UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Project Final Paper 2022

Title:

Semantically Meaningful Clustering of Cultural Heritage Point Cloud Data using Hierarchical Methods

Author:

Jared May

Project Abbreviation:

CHSEG

Supervisor(s):

Prof. Patrick Marais, Luc Hayward

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	20
System Development and Implementation	*0*	*20*	10
Results, Findings and Conclusions	*10*	*20*	20
Aim Formulation and Background Work	*10*	*15*	10
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	0
Total marks		**80**	**80**

Semantically Meaningful Clustering of Cultural Heritage Point Cloud Data using Hierarchical Methods

Jared May*
MYXJAR002@myuct.ac.za
University of Cape Town
Cape Town, South Africa

ABSTRACT

The preservation of Cultural Heritage (CH) sites is vital in ensuring the transmission of past human activity to future generations. The 3D point clouds obtained in the digital preservation of these sites contain a wealth of information such as structural information, but also erroneous data such as noise, people, scaffolding, and other natural objects. The supervised labelling and cleaning of this data is expensive in terms of time and labour.

This project explores the application of hierarchical clustering methods to investigate the ability of these clustering methods to produce semantically meaningful clusters on 3D point cloud data of CH sites. Hierarchical clustering methods are compared against a baseline of K-Means clustering. This project is experimental in nature with algorithms being tested on multiple feature sets of differing dimensionalities.

The results from the Hierarchical clustering algorithms produce promising results in their ability to produce semantically meaningful clusters. These algorithms also perform as well if not better than K-means clustering in multiple methods. These clusters could be used to improve tasks such as point cloud cleaning.

CCS CONCEPTS

• **Computing methodologies** → **Unsupervised learning**; *Cluster analysis*; *Feature selection.*

KEYWORDS

unsupervised machine learning, clustering, hierarchical clustering

1 INTRODUCTION

The digital preservation of Cultural Heritage (CH) sites is a vital means of transmitting a testimony of past human activity to future generations. The Visual Computing Group (VCG) at CNR-ISTI [1] and the Zamani Project at UCT [11] construct 3D models from point clouds acquired from 3D laser scanning. These models can be used for CH preservation and study since preservationists can use these models to decipher where a site needs preservation or use these models for education of future generations. However, the process of acquiring and processing point cloud data into accurate and useful 3D models is not simple. Terrestrial Laser Scanning (TLS) of CH sites, results in point clouds that contain both wanted and unwanted data, such as foliage, scaffolding, people, and animals that do not belong to the CH site and which must be removed. This removal process - point cloud cleaning – is currently a mainly manual, labour intensive, and time-consuming process. Approaches to automate and speed up the cleaning process have included: the application of supervised machine learning methodologies to classify data into keep and discard labels [4], and the application of unsupervised machine learning to denoise data [cite].

This project explores additional experimentation into the unsupervised segmentation of point cloud data. Through experimentation this project aims to investigate the performance of unsupervised clustering methodologies in the decomposition of point clouds into semantically meaningful clusters of 3D points. Ideally, such clusters will correspond to point collections that make sense to a human, such as noise, or all the points sampling a tree in the 3D scene. Accomplishing this step could significantly simplify and accelerate the arduous task of point cloud cleaning. In addition, the project aims to assess whether or not clustering can augment the classification process. The resulting clusters will be assigned a binary label, using the majority ground truth label of points per cluster, to form groupings that can be classified using binary classification labels of '*keep*' and '*discard*'.

Hierarchical clustering is of special interest to this project. Hierarchical clustering methodologies include BIRCH [14], CURE [3], and Agglomerative clustering.

These clustering methods show promise in their ability to cluster spatial data sets. Hierarchical clustering algorithms build hierarchical clusters by merging (Agglomerative methods) or dividing (Divisive methods) clusters successively. Hierarchical clustering algorithms are known for producing good separation of clusters.This hierarchy of clusters is represented as a tree or dendrogram. The root of the tree is the unique cluster that gathers all the samples, the leaves being the clusters with only one sample.

1.1 Overview

This paper consists of: a Background and Related works section, introducing some of the more domain specific topics. An Experiments section covering the experimental aims of this project, experimental design, evaluation metrics, and project specific constraints. The System Design and Implementation section covers the means by which the experiments were implemented and performed. The Results section gives a description of the results of this project. The Discussion section is where those results are discussed. The Conclusions section provides a summary of the results and discussion and whether they meet the experimental goals of this project.

2 BACKGROUND AND RELATED WORK

2.1 Point clouds

3D point clouds are a datatype comprised of a set of points in euclidean space each defined by three spatial coordinates (x, y, z). Point clouds may include additional labelling - features — which

*Research was conducted in parts as part of a research group.

can be basic descriptors — such as intensity to laser or RGB colour, which describe useful characteristics of points or point neighbourhoods. Point clouds can be aquired through laser scanning such as the data set in this project or by other means such as aerial photogrammetry.

2.2 Clustering

Clustering is an unsupervised learning approach. It is the unsupervised grouping of similar or related data into groups or clusters. This clustering algorithm provides the criterion to cluster data together. These criteria can be the distance numerically of data points on some metric be it distance based or based on numeric. Hierarchical clustering methods produce hierarchical clusters by merging or splitting groupings based on criterion set by the algorithm, be it average relationship between points or point representors. Hierarchical clustering algorithms build hierarchical clusters by merging (Agglomerative methods) or dividing (Divisive methods) clusters successively. This hierarchy of clusters is represented as a tree (or dendrogram). The root of the tree structure is the unique cluster that gathers all the samples, the leaves being the clusters with only one sample.

2.3 Classification

Classification is the supervised labelling of data points as belonging to one of two (binary classification) or more then two (classification) classes. This can be as is the case in this paper - keep or discard - but could also be anything else such as colour, wall, or person. These labels are fit to the data using the application of a model for unseen data or a set of predetermined ground truth values such as in this paper. This paper considers whether the majority of points in a particular cluster fall into keep or discard labels.

2.4 Clustering algorithms used in this project

2.4.1 K-Means Clustering. K-means clustering is a clustering method that aims to partition n data points into k clusters in which each data point belongs to the cluster with the nearest mean (cluster centroid), which serves as a representor of the cluster. K-means clustering minimizes within-cluster variances (squared Euclidean distances), but not regular Euclidean distances. Better Euclidean solutions can be found using k-medians and k-medoids. The pure k-means problem is NP-hard, however heuristics are used to converge quickly to a local optimum. K-means clustering is often used on data sets containing spatial information, such as point clouds.

2.4.2 Agglomerative Clustering. The Agglomerative Clustering algorithm is a hierarchical clustering method using a bottom up approach. Each data point or observation starts in its own cluster, and clusters are successively merged together. It considers at each step all the possible merges. This project uses a precomputed k-nearest neighbour (knn) graph to reduce this complexity and improve the performance of the algorithm. Agglomerative clustering can also be extremely spatially expensive without optimisation such as connectivity constraints.

Agglomerative Clustering as mentioned can scale to large number of samples when it is used jointly with a connectivity matrix, but is computationally expensive when no connectivity constraints are used.

The linkage criteria determines the metric used for the merge strategy: One such criteria and the one used in this project - Ward - minimizes the sum of squared differences within all clusters. Other linkage criteria are - maximum or complete linkage - minimizes the maximum distance between observations of pairs of clusters, average linkage - minimizes the average of the distances between all observations of pairs of clusters and single linkage - minimizes the distance between the closest observations of pairs of clusters.

2.4.3 BIRCH Clustering. The Birch algorithm [14] uses a tree structure to decide on when to merge clusters. It builds a tree called the Clustering Feature (CF) Tree. This is made up of CF Nodes. The CF Nodes have a number of subclusters - CF Subclusters and these CF Subclusters can have CF Nodes as children. The CF Subclusters hold the necessary information for clustering which prevents the need to hold the entire input data in memory. This helps keep down Birch's space complexity. The BIRCH algorithm has two parameters, the threshold and the branching factor. The branching factor limits the number of subclusters in a node and the threshold limits the distance between the entering sample and the existing subclusters. Birch reduces the data to improve performance.

2.4.4 CURE Clustering. The CURE clustering algorithm [3] was proposed as an efficient hierarchical clustering algorithm for large data sets such as databases, an improvement on Agglomerative clustering. CURE uses a processing pipeline of partial clustering and then full clustering this reduces the data set and number of clustering decsions. It has an expensive time complexity of $O(n^2 log(n))$ in the worst case. It has two hyper parameters compression and number of representative points which are used for decisions of when to merge clusters.

2.5 Cultural Heritage preservation and 3D Point Cloud Cleaning using Machine Learning

Researchers have explored Point Cloud Classification through various approaches that aim to classify 3D points in laser scans as either relevant or not, such as Marais et al. [5]. Supervised learning methods could be used to speed up the process of point cloud cleaning in the field of digital cultural heritage preservation. Examples of cultural heritage preservation through the acquisition of large 3D point cloud data can be seen in the work of the Zamani project [11].

3 EXPERIMENTATION

3.1 Experimental Aims

The central research question of the project is how well are unsupervised hierarchical clustering algorithms able to produce semantically meaningful clusters of 3D point cloud data on cultural heritage sites. Semantically meaningful meaning clusters adhere to groupings of data that are relevant and understandable spatially and structurally in the 3D point cloud or which show promise in their coverage and separation of the data set for binary classification (keep/discard labelling) for use in applications such as 3D point cloud cleaning. Specifically, the main research question focuses itself on how well popular hierarchical clustering methodologies - BIRCH, CURE and Agglomerative Clustering - compare to k-means clustering as a baseline and common clustering algorithms and each

other in terms of their ability to produce semantically meaningful clusters. This is evaluated using their comparative performance in cluster evaluation and binary classification metrics.

Secondarily, we aim to determine if feature selection such as the use of extracted geometric/covariance features, or very high dimensional feature sets (such as those output by the Pointnet++ deep neural network) improve the performance of hierarchical clustering algorithms in producing semantically meaningful clusters.

Finally a tertiary point of interest is if the application of hierarchical clustering algorithms could be viable as a means to improve or augment the process of automated or semi-automated point cloud cleaning.

3.2 Experiment Design

The range of exploration (lower and upper bound) for which number of clusters (k) could produce semantically meaningful clusters was explored as an initial "path finding" step. This lead to a range of exploration of $100 <= k <= 750$ used by all following experiments. During the exploration of $k < 100$ and $k > 750$, preliminary tests seemed to portray volatility in the results of clustering metrics for $k < 100$. Additionally, using $k < 100$ meant poor separation of structures and the clusters produced were not fine enough to be applicable for binary classification i.e. the majority labels we generally keep. On the other hand, clustering on $k > 750$ clusters created clusters so small that separation of structures was too high. Increasing k tended towards individual clusters being semantically meaningless (indiscriminate point blobs). Deceptively $k > 750$ improved binary classification results, but this is due to majority labels of each cluster fitting increasingly closer to the ground truth labels.

With the range of exploration determined as $100 <= k <= 750$ experimentation followed under the following methodology.

A suitable level of downsampling for each algorithm A and data set D was determined by running an algorithm on $k = 100$ clusters, with a data set downsampled at $ds_{amt} = 0.05$. If this run took longer than ±2 minutes, a higher level of downsampling was chosen. 2 minutes was a rough limit set as the baseline k-means data set can produce 100 clusters in about this time on the raw data set.

The experimentation followed is such that for each hierarchical clustering algorithm (Agglomerative, BIRCH, CURE) and the baseline K-means algorithm A, and using each of the 3 data sets D suitably down sampled, A is run for $100 <= k <= 750$ clusters. After each run, the resultant clusters are labelled as either keep or discard based on the majority class of which the cluster is comprised when compared to the same set of points in the ground truth data. Clustering metrics (Davies-Bouldin Index (DB); and Rand Score) and Classification metrics (F1 Score; Intersection-over-Union (IoU) score; Precision score; Recall score; and Absolute and Squared mean error scores) are determined after each run and saved with the details of the run for later analysis.

It should be noted that the three hierarchical clustering algorithms require the selection of certain hyper-parameter values, the selection of these is described elsewhere.

3.3 Clustering Metrics

Clustering metrics were used to evaluate the quality of the unsupervised clustering algorithms used in this project. Due to the unsupervised nature of clustering, metrics such as the Davies-Bouldin index use an internal evaluation scheme, where the result is based on the clustered data itself.

3.3.1 Davies-Bouldin (DB) index: [2] Is a metric for the evaluation of clustering algorithms looking at cluster separation. The score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of intra-cluster distances to inter-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score. The minimum score is zero, with lower values indicating better clustering.

3.3.2 Rand index / Rand Measure: [10]The Rand index computes a similarity measure between two clusterings by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings.The Rand index helps determine the accuracy of clusters. It requires knowledge of true and predicted labelings, and thus is not a metric that can be used in a true unsupervised application. RI = (n agreeing-pairs) / (n pairs).

3.4 Classification Metrics

Classification metrics were used to evaluate the performance of the majority based binary classification used in this project.

3.4.1 Precision / Positive predictive value: The number of true positive results divided by the number of all positive results including false positives (or results that should have been labelled negative).

3.4.2 Recall / Sensitivity: The number of true positive results divided by the number of false negatives (or results that should have been labelled positive).

3.4.3 F1 score / F-measure: The F1 Score or F-measure is calculated using the precision and recall of the experiment. The F1 score is the harmonic mean of the precision and recall. The F1 score reaches its best value at 1 and worst value at 0.

3.4.4 Jaccard index / Intersection over Union (IoU):. The Jaccard index or Jaccard similarity coefficient measures the similarity between two sets of labels for some sample set and is defined as the size of the intersection divided by the size of the union of the two sets of labels. In other words, the measure provides a heuristic for how well predicted labels fit ground truth labels.

3.4.5 Mean absolute error (MAE:. Computes a risk metric or loss function representing the expected value of the absolute error loss. In simple terms the MAE represents the mean absolute distance of the predicted labels from the ground truth labels. The MAE is a measure of errors between paired observations i.e. data representing the same phenomenon for example predicted vs. ground truth labels on the same dataset.

3.4.6 Mean squared error (MSE:. Computes a risk metric or loss function representing the expected value of the squared (quadratic) error or loss. In simple terms the MSE represents the mean squared distance of the predicted labels from the ground truth labels. The

MSE is a measure of the quality of an estimator like the predicted ground truth labels thus its usefulness in this application.

3.5 Project specific considerations and compromises

Multiple project specific decisions and compromises were made in order to complete this project with the scope and time allowed by an honours project.

Namely, there is not a concerted focus on hyper-parameter tuning for this project. This could negatively affect results. Some effort was made to tune algorithms to run as best as possible. Constraints of the algorithms themselves come into play.

Starting with Agglomerative clustering, using ward as a linkage criterion and euclidean affinity for computing the linkage. The memory requirements and time complexity to compute the full linkage of the algorithm was impossibly high. As such a pre-computed k-nearest neighbours graph was computed on the input data and used as a pre-computed connectivity matrix for the algorithm.

Following on from the compromises for agglomerative clustering. With the fact that the library implementation of the CURE algorithm we used was not implemented to run on multiple cores, the performance in its worst case was very low, with a time complexity of $O(n^2 logn)$. This meant downsampling as much as 0.350 for the Pointnet++ data set on CURE.

Overall every result was run on a downsampled data set. As mentioned, this was to make sure each algorithm ran in about 2 minutes on the system used for this research Table 4. The specific voxel downsampling criteria for each algorithm and data set combination are described in Table 3

4 SYSTEM DESIGN AND IMPLEMENTATION

The system developed to conduct experiments for the project consists of a number of scripts and classes, written in Python. Functionally the system consists classes for data acquisition; data processing; clustering; classification; evaluation; experimentation; and graphing. There are also many helper scripts and classes, that provide the automated running of experiments; additional functionality to create Pointnet++ datasets; clean or manipulate datasets; access or update result data and more.

4.1 Experimentation and development platform

Python 3.6.13 was used as the programming language of the project due to its ease of use as a scripting language and power and simplicity in computational tasks as well as its compatibility with all chosen Libraries. The Libraries used were Sci-kit Learn [7] for Agglomerative and BIRCH Clustering as well as Metrics and Pyclustering [6] for CURE Clustering. For visualization PPTK was used. Cloud Compare was used to prepare geometric features and for visualisation during testing. Additionally matplotlib was used for graphing, pandas for saving and accessing results data, and a Pointnet++ implementation in Python using Pytorch by Yan [13], which was customised to provide the feature extraction capabilities we needed from Pointnet++. Open3D Team [12] was used for downsampling.

A summary of the experimental platform used can be found at Table 4.

4.2 Data acquisition and preparation

4.2.1 Data and feature set overview. This project uses a single data set from The Visual Computing Group (VCG) at CNR-ISTI [1]. This data set of a cultural heritage site is a point cloud of ±22 million points as well as scanning intensity and ground truth (keep/discard) labels. This data set is used as the base data set for the project. Two additional data sets are created from the raw data set. These data sets contain additional feature sets as well as the original intensity and ground truth labels. A summary of the three data sets is described in Table 1. To create the feature sets described and to provide optimal input data to the clustering algorithms used in this project three data preparation steps are needed - feature extraction, downsampling, and ground truth removal for clustering.

Table 1: Summary: data set description

Raw features	x,y,z; intensity; ground truth labels
Geometric/covariance features	x,y,z; intensity; ground truth; 22 geometric/covariance features generated using CloudCompare
PointNet++ features	x,y,z; intensity; ground truth; 128 features generated using PointNet++ [9]

4.2.2 Feature extraction. Feature extraction was the first step in the data preparation pipeline. The process of feature extraction of Geometric/Covariance features and features from the Pointnet++ deep neural network are as follows.

CloudCompare an OpenSource 3D point cloud processing and visualisation software was used to extract 22 geometric and covariance features (screenshot: Figure 17) from the raw data set. Some of these features are described in the table Figure 18, which can be found in the appendix. The raw data set stored as a PLY file was loaded into the software. The intensity and ground truth labels were loaded as scalar values into the software. The "compute geometric features" tool was used to extract the 22 geometric features shown in the screenshot in figure Figure 17. These features were appended to the original data set and then exported in the LAS file format which proved the easiest to read into our testing suite.

We used the PointNet++ deep net to extract ±128 non human-understandable features. We have used and modified a PyTorch implementation [13] of PointNet++ and its pre-trained model to extract these features from our original data set. We have used this implementation instead of the original version written in C++ since our project was written in Python. PointNet++ requires our raw data set to be fed into the model as (xyz, intensity, intensity, intensity) as the model expects (xyz, rgb) values as input. We reshaped our data accordingly to conform with this. The data set is run through the pre-trained PointNet++ model until just before the classification step where we extracted the data set with the additional features the model has extracted. We saved the new data set in the NPY file format as the output is a Numpy array.

4.2.3 Downsampling. Downsampling of the point cloud data was needed as the original data set of ±22 million points was too large

to experiment within a realistic amount of time, using many of our algorithms or with the memory constraints of the machinery used. Downsampling was implemented in Python and uses the Open3D Library Zhou et al. [15] to implement voxel down sampling. A voxel size is used as the parameter for how much to downsample, with a value of 0.05 producing a point cloud with 297 098 points. This was much better for working with and while still retaining the structure and a fair amount of the fine information of the original.

4.2.4 Removal of ground truth labels. Before any of the clustering algorithms were run, our data set had to have its ground truth labels removed as these labels would have affected the clustering algorithms and would have consequently produced invalid clusters. The removal of the ground truth labels was conducted when the data sets were loaded into the test suite. Labels were removed from the Numpy array containing the point cloud data and a new clustering ready array was passed to the clustering algorithms.

4.2.5 Data ingestion. The data was read in using Numpy for .npy files, Open3D for .ply files, and Laspy for .las files. All data sets were read in and stored in memory in two forms: with truth labels and without truth labels. Geometric/covariance features from CloudCompare were saved as .las files, while Pointnet++ data sets were saved as .npy files. The structure of the data is described in Table 2.

Table 2: Summary: data set structure

Data set	Labels	ndarray shape (with truth)
Raw	points; intensity; (truth)	(n, 5)
Geometric / Covariance	points; intensity; (truth); 22 features	(n, 27)
Pointnet++	points; (truth); 128 features	(n, 132)

4.3 Data visualisation

The Point Processing Toolkit (PPTK) was used for data visualisation. Methods were written in a viewer class that split the point cloud data into its points and features for visualisation in PPTK. PPTK is a lightweight framework that allows one to view the point cloud data in 3D. This is useful in visually evaluating and understanding the qualities of the clusters, like separation.

4.4 Design constraints

This project was written in Python and utilised open library implementations of the clustering algorithms and metrics used. These libraries although open source, may still contain errata. These libraries also may or may not implement optimisations that improve or worsen the quality of results and real time performance of the algorithms. For example, the pyclustering algorithm does not implement CURE to use multi-threading and as such it runs on one core. Additionally, a modern CUDA capable graphics card with at least 4GB of RAM (not extensively tested) is required to run the feature extraction from Pointnet++.

4.5 Automation of experimentation

Testing was automated so that batches of runs for a selected combination of clustering algorithm, data set, and downsampling were run. Metrics where run on these clusters and the results automatically saved to a unique .csv file for the selected combination of criteria of: clustering algorithm, data set, and downsampling. These saved results were then used to pull the required results to plot and explore them below.

5 RESULTS

Based on the experiment described above, the following resultant metric scores were obtained and plotted against $100 <= k <= 750$ clusters using an automated testing approach.

5.1 Cluster Analysis/Cluster Metrics

The description and plots of the the performance of each algorithm against the Davies-Bouldin (DB) and Rand Index follow. A smaller DB index score represents better defined clusters. As a measure of accuracy, a higher Rand index score is desired. Overall the rand index is lower than one may expect.

5.1.1 K-Means (baseline). The DB Index on K-Means is lowest and therefore seen as better on the raw dataset. The Geometric and Pointnet++ data sets have an overall logarithmic shape which diverges with the Pointnet++ data set, showing worse clustering performance with additional clusters. Over approximately 200 clusters, the Geometric and K-Means data sets seem to be stable and show no significant improvement or worsening of their score.

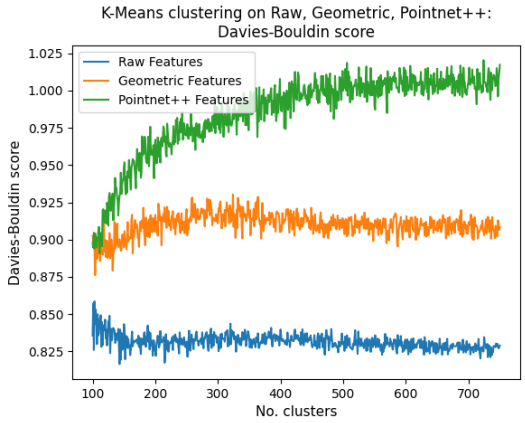

Figure 1: DB Index on K-Means Algorithm

The rand index on K-means (Figure 19) shows an overall similar shape across all 3 datasets. Pointnet++ performs best owing to its relatively high scores. However the scores are low overall, with all showing minimal change over approximately 200 clusters. The shape of the plots shows a slight exponential decay with the initial drop off likely happening at $k < 100$.

5.1.2 CURE. The Davies-Bouldin Index for the CURE clustering algorithm (Figure 2) shows similar worst results at about the 500 cluster mark. All 3 data sets present an overall logarithmic shape,

with Pointnet++ including an anomalous local maximum between approximately 375 and 500 clusters. This may be an error in execution of the experiments in that range, or may be due to the effect of hyper-parameter choices for this algorithm. Due to this, the clusters could split or merge in this fashion for this series of k clusters.

There is no definitive data set that achieves a clear minimum DB index score. As a final note, we see instability in the range of about 100 to 300 clusters, after which there is minimal difference.

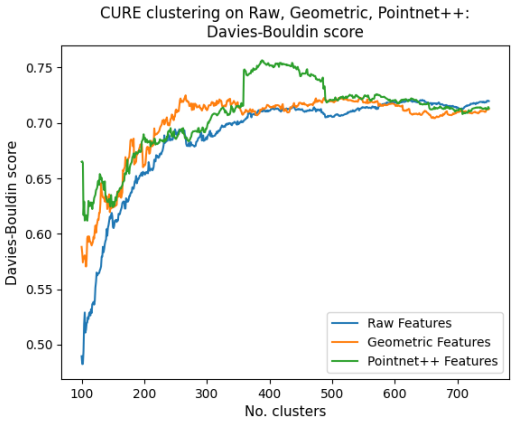

Figure 2: DB Index on CURE Algorithm

The rand index (Figure 22) shows the similarty between the clusters and their fit to ground truth labels across datasets. The overall poor score may be due to the fit of clusters over the ground truth labels being poor.

The rand index shows similar results for Geometric and Pointnet++ features. With some of the highest scores in CURE clustering and some of the highest scores over all algorithms. All scores stabilise at about 300 clusters. With no improvement with increased number of clusters. The shape of the plots show an exponential decay.

5.1.3 BIRCH. The BIRCH algorithm seems to suggest the best clustering on the raw data set as the DB index on the raw data set is lowest. Pointnet++ features rapidly diverge from Geometric features with a worse score overall. However both Geometric and Pointnet++ DB scores trend down with added clusters. Both Geometric and Pointnet++ data sets reach a local maximum (worst score) at approximately 150 and 200 clusters respectively. We see DB index decays faster in the Geometric data set than it does in the Pointnet++ data set. The results of the raw data set are starkly different to the Geometric and Pointnet++ data sets, showing small fluctuations around a score of approximately 0.87.

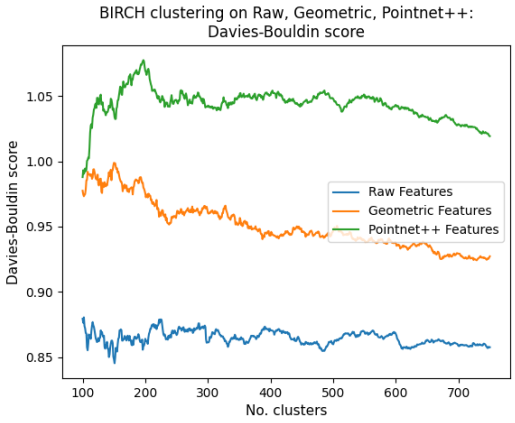

Figure 3: DB Index on BIRCH Algorithm

In terms of the Rand score (Figure 20) the 3 data sets produce the same overall shape of a slight exponential decay, with the initial drop off likely happening at $k < 100$ clusters. Pointnet++ outperforms the Geometric and raw data sets, with a score stabilising at around 0.24. The scores show minimal change as the number of clusters increases, and are low overall.

5.1.4 Agglomerative Clustering. The DB Index on the Agglomerative algorithm shows the best performance on the raw data set which has the lowest overall scores. The DB index trends downwards for the Geometric and raw datasets after approximately 200 clusters. This is in stark contrast to Pointnet++ stabilises but fluctuates constantly around a score of 1.03. All 3 data sets show an initial peak at around $150 < k < 200$.

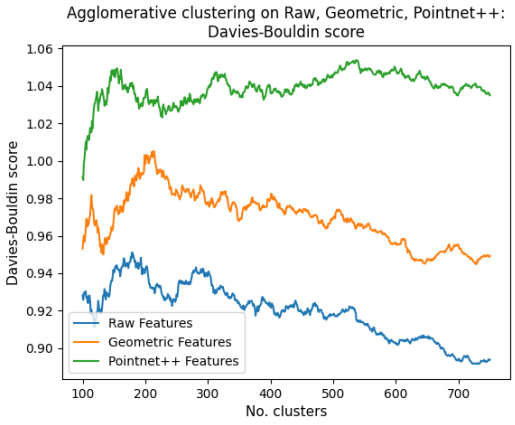

Figure 4: DB Index on Agglomerative Algorithm

The Rand scores (Figure 21) of Geometric and Pointnet++ datasets are almost identical on the Agglomerative algorithm. However all 3 datasets show an exponential decay with the Geometric data set decaying fastest as the number of clusters increases. Pointnet++ and raw data sets show the overall highest Rand scores indicating the best overlap between ground truth and the produced clusters.

5.2 Classification Metrics

A summary of the following scores: F1, IoU, Precision and Recall, for the classified clustered data are provided below. The F1 and IoU Metrics give an indication of how well the classified data fits over the ground truth labels. A higher score for both these metrics generally indicates well fitted data. The precision metric shows how precise the positive labels of the clustered data are. False positives bring down this score. While the recall metric provides insight on the prevalence of false negatives which are preferably minimised. A higher recall precision score indicates more accurate classification. We will also touch on absolute and mean squared error scores which can be found in the appendix. Error scores preferably are low which indicates low error rates.

5.2.1 K-Means (Baseline). All three data sets show an overall logarithmic shape on Figure 5, with Geometric and Pointnet++ data sets having an overall higher F1 score. The Pointnet++ data set begins to diverge above the Geometric dataset at about 400 clusters and eventually outperforms the other 2 algorithms at approximately 600 clusters with an F1 score of 0.8.

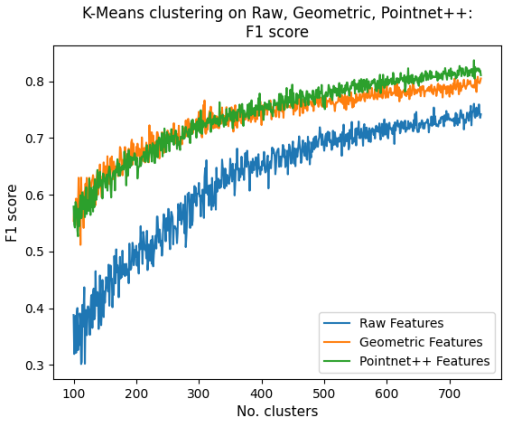

Figure 5: F1 Score on K-Means Algorithm

Similarly to the F1 score, all three data sets show an overall logarithmic shape on Figure 23 (IoU). Again we see Pointnet++ and Geometric data sets having similar shapes, with Pointnet++ diverging at approximately 400 clusters. Pointnet++ clearly outperforms the other tow algorithms at around 600 clusters, with an IoU score of 0.65.

On both Figure 5 (F1) and Figure 23 (IoU) we notice that the raw data set has a noticeably lower score overall. The F1 and IoU Scores are best on the larger feature sets. There is an overall upward trend of both scores as the number of clusters increases.

The Precision score shown in Figure 6 is very noisy when compared to the same metric on the hierarchical algorithms. The overall shape across all 3 data sets follows an S-shape with a point of inflection at around 250 clusters for Geometric and Pointnet++ and 350 clusters for raw. The precision starts to stabilise at around 400 clusters for Geometric and Pointnet++ and approximately 500 clusters for the raw data set. From approximately 200 clusters we see that an increase in the number of clusters causes an increase in precision up until approximately when the precision scores begin to stabilise. All 3 datasets show very promising and high levels of precision at clusters over approximately 500.

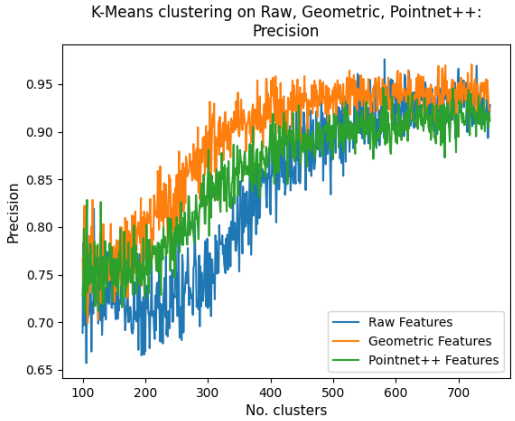

Figure 6: Precision on K-Means Algorithm

The recall scores as seen in Figure 7 shows a logarithmic increase over an increasing number of clusters across all 3 data sets. The Pointnet++ and Geometric datasets outperform the raw data set overall. The two larger feature sets closely follow the same trend up until approximately 500 clusters, at which point Pointnet++ outperforms all 3 datasets. The recall scores are high overall.

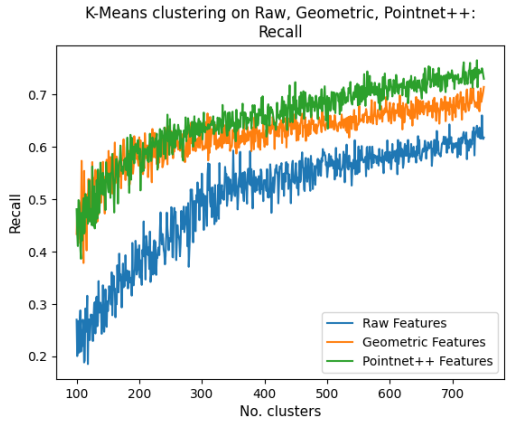

Figure 7: Recall on K-Means Algorithm

Mean absolute and squared error as seen in Figure 27 and Figure 28 are very similar on K-means, with results being low across the board. For both metrics we see that raw outperforms the other 2 data sets, with raw having the lowest error of approximately 0.03.

5.2.2 BIRCH. The F1 score (Figure 8) and IoU score (Figure 24) on Geometric and Pointnet++ feature sets produce very similar scores with both feature sets out performing the raw dataset on the respective metrics. Across both metrics all 3 data sets have an overall logarithmic shape with a step-like structure. This is

expected of hierarchical algorithms. The Geometric and Pointnet++ data sets show an overall increase in both metrics as the number of clusters increase. The raw data set has periods of stability without improvement of the score at the intervals of approximately 350 to 500 clusters and 650 clusters to 750 clusters on both metrics.

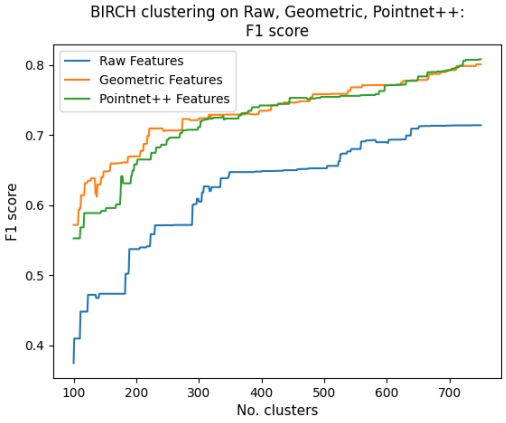

Figure 8: F1 Score on BIRCH Algorithm

The precision metric as seen in Figure 9 shows instability across all data sets for especially in the range of 100 to 350 clusters. The Pointnet++ and raw data sets have a overall increasing linear trend. The same cannot be said for the Geometric feature set. Between 300 and 600 clusters we see a local maximum with a peak precision score sitting at approximately 0.95 for the Geometric data set. The overall precision across all 3 data sets is relatively very high (above 0.80 for $k > 300$).

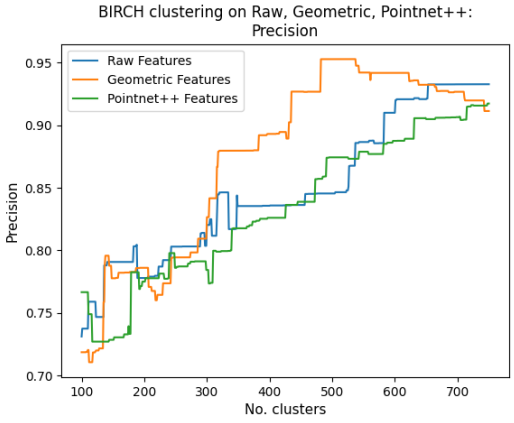

Figure 9: Precision on BIRCH Algorithm

Recall on BIRCH has a sharp increase on all feature sets for $k < 350$ with a flattening out the overall logarithmic curve thereafter. The geometric and Pointnet++ data sets begin to steadily increase again after a period of stability at around $k > 550$. Overall the recall metric is good with relatively high scores. The Pointnet++ and Geometric data sets follow closely.

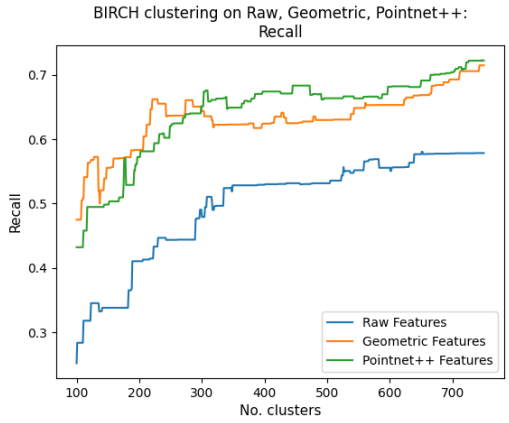

Figure 10: Recall on BIRCH Algorithm

The mean absolute error (Figure 29) and mean squared error (Figure 30) are similar on all feature sets with Pointnet++ haveing the highest overall error on both metrics. However these scores are still relatively low overall. Both errors decrease with increasing number of clusters.

5.2.3 Agglomerative. The F1 (Figure 11) and IoU (Figure 25) scores are very similar across all datasets. Both metrics have a sharp increase in score from $100 < k < 300$. All data sets then have a slower increase in score with increasing clusters in both cases. The rate of change for the raw and Pointnet++ datasets is very similar, this is in contrast to the Geometric data set which seems to be quite linear from approximately 300 clusters. All three data sets on both metrics increase to around a score 0.8 on F1 and 0.7 on IoU at 750 clusters.

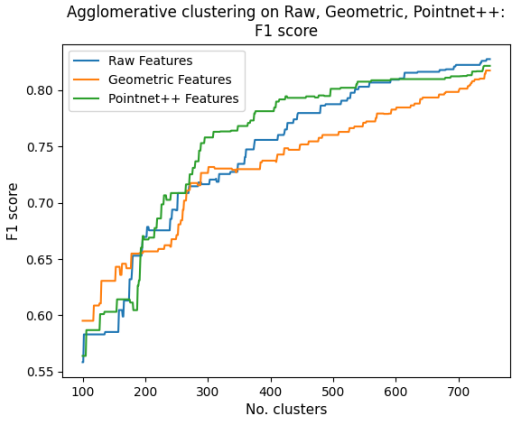

Figure 11: F1 Score on Agglomerative Algorithm

The Precision score (Figure 12) increases rapidly on the Pointnet++ and Geometric data sets from approximately $175 < k < 250$ after a drop in the preceding range. While raw and Pointnet++ seem to stabilise at 300 and 450 clusters respectively around a precision of approximately 0.9, the Geometric data set continues to

increase before reaching a local maximum of approximately 0.95 at 550 clusters before stabilising and then dropping at approximately 650 clusters. All 3 data sets achieve high precision scores across the board.

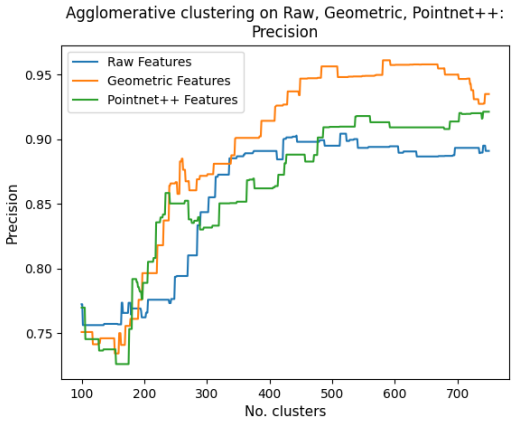

Figure 12: Precision on Agglomerative Algorithm

The Recall score (Figure 13) increases sharply from $100 < k < 300$ clusters for the Geometric and raw data sets and continues increasing sharply for the Pointnet++ dataset increasing to a noticeably higher score then the other two at approximately 300 clusters. The Pointnet++ data set seems to stabilise at a score of approximately 0.7 from the 300 cluster mark. the raw data set increases linearly to a maximum score of around 0.75 at 750 clusters and the Geometric score has the lowest score the range after 300 clusters with a gradual increase to 750 clusters.

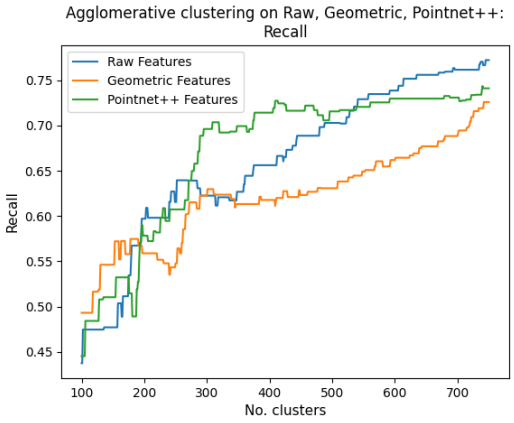

Figure 13: Recall on Agglomerative Algorithm

Absolute and squared mean error is almost identical as seen in Figure 31, and Figure 32 with scores being low overall and decreasing as clusters increase. There is an exponential drop off with scores for all data sets, converging at a minimum of approximately 0.05 at 750 clusters.

5.2.4 CURE. The highest overall F1 (Figure 14) and IoU (Figure 26) score is achieved on the Geometric dataset. The second highest was found to be raw, and third highest being Pointnet++. All three data sets gradually increase in both metrics as the number of clusters increases. Pointnet++ has limited instability which should be noted. Overall on F1 all data sets achieve quite high results (> 0.7) over 350 clusters, with the geometric data set reaching this score at approximately 200 clusters. Similarly on IoU high scores are achieved overall.

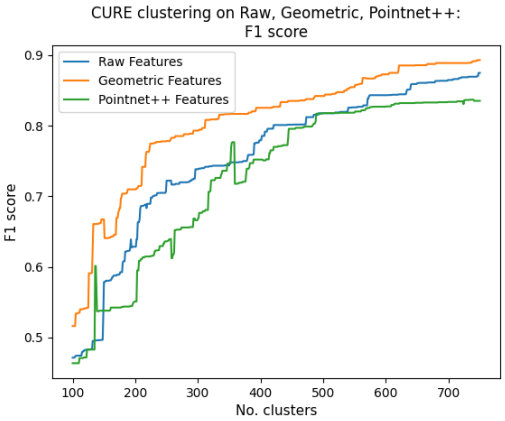

Figure 14: F1 Score on CURE Algorithm

Precision (Figure 15) on the CURE algorithm is highly unstable on the Pointnet++ data set, with multiple peaks and troughs throughout the range of clusters. However there is instability across data sets in the first 300 clusters. The Geometric and raw feature sets show a more stable increase with increasing clusters. Despite this erratic behaviour the precision scores for the raw and Geometric data sets are relatively high. The raw feature set performs best in this metric.

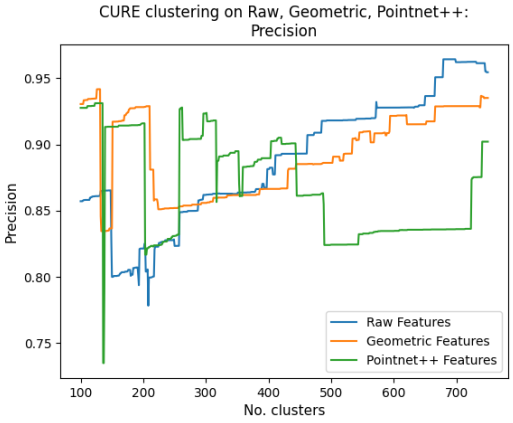

Figure 15: Precision on CURE Algorithm

Recall on CURE (Figure 16) is a lot more stable. On the raw and Geometric data sets the metric gradually increases from approximately 200 clusters. The Pointnet++ data set also increases with

additional clusters on the whole, but is much more erratic than the other two. The Geometric feature set performs best overall on this metric.

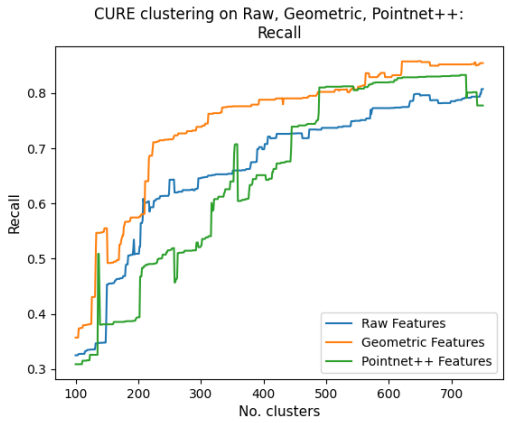

Figure 16: Recall on CURE Algorithm

Mean absolute error (Figure 33) and mean squared error (Figure 34) decrease consistently with additional clusters. Geometric and raw feature sets perform the best in both metrics.

6 DISCUSSION

From the results seen above, the performance of unsupervised hierarchical clustering algorithms in producing semantically meaningful clusters on point clouds is analysed. The results also indicate whether unsupervised hierarchical clustering methods can be used as a step in point cloud cleaning. Both clustering and classification metrics are analysed for both hierarchical and K-means algorithms and are discussed below.

6.1 Unsupervised clustering performance

The algorithm with the lowest and most consistent DB index score across all data sets was the CURE clustering algorithm (Figure 2), with a worst score of around 0.75 and an average score on the interval for which it stabilises ($300 < k < 750$) of approximately 0.7. This is promising as a lower DB score means that clusters are more unique/separate from other clusters. This means CURE may produce semantically meaningful clusters from an unsupervised lens.

Additionally, the fact that the score is so consistent across data sets means that there may be little need to feed the algorithm higher dimension feature sets. Another interesting result is that the raw dataset produced the lowest DB score across all algorithms including K-means. These two observations are beneficial to the performance of the pipeline, as there are no preprocessing steps required, and one would benefit from faster run-times.

All hierarchical algorithms have less noisy DB graphs due to what is assumed to be a result of the tree structure behind these algorithms, which produces “step-like” graphs.

BIRCH (Figure 3) and Agglomerative (Figure 4) clustering performed very similarly to each other in this metric. There is a clear split between the performances of the respective data sets on these algorithms. This is a similar result to DB score of K-means (Figure 1. Data sets are, from best to worst performing: raw, geometric, Pointnet++. This reason for this similarity is not certain, but could be due to the precomputing of a k-nearest neighbours graph on the Agglomerative algorithm and the efficient cf-tree characteristics of BIRCH that makes its performance similar to the k-family of clustering algorithms. This similarity could mean that the clusters have the improved separation of a hierarchical clustering method without the influence of spatial outliers such as on K-means.

While K-means clearly has its DB scores stabilise on each data set at a particular point, the BIRCH algorithm sees a gradual decrease in the scores on each data set with an increase in the number of clusters and, the Agglomerative algorithm sees the same trend on the Geometric and raw data sets. It is not clear why there is not the same trend on the Pointnet++ data set for the Agglomerative algorithm. The higher dimensionality could be slowing this trends appearance, but this is unclear.

Regardless, the decreasing trend hints that better clustering can be found at greater k values, but as mentioned in the experiment design, setting the number of clusters to values exceeding 750 starts producing clusters that are unrecognizable as individual features, which, by definition, is not a semantically meaningful cluster.

When looking at the rand scores across clustering algorithms the results are very poor, on K-means and all the hierarchical clustering algorithms. Technically, CURE has far and away the best rand score with a similar exponential decay to agglomerative clustering (Figure 22). A similar graph would be expected on BIRCH (Figure 20) but this is not the case. BIRCH has a graph similar to K-means. This result is neither here nor there.

In fact, in analysing the results of the rand score across algorithms the scores seem remarkably poor, so much so that there is a chance we have implemented the metric incorrectly and these are erroneous results. We feel there is no need to dig deeper into this metric as the results are not helpful in finding a number of clusters that could produce semantically meaningful clusters.

In terms of the ideal number of clusters based purely on cluster analysis, it seems the best DB scores across hierarchical algorithms lie between 300 and 750 clusters with the best DB scores on the raw data set followed by the geometric data set. On cure there is little change in the DB index between 300 and 750 clusters, while on BIRCH and Agglomerative clustering there are some increases in this range with a general trend of the score decreasing with increasing k. When looking at the results of K-means, we see that it is noisy but stable over the range 300 -750 clusters on geometric and raw feature sets. It also has a better DB index than birch or Agglomerative in this range. From these results, CURE followed by K—means seem to be the most promising algorithms with an ideal number of clusters between 300 and 750 clusters.

6.2 Binary classification

The discussion of the binary classification metrics is split up into its similar metrics.

6.2.1 F1 Score and IoU Score. The best F1 and IoU scores on the K-Means (Figure 5 and Figure 23) and BIRCH (Figure 8 and Figure 24) algorithms are found on the Pointnet++ and Geometric data sets.

These data sets meaningfully outperform (0.1 higher score) the the Raw data set on these algorithms. Whereas on the Agglomerative (Figure 11 and Figure 25) and CURE (Figure 14 and Figure 26) algorithms the best scores are found on the Geometric and Raw data sets. However, these data sets only marginally outperform the Pointnet++ data set. The curves across all three data sets are much tighter on the Agglomerative and CURE algorithms, with the biggest variance between data sets seen on the CURE algorithm which still has a variance between data sets comparatively smaller then that seen on K-Means and BIRCH. The F1 and IoU scores are high across all algorithms and data sets but the best overall results across all data sets are seen on the Agglomerative and CURE algorithms. The highest overall F1 and IoU scores (0.9 and 0.8) are achieved on the CURE algorithm using the Geometric dataset. The performance in this score across data sets using the Agglomerative and CURE algorithms is very promising. If the extra dimensionality of the Geometric and Pointnet++ data sets only marginally improves or even reduces these scores, then it could be argued to disregard these higher dimensional datasets, and just use the Raw data set on these algorithms. This will improve the speed of both Agglomerative and CURE clustering, while not sacrificing binary classification performance.

There is a general trend of a fast increase in these scores on the range of about $100 < k < 300$, followed by slow growth and flattening out of these scores in the approximate range $300 < k < 750$ on all algorithms. The improvement in the scores as the number of clusters over 300 increases towards 750 clusters, with a change of around 0.1 seen in this range on all algorithms. This change is significant but not so massive that if one wanted to cluster on 500 clusters instead of 300 clusters the improvement in accuracy of the binary classification would not be more then about 0.05 or 5%. This is good to know as it can be used to minimise the number of clusters one would like to use if they have a particular level of accuracy in mind.

In terms of unusual or outlier trends not mentioned, it should be noted that K-Means is much noisier than the hierarchical algorithms. This variance from k clusters to k clusters +1 makes it difficult to understand exactly what these scores are at a particular k. Secondly Agglomerative clustering on the Geometric feature set sees a plateau in the range of around 300 to 400 clusters and then sees a sharp linear growth from around 400 clusters as clusters increase to 750. Additionally it should be noted CURE has intermittent periods of instability in its F1 and IoU scores, this behaviour is mostly gone by approximately 350 clusters. This behaviour is seen in the other hierarchical algorithms but to a degree (spikes and drops are very small) that there is no need to consider it.

6.2.2 Precision Score and Recall Score. All data sets perform well (> 0.65) on the precision score and have a trend that the precision score across data sets tends to converge as the number of clusters increases towards 750. The scores across the algorithms and data sets are quite unique.

K-means (Figure 6) has an S-shape precision score with very high stable scores over about 350 clusters on all data set, but the score is very noisy with a lot of variance in the score with the jump between clusters of around 0.05 to 0.1 a significant variance.

The precision score on BIRCH (Figure 9) is very unstable for $k < 350$. Overall the precision score improves slowly as the number of clusters increases. The score also shows greater stability on the Raw and Pointnet++ data sets, especially from approximately 350 clusters. Notably there are a big spikes in the score on the Geometric feature set at approximately 300, 425, and 480 clusters after which precision drops slowly form about 500 clusters. This local maximum could mean this is a be the sweet spot for birch on the Geometric data set with a very high score at around 0.95.

Agglomerative precision (Figure 12) looks similar to K-means and performs similarly if not slightly worse. The score increases rapidly to about 250 clusters then again from about 300 on Geometric and Pointnet++. Raw grows more slowly overall and stabilises earlier at around 380 clusters. Pointnet++ seems to stabilise at around 400 clusters. While on the Geometric data set the score reaches a local maximum at around 570 clusters and has very high precision exceeding 0.95.

CURE precision (Figure 15) is extremely unstable and produces an almost unreadable plot for Pointnet++ at around 280 and 300 clusters the are high scores on Pointnet++ similarly there is a period of high precision on Pointnet++ and geometric features from around 150 to 200 clusters. Pointnet++ decreases on the whole to around 480 clusters, and stabilises at a score of around 0.825 until about 700 clusters then rapidly increases to a score of 0.9. The instability may come from the choices of representors and compression. CURE is very sensitive to these hyper parameters. Raw and Geometric features perform best with scores above 0.95 seen on raw and scores over 0.9 on geometric. On these data sets scores slowly increase from around 250 to 750 with a very high local maximum of (> 0.95) on raw at around 680 features.

Overall the precision scores for K-Means, BIRCH and Agglomerative are very impressive with the scores showing clear trends that could be utilised to maximise this score. These scores also show positive results for the Raw data set on BIRCH, Agglomerative and K-means which could reduce the complexity of clustering on these algorithms. CURE has some outstanding results but its general instability does not lend itself to any strong predictive trends especially on the Pointnet++ data set, maybe with different hyper parameters this would not be the case. The best performing combinations of algorithm and data set minimise false positives.

In terms of recall, on K-means Figure 7 recall is similar shape to its F1 and IoU score increasing at a higher rate to about 350 clusters thereafter growing slowly but mostly stable slow growth. Variance from noise is lower then precision. It is clear that Pointnet++ and the geometric feature produce the best results, but there are good and consistent results overall.

The BIRCH (Figure 10) recall scores are similar to K-means but this algorithm seems to perform better with stability and slow growth seen at $k > 350$. The recall score on BIRCH shows best scores on the Pointnet++ and Geometric data sets. Raw seems to have its recall score stabilise at around 325 clusters then increase again at about 500 clusters. Pointnet++ and geometric seem to increase their rate of improvement again from around 500 clusters.

Agglomerative recall Figure 13 has lots of instability leading up to around 300 clusters. There is a local maximum on Pointnet++ at around 400 clusters, but overall Raw achieves the best result at about 500 clusters. Pointnet ++ stabilises around 280 clusters.

Geometric has a comparatively low local maximum at around 300 clusters then dips followed by a steady rise to around a score of 0.70 at 750 clusters.

On CURE Figure 16 the recall score is unstable on Pointnet++ but there is a general trend on all the algorithms of improvement in this score as the number of clusters increases.

The CURE and Agglomerative algorithms achieve the best scores overall in terms of the recall score, with the raw metric on both algorithms performing comparatively high. Notably the Geometric data set on CURE and the Pointnet++ data set on Agglomerative perform comparatively high at lower numbers of clusters such as 250. While BIRCH and K-means provide scores that are predictable and relatively high overall on the Geometric and Pointnet++ datasets.

The performance of these algorithms in the recall metric on these particular data sets displays that certain algorithm and data set combinations minimise false negatives.

6.2.3 Mean Absolute and Squared Error Scores. Mean and absolute error results were almost always very similar to each other across the various algorithms. Results were also positive in that they were low on these metrics across the board.

On Agglomerative clustering (Figure 31, Figure 32) as the number of clusters approach 750 the mean and absolute error scores on all data sets appear to converge. On K-means (Figure 27, Figure 28) the Geometric and Pointnet++ feature sets seem to converge as k approaches 750 with Raw having the best result overall. On BIRCH (Figure 29, Figure 30) and CURE (Figure 33, Figure 34) the Geometric and raw feature sets appear to converge as k approaches 750 with the worst result on the Pointnet++ dataset. BIRCH has the lowest scores overall followed closely by Agglomerative and K-Means.

Agglomerative has a sharp downward trend as k increases to about 300 clusters where its scores fall below 0.07 on all data sets. This indicates that Agglomerative produce the lowest mean error scores over 300 clusters on all data sets, while BIRCH produces the lowest mean error scores on lower numbers of clusters $k < 300$ for the Geometric and Raw data sets but also has respectable results for the Pointnet++ data set. However the Agglomerative algorithms results are promising as a catch all algorithm with the caveat of performing best only over a certain threshold.

7 CONCLUSIONS

The results of this project produce promising results, with hierarchical clustering algorithms producing semantically meaningful clusters.

7.1 Overall hierarchical algorithm performance and applicability

The results of the metrics discussed underline some of the positive characteristics and some of the weaknesses of the algorithms used for the experiments conducted in this project. From the results discussed it seems that not only are hierarchical clustering algorithms able to perform comparatively well to K-means on every single metric examined but on many of the metrics some of the hierarchical algorithms outperform K-means.

Overall the clustering algorithms presented including K-means can produce clusters that are semantically meaningful. This is supported by the high results in unsupervised clustering metrics like the DB Index and classification metrics like the F1, IoU, and Precision and Recall scores.

The results of the unsupervised hierarchical clustering algorithms show that these algorithms provide good intra-cluster similarity while providing good inter-cluster separation. CURE stood out as a useful hierarchical clustering algorithm with the lowest overall scores and good results on the DB index. It is also the most similar across data sets. Overall the data set with the lowest DB index offering the best performance on all algorithms except CURE is the raw data set. This is a very important result because if one can reduce the dimensionality of the input data while still providing strong performance in these metrics it can reduce the need for extra dimensions and improve the performance of the clustering algorithms without losing performance.

Binary classification is a proxy for semantically meaningful clusters, as the quality of the binary classification represents the similarity of the clusters to the ground truth data. Hierarchical clustering algorithms performed well here too with positive results. What is promising is the performance of the raw and Geometric data sets in these results with the results not being dominated by very high dimension data sets like Pointnet++. Algorithms like agglomerative clustering stood out as being able to score positive results on the raw data and Geometric data set. Overall the hierarchical algorithms performed well and with some outperforming K-means in some metrics.

The trends seen in the results of the hierarchical algorithms show generally that increasing the number of clusters k results in a positive improvement in the classification metrics. With the intention to produce clusters that are meaningful and not just small blobs of points it seems the $k < 750$ cut-off worked well to show the best practical performance of these algorithms. In general the range $300 < k < 550$ is where one sees the best performance while maintaining semantically meaningful clusters. A point cloud visualising 500 clusters using the Agglomerative and K-Means clustering algorithm on the Pointnet++ data set can be found in the appendix (subsection F.1 and subsection F.2). The hierarchical algorithms lend themselves nicely to optimisation and have clear ranges and local maxima where performance could be maximised.

7.2 Adherence to aims

Hierarchical clustering algorithms can perform well and match if not best k-means in producing semantically meaningful clusters. This is displayed in the results and discussion. They can produce clusters with performant inter-cluster separation and intra-cluster similarity as shown by their performance in the DB index. Hierarchical algorithms also perform very well in binary classification metrics where binary classification performance is a representation of semantic meaning of the clusters.

Hierarchical algorithms do come at a heavy computational cost in terms of time complexity and memory overhead. They seem to be sensitive to hyper parameters that in an application as say a intermediate step in the creation of a model for point cloud cleaning would mean that they would need these parameters tweaked for every new data set. Hierarchical algorithms have proven capable of performing well on lower dimension data sets and so the performance drawback may not be of concern. There are also heuristics

that can be used like the creation of k nearest neighbour-graphs or other preliminary steps that can reduce the complexity of the data fed into the algorithms.

Overall hierarchical clustering algorithms produced clusters that were able to identify and cluster on the fine details and noise that are required to extract the discard label points in this project's data set. They seem to do this with clear ranges of stability and this may give one the ability to tune in the hyper parameters required to get the most out of these algorithms in the real world.

A scheme of clustering on downsampled data, finding a sweet spot, tuning hyper parameters automatically or manually and then running hierarchical clustering algorithms once could be a strong consideration for a real world application of these algorithms.

7.3 Constraints and future work

As a brief summary some constraints of this project were: The knowledge base of the research team going into this project as this was our fist time working on any machine learning project; the fact that there was a single raw data set; and the inability to train the Pointnet++ model for the particular data set at hand. Due to the lack of speed of computation, the time allotted to complete all research caused a serious constraint in ensuring rigorous testing. A final large constraint was that all hierarchical algorithms and K-means required downsampling on this projects data set however CURE required a lot more downsampling due to its $O(n^2 logn)$ time complexity. The large downsampling causes too sparse a point cloud, which loses semantic meaning.

Future work on this project could include: Additional experiments on tuning hyper parameters; training the Pointnet++ model on the data set used instead of using its Stanford pretrained model; and rerunning the experiments conducted in this project. Experimentation on more data sets would be ideal in ensuring a thorough investigation is done. One could also consider the investigation of ROCK clustering as an improvement on CURE, and the investigation of other pre-computed graphs for Agglomerative clustering as a future investigation.

7.4 Quality and usefulness of results

To the best of our knowledge the results presented and discussed were accurate, bar the Rand score results which are likely incorrectly implemented. The results come with discussed constraints such as the requirement to downsample the data sets. The hierarchical algorithms do match or exceed K-means on certain metrics and come with their own beneficial characteristics such as their tunability, clear cutoffs and maxima in many metrics that allow for them to be used to their full potential in some cases. This, however, does not mean that the results of these experiments wholly conclude that hierarchical clustering methods are the best solution for this problem, with overheads previously discussed.

7.5 Summary of conclusion

Hierarchical clustering methods (namely: BIRCH, CURE, and Agglomerative) can match or surpass the performance of K-means clustering in some metrics. These methods can and have been shown to produce semantically meaningful clusters on 3D point cloud data of cultural heritage sites. These clusters can also perform well in binary classification when classified according to their overlap, with points labelled as keep or discard in a set of ground truth labels. They show promise as theoretically being used as an intermediate step in speeding up point cloud cleaning by clustering on semantically meaningful structures that could form parts of keep or discard labels.

ACKNOWLEDGMENTS

Thank you to my supervisor, Assoc. Professor Dr. Patrick Marais, and co-supervisor, Luc Hayward for their continuous support and guidance. And to Leah Gluckman and Jemma Sundelson for their parallel work on the preliminary part of this project.

REFERENCES

[1] Paolo Brivio. 2022. *Visual Computing Lab.* Visual Computing Lab. Retrieved May 22,2022 from http://vcg.isti.cnr.it/

[2] David L. Davies and Donald W. Bouldin. 1979. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1, 2 (1979), 224–227. https://doi.org/10.1109/TPAMI.1979.4766909

[3] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. 1998. CURE: An Efficient Clustering Algorithm for Large Databases. *SIGMOD Rec.* 27, 2 (jun 1998), 73–84. https://doi.org/10.1145/276305.276312

[4] Patrick Marais, Matteo Dellepiane, Paolo Cignoni, and Roberto Scopigno. 2019. Semi-Automated Cleaning of Laser Scanning Campaigns with Machine Learning. *ACM J. Comput. Cult. Herit.* 12, 3, Article 16 (June 2019), 29 pages. https://doi.org/10.1145/3292027

[5] Patrick Marais, Matteo Dellepiane, Paolo Cignoni, and Roberto Scopigno. 2019. Semi-automated Cleaning of Laser Scanning Campaigns with Machine Learning. *ACM Journal on Computing and Cultural Heritage* 12, 3 (2019), 1–29. https://doi.org/10.1145/3292027

[6] Andrei Novikov. 2019. PyClustering: Data Mining Library. *Journal of Open Source Software* 4, 36 (apr 2019), 1230. https://doi.org/10.21105/joss.01230

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[8] Christopher Pocock. 2019. *3D Scan Campaign Classification with Representative Training Scan Selection.* Master's thesis. Faculty of Science, Department of Computer Science.

[9] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems* 30 (2017), 1–10.

[10] William M. Rand. 1971. Objective Criteria for the Evaluation of Clustering Methods. *J. Amer. Statist. Assoc.* 66, 336 (1971), 846–850. https://doi.org/10.1080/01621459.1971.10482356 arXiv:https://www.tandfonline.com/doi/pdf/10.1080/01621459.1971.10482356

[11] Heinz Rüther, Christoph Held, Roshan Bhurtha, Ralph Schroeder, and Stephen Wessels. 2012. From Point Cloud to Textured Model, the Zamani Laser Scanning Pipeline in Heritage Documentation. *South African Journal of Geomatics* 1, 1 (2012), 44–59.

[12] Editorial Team. 2018–2021. *Point Cloud.* Open3D. http://www.open3d.org/docs/release/tutorial/geometry/pointcloud.html

[13] Xu Yan. 2019. Pointnet/Pointnet++ Pytorch. *https://github.com/yanx27/Pointnet*$_p ointnet2_p ytorch$(2019).

[14] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1996. BIRCH: An Efficient Data Clustering Method for Very Large Databases. *SIGMOD Rec.* 25, 2 (jun 1996), 103–114. https://doi.org/10.1145/235968.233324

[15] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. 2018. Open3D: A Modern Library for 3D Data Processing. *CoRR* abs/1801.09847 (2018), 1–6. arXiv:1801.09847 http://arxiv.org/abs/1801.09847

A EXPERIMENTS

A.1 Considerations

Table 3: Downsampling across algorithms and datasets

	Raw	Geometric	Pointnet++
CURE	0.195	0.300	0.350
BIRCH	0.050	0.075	0.175
Agglomerative	0.205	0.215	0.205
K-Means	0.050	0.150	0.205

B SYSTEM DESIGN AND IMPLEMENTATION

B.1 CloudCompare Feature Extraction

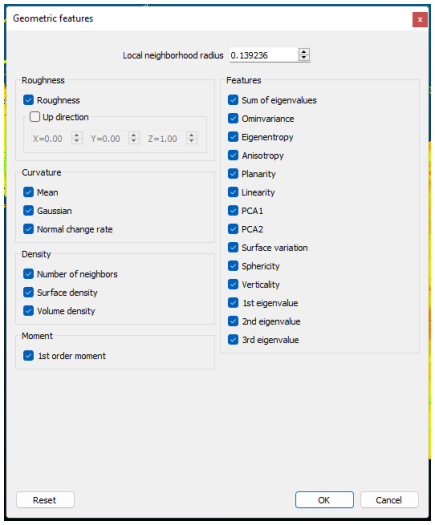

Figure 17: CloudCompare Feature Extraction

B.2 Platform Details

Table 4: Platform summary

Programming language	Python 3.6.15
Core Libraries	Sci-kit Learn [7] Pyclustering [6] PPTK Open3D [12] Pointnet/Pointnet++ Pytorch [13]
Software tools	CloudCompare
Version Control	Git using Github.com
System	Intel Core i7 6700k Processor Nvidia GTX 1060 6GB Graphics Card 16GB 3200MHz DDR4 RAM

Name	Symbol	Definition
Covariance/shape features		
Verticality	V	$1 - \lvert\langle [001], e_3 \rangle\rvert$
Linearity	L_λ	$(\lambda_1 - \lambda_2)/\lambda_1$
Planarity	P_λ	$(\lambda_2 - \lambda_3)/\lambda_1$
Curvature	C_λ	$\lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$
Sphericity	S_λ	λ_3/λ_1
Omnivariance	O_λ	$\sqrt[3]{\lambda_1 \cdot \lambda_2 \cdot \lambda_3}$
Anisotropy	A_λ	$(\lambda_1 - \lambda_3)/\lambda_1$
Eigenentropy	E_λ	$-\sum_{i=1}^{3} \lambda_i \cdot \ln(\lambda_i)$
1st Order Moment 1	M_1	$\sum_{i \in \mathrm{P}} \langle \mathrm{P}_i - \mathrm{p}, e_1 \rangle$
1st Order Moment 2	M_2	$\sum_{i \in \mathrm{P}} \langle \mathrm{P}_i - \mathrm{p}, e_2 \rangle$
2nd Order Moment 1	M_3	$\sum_{i \in \mathrm{P}} \langle \mathrm{P}_i - \mathrm{p}, e_1 \rangle^2$
2nd Order Moment 2	M_4	$\sum_{i \in \mathrm{P}} \langle \mathrm{P}_i - \mathrm{p}, e_2 \rangle^2$
Sum of EVs	$\Sigma_{\lambda 3\mathrm{D}}$	$\lambda_1 + \lambda_2 + \lambda_3$
Sum of EVs (2D)	$\Sigma_{\lambda 2\mathrm{D}}$	$\lambda_{2\mathrm{D}_1} + \lambda_{2\mathrm{D}_2}$
Ratio of EVs (2D)	$\mathrm{R}_{\lambda 2\mathrm{D}}$	$\lambda_{2\mathrm{D}_2}/\lambda_{2\mathrm{D}_1}$
Geometric features		
Radius	r_{3D}	$\mathrm{dist}(\mathrm{p}, \mathrm{P}_k)$
Density	D_{3D}	$(k+1)/(\frac{4}{3}\pi {r_{3D}}^3)$
Radius (2D)	r_{2D}	$\mathrm{dist}(\mathrm{p}_{2\mathrm{D}}, \mathrm{P}_{2\mathrm{D}k})$
Density (2D)	$\mathrm{D}_{2\mathrm{D}}$	$(k+1)/(\pi {r_{2D}}^2)$
Height features		
Height difference	$\Delta\mathrm{H}$	$z_{\max} - z_{\min}$
Height std. deviation	$\sigma\mathrm{H}$	$\sqrt{\sum_{i=1}^{k} (z_i - \overline{z})^2/(k-1)}$
Vertical range (cylinder)	$\mathrm{H}_{\mathrm{range}}$	$z_{\max} - z_{\min}$
Height above (cylinder)	$\mathrm{H}_{\mathrm{above}}$	$z_{\max} - z$
Height below (cylinder)	$\mathrm{H}_{\mathrm{below}}$	$z - z_{\min}$

Figure 18: Table of geometric and covariance features from Pocock 2019 [8]

C RAND SCORE GRAPHS

C.1 K-means

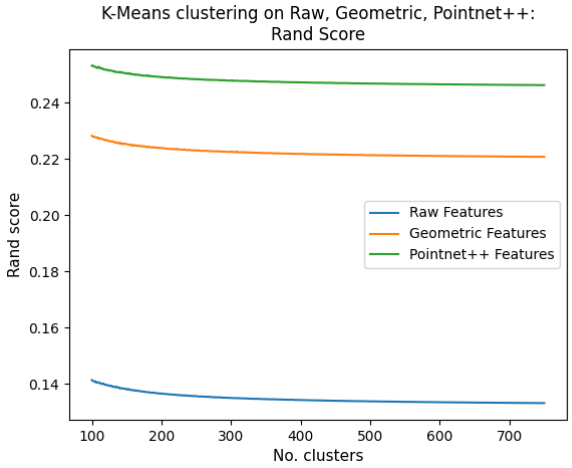

Figure 19: Rand Index on K-Means Algorithm

C.2 BIRCH

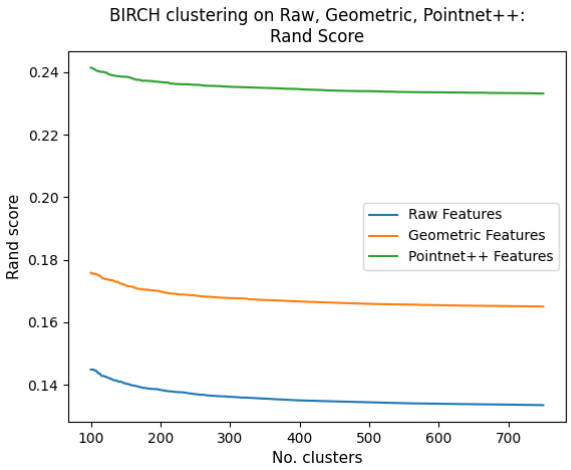

Figure 20: Rand Index on BIRCH Algorithm

C.3 Agglomerative

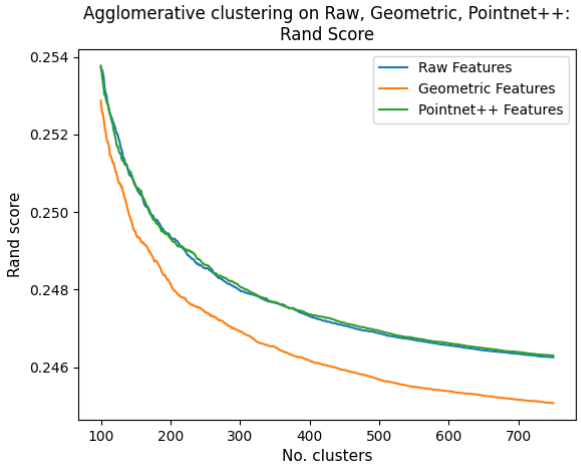

Figure 21: Rand Index on Agglomerative Algorithm

C.4 CURE

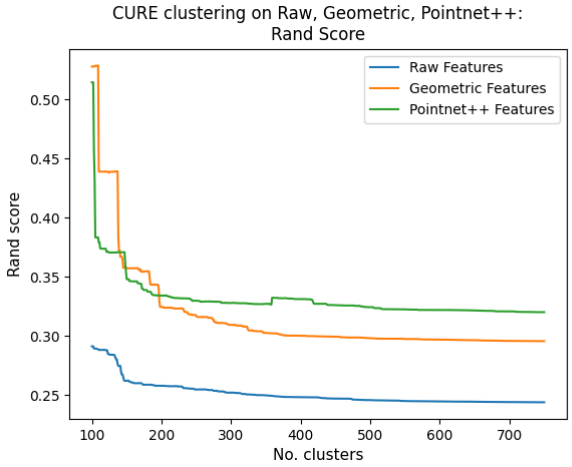

Figure 22: Rand Index on CURE Algorithm

D INTERSECTION OVER UNION GRAPHS

D.1 K-Means

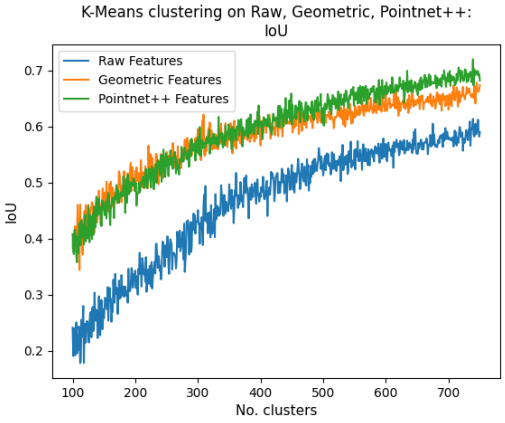

Figure 23: IoU on K-Means Algorithm

D.2 BIRCH

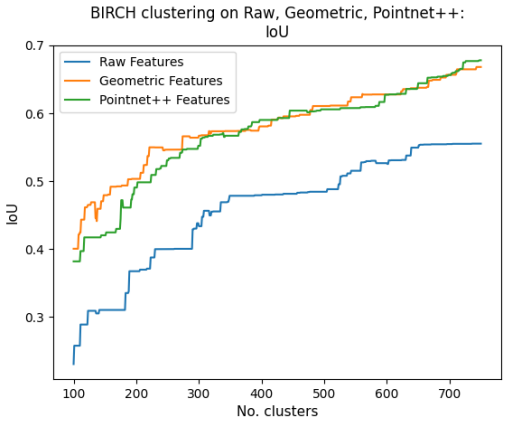

Figure 24: IoU score on BIRCH Algorithm

D.3 Agglomerative

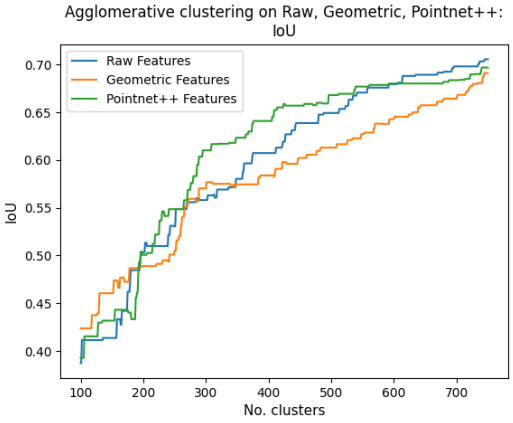

Figure 25: IoU Score on Agglomerative Algorithm

D.4 CURE

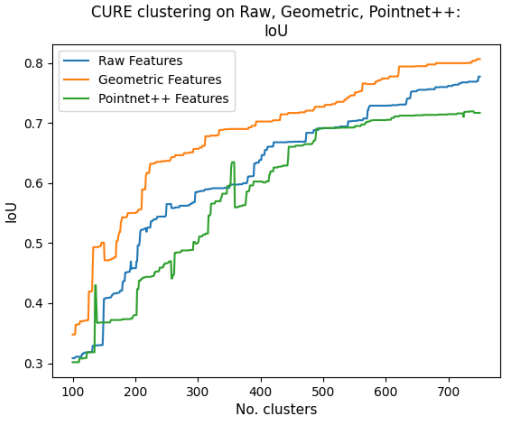

Figure 26: IoU Score on CURE Algorithm

E MEAN ERROR GRAPHS

E.1 K-means

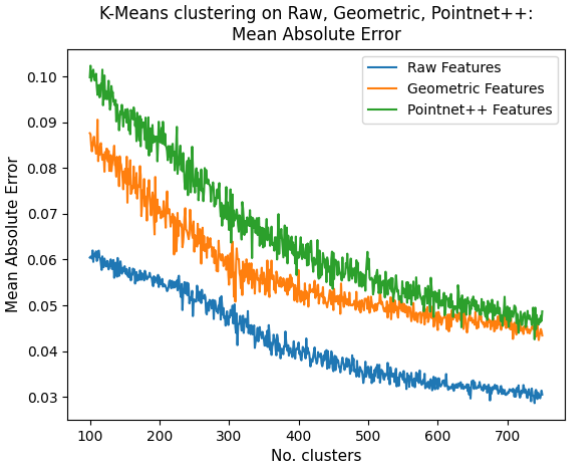

Figure 27: Mean Absolute Error on K-Means Algorithm

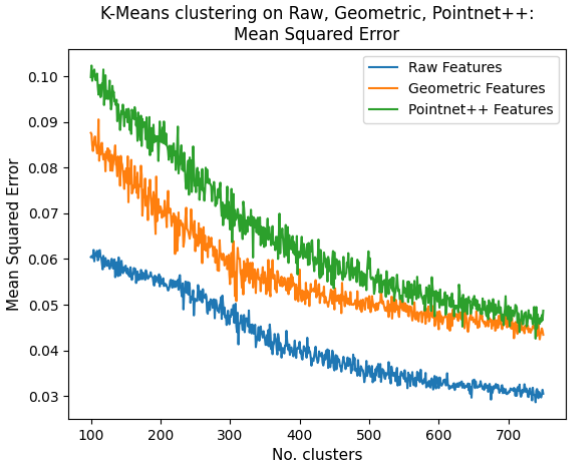

Figure 28: Mean Squared Error on K-Means Algorithm

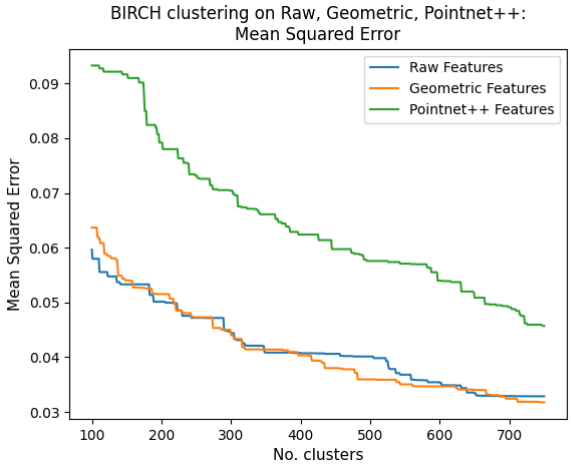

Figure 30: Mean Squared Error on BIRCH Algorithm

E.2 BIRCH

E.3 Agglomerative

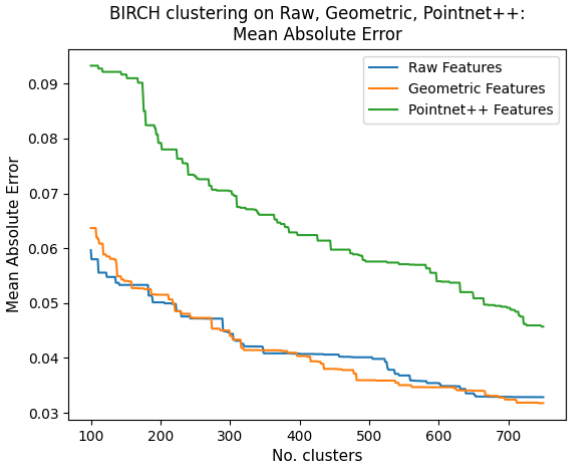

Figure 29: Mean Absolute Error on BIRCH Algorithm

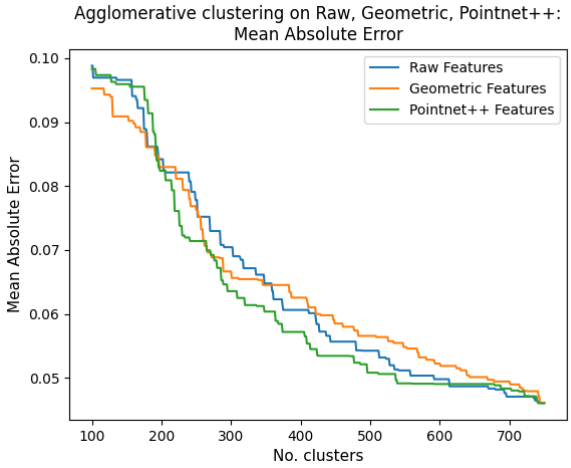

Figure 31: Mean Absolute Error on Agglomerative Algorithm

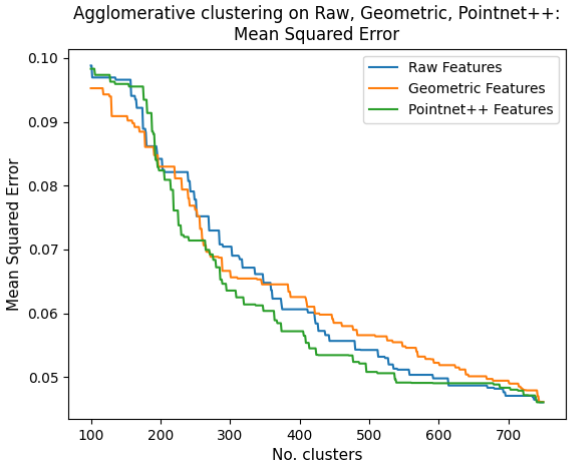

Figure 32: Mean Squared Error on Agglomerative Algorithm

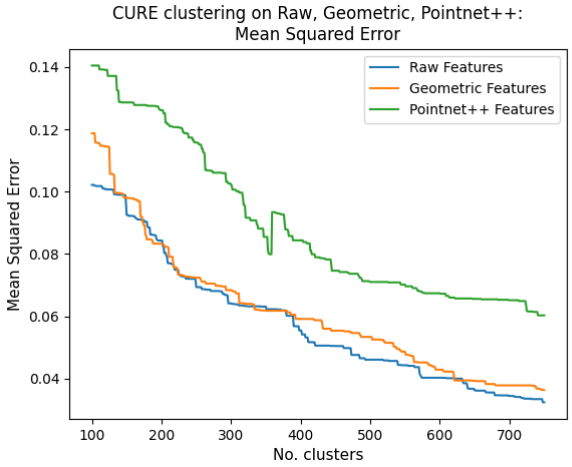

Figure 34: Mean Squared Error on CURE Algorithm

E.4 CURE

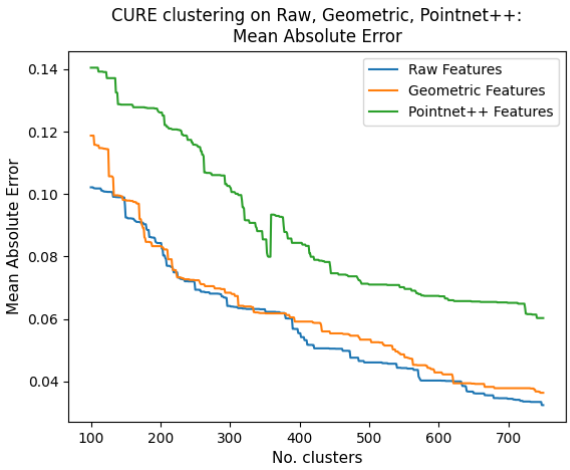

Figure 33: Mean Absolute Error on CURE Algorithm

F POINT CLOUD VISUALISATION

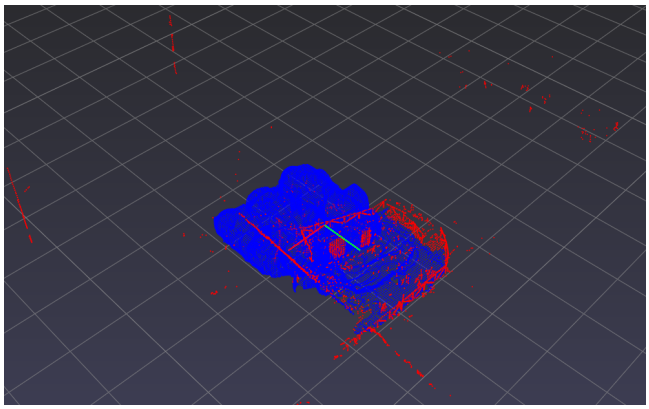

Figure 35: Ground truth labels

F.1 Agglomerative clustering

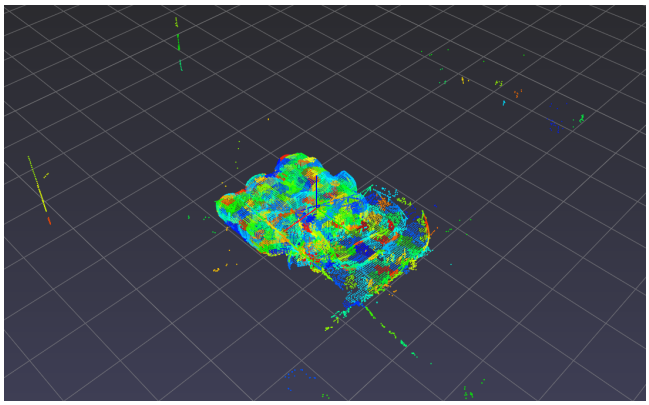

Figure 36: Clusters: Agglomerative clustering at 500 clusters

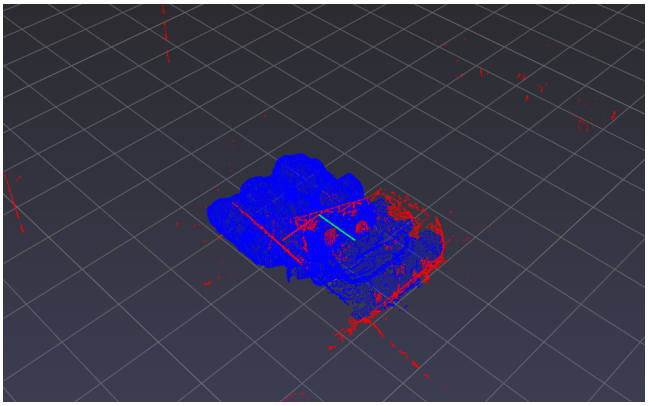

Figure 37: Predicted truth: Agglomerative clustering at 500 clusters

F.2 K-means clustering

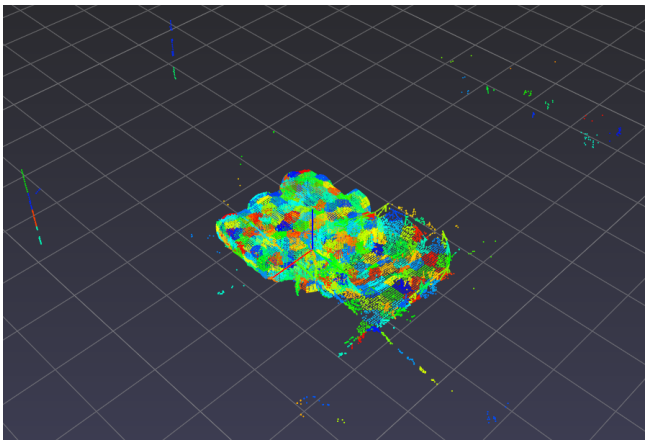

Figure 38: Clusters: K-means clustering at 500 clusters

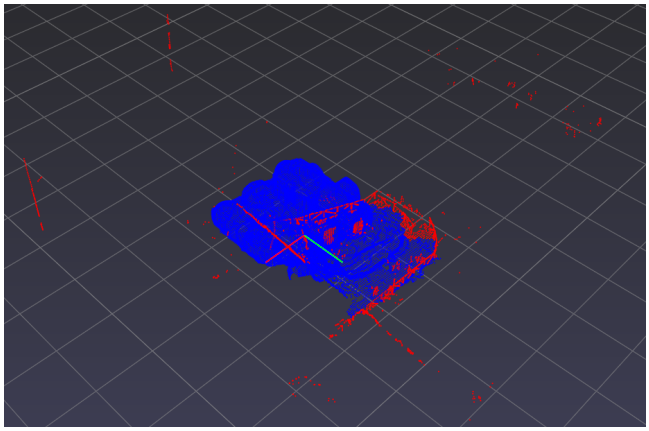

Figure 39: Predicted Truth: K-means clustering at 500 clusters