y, z). The script also incorporates a cloud analyzer that determines the value of m for each combination. The resulting strings of (x, y, z, m) are saved in a CSV file.

The automation script is divided into 64 subprocesses to speed up the action, which otherwise is estimated to take 5 days to generate a test set of 5000 samples.

3.6 Machine Learning Model Development

In the next step, a machine-learning model is developed using the CSV file generated in the previous step. The model's purpose is to predict a set of (x, y, z) values that would result in the desired m value. Multiple machine learning models, such as linear regression, random forest, decision trees, and genetic algorithms, can be used. For this research, a decision tree model is used. The model is trained on the generated CSV file, and the code is modified to take a desired m value as input. The model then returns a set of (x, y, z) values that, when implemented in the order, would result in the desired m value.

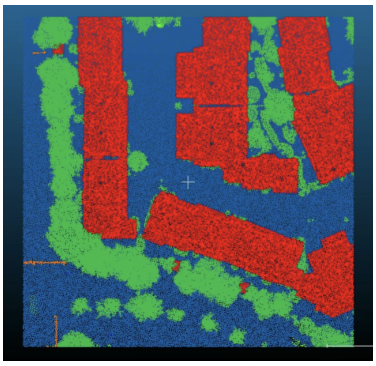
3.7 Testing and Validation

The final step involves testing the proposed method on real-life point cloud data to verify its accuracy. The process involves applying the machine learning model to the data and obtaining the predicted (x, y, z) values. These values are then applied in the order suggested by the model to obtain the desired m value. The accuracy of the results is then verified by comparing them with the actual number of minor classes retained in the compressed point cloud file.

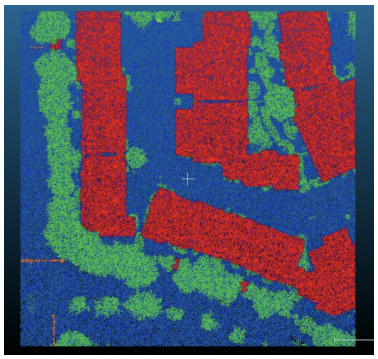
4. **Results and Conclusion**

Raw and subsampled data visualisations are shown for (x, y, z, m) as (0.65. 0.625, 0.625, 0.64). The samples are created in CloudCompare software.

The analyser gives (5, 6, 7, 8) to be the values used for minor classes. The subsamplers run on specified loss levels as described above to give multiple values of m. As observed, all values of m lie in the range of 0.3 to 0.8, which is 30% of minor classes retention to 80% retention.



Cloud without loss



Cloud with 64% minor class retention

The proposed methodology presents a comprehensive approach to compressing and subsampling point cloud data while retaining the minor classes for further analysis. The method combines various subsampling techniques and machine learning algorithms to achieve the desired results. The first step involves identifying the minor classes within the point cloud dataset, which are usually less prevalent but may still be of significant importance. The scripts are then modified and executed to obtain a compressed point cloud file, which is subsampled using various techniques, namely the Random Sampler, Space-Based, and Voxel-Based, in a particular order. The compressed file is then analyzed using point cloud analytics to determine the number of minor classes retained in the file. Multiple combinations of (x, y, z, m) are generated using a Python script, and a machine learning model is developed to predict a set of (x, y, z) values that would result in the desired m value.

The proposed methodology was tested on real-life point cloud data to verify its accuracy, and the results were compared with the actual number of minor classes retained in the compressed point cloud file. The proposed method presents an effective approach to point cloud data compression and subsampling while retaining the minor classes for further analysis.

4.1 Additional Information

The data has been uploaded to a GitHub repository, and the ML code has been edited to run on any system.
For all the other files, the directories and bins need to be modified according to the system. Libraries necessary for running these projects are mentioned in the requirements.txt file bundled with the code zip. To access the files and the data, visit my repository **here**.

5. **Bibliography**

[1] Teruggi, S., Grilli, E., Russo, M., Fassi, F., & Remondino, F. (2020). A Hierarchical Machine Learning Approach for Multi-Level and Multi-Resolution 3D Point Cloud Classification. *Remote Sensing*, *12*(16), 2598. **https://doi.org/10.3390/rs12162598**

[2] Yang, S., Hou, M., & Li, S. (2023). Three-Dimensional Point Cloud Semantic Segmentation for Cultural Heritage: A Comprehensive Review. *Remote Sensing*, *15*(3), 548. **https://doi.org/10.3390/rs15030548**

[3] Lin, H., & Nguyen, M. C. (2020). Boosting Minority Class Prediction on Imbalanced Point Cloud Data. *Applied Sciences*, *10*(3), 973. **https://doi.org/10.3390/app10030973**