KLE Technological University

Creating Value
Leveraging Knowledge

School

of

Electronics and Communication Engineering

Mini Project Report

on

POINT CLOUD UPSAMPLING AND REFINEMENT

By:

1. **Sanskruti BK** USN:01FE18BEC258
2. **Avinash Reddy T** USN:01FE18BCS057
3. **Rasheed Nadaf** USN:01FE18BEC132
4. **Megha Somaradder** USN:01FE18BEC079

Semester: V, 2020-2021

Under the Guidance of

Ramesh Ashok Tabib

K.L.E SOCIETY'S
KLE Technological University,
HUBBALLI-580031
2020-2021

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that project entitled **" Point cloud upsampling and refinement "** is a bonafide work carried out by the student team of **"Sanskruti B Karabasannavar: 01FE18BEC258, Avinash Reddy T: 01FE18BCS057, Rasheed Nadaf: 01FE18BEC132 , Megha Somaradder :01FE18BEC079 "**. The project report has been approved as it satisfies the requirements with respect to the mini project work prescribed by the university curriculum for BE (V Semester) in School of Electronics and Communication Engineering of KLE Technological University for the academic year 2020-2021.

Ramesh Ashok Tabib	**Nalini C. Iyer**	**N. H. Ayachit**
Guide	**Head of School**	**Registrar**

External Viva:

Name of Examiners **Signature with date**

1.

2.

ACKNOWLEDGMENT

The feeling of fulfillment that lies within the effective completion of our Mini-project would be unfinished without citing the names of the individuals who made a difference in completing this project as their consistent direction, support and encouragement brought about in its realization. We are thankful to our esteemed institute KLE Technological University, Hubballi which has provided us an opportunity to fulfill the most cherished desire to reach our goal. We express a deep sense of appreciation and gratitude to our Head of School of Electronics and Communication, Dr. Nalini Iyer for giving us the motivation and for having provided us academic environment which nurtured our practical skills contributing to the success of our project. Furthermore, we would also like to acknowledge with much appreciation the decisive role of our guide Dr. Ramesh Tabib, for encouraging our efforts and guiding us in the right manner. We have to appreciate the guidance given by other supervisors as well as the panels especially in our project management that has improved our project management skills. Thanks for their comments and advice. Lastly, we would like to thank all our classmates for their support and help.

-The project team
Sanskruti B.Karabasannavar:
Avinash Reddy T
Megha Somaradder
Rasheed Nadaf

ABSTRACT

Point Cloud is a fundamental 3D representation and an important type of geometric data structure that has drawn increasing attention due to the popularity of various depth scanning devices. The problem with Raw point clouds that they are sparse, non-uniform, and do not have any proper spatial order. Hence to meet these challenges we tried implementing different algorithms and worked with point clouds to up sample and refine such sparse point clouds in order to reconstruct and to have a better representation of point clouds. We tested our algorithm with the dataset available and optimized our algorithm to achieve desired results. In the process the team performed a literature survey, we surveyed through many research papers to gain extensive knowledge about point clouds and other network architectures of Point Net, Point Net++, and PU-Net. The report gives a brief description of how we used different approaches and algorithms and chose the best approach to achieve our goal.

Contents

List of Figures

Chapter 1

Introduction to Upsampling of Point Clouds

Point clouds are the simplest representation of 3D models. They are nothing but a collection of individual points plotted in 3D. Every point represents several measurements including its coordinates along the X, Y, and Z axes and sometimes additional information such as normal, color (RGB format), and luminance value. Point clouds have received wide popularity due to their effective means to process 3D geometry and compact representation of 3D data. Raw point clouds which are generated from depth cameras and LIDAR sensors are often sparse and non-uniform and hence many key structures will be missing. Hence comes in the picture of up-sampling and refinement is required because it leads to better representation and reconstruction of 3D objects. Dealing with 3D points than a 2D grid of pixels like an image poses new challenges. Point clouds do not have any spatial order and regular structure hence up-sampling and refinement are required so that data is efficiently represented and can be used for other applications like CAD modeling

1.1 Motivation

3D representation of data has gained wide popularity lately, owing to the development of various depth scanning devices. However, the data obtained from these devices need to be pre-processed for better representation of the underlying geometry. Upsampling the sparse point clouds is one such processing method. We plan to propose a deep learning method to process unstructured point cloud data and to use deep learning to achieve various tasks involving refinement and upsampling of point clouds. There are many other representations present for 3D data, ex: meshes, voxels, grids, etc but these representations lead to unnecessarily voluminous data while rendering. The point clouds are generally created using lidar sensors, but the resultant point cloud is often sparse and noisy. Thus up-sampling and refinement process is important for better representation and post-processing of point clouds.

1.2 Objectives

i.Perform an extensive literature survey and provide analysis on the existing methods for refinement and up-sampling of the point cloud.
ii.Use different algorithms to perform up-sampling and refinement and compare the results.

1.3 Literature survey

I.PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation.[1]

PointNet is an unified architecture that is used for number of 3d recognisation tasks such as classification, segmentation and part segmentation. This architecture learns both local and global features efficiently.The architecture is divided into two parts:
Classification Network: This network takes n points as input and then applies input and feature transformations and the with the help of max-pooling point features are aggregated. For k classes, output classification scores are received.
Segmentation network:This network is an extension of the classification network. The global and local features are concatenated with outputs per point scores. ReLU along with batch norm is used for all layers. Finally, dropout layers are used for the last mlp in the classification net. The spatial encoding of each point is learned to aggregate all individual point features to a global point cloud signature is the basic idea of PointNet.

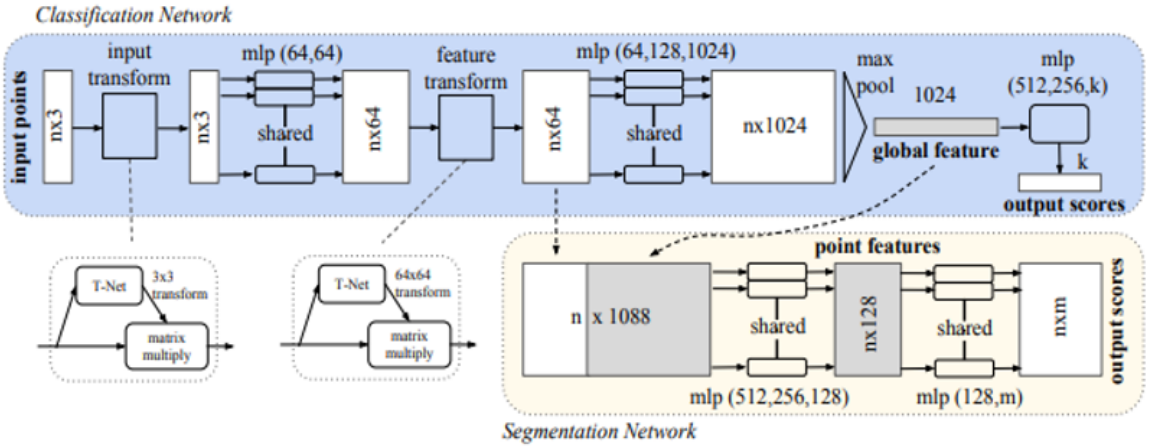


Figure 1.1: PointNet Architecture:
Classification network takes n points as input, applies feature and input transformations, mlp converts $n \times 3$ dimension to $n \times 1024$. Max pooling layer aggregates the point features in a feature vector. k classification score is the output for classification. Segmentation concatenates both local and global features and provides $n \times m$ output score.

Dataset used: Classification: ModelNet40 ,3D Object Part Segmentation : ShapeNet part data set.

Advantages:
1.A unified architecture is provided by PointNet for various applications including object classification, part segmentation, and scene semantic parsing.
2.Over one million points can be processed for the point cloud classification network by the Point-Net. This network is robust with respect to corruption and input perturbation.

Limitation: Local structures induced by the metric space cannot be captured by PointNet.Thus limiting its ability to recognize fine-grained patterns and generalizability to complex scenes.

II. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space[2]

PointNet++ architecture applies PointNet recursively on nested partitioning of the input point set by introducing a hierarchical neural network. The local features are learnt in this network by exploiting metric space distances with increasing contextual scales. The architecture proposes two novels set abstraction layers that according to local point densities aggregate multi-scale information to handle the non-uniform point sampling issue.The design of PointNet++ addresses two issues: how to generate the partitioning of the point set, and how to abstract sets of points or local features through a local feature learner.

Datasets used: MNIST, ModelNet40, SHREC15 and ScanNet.

Advantages: Along with global features, local features are learnt efficiently and effectively.

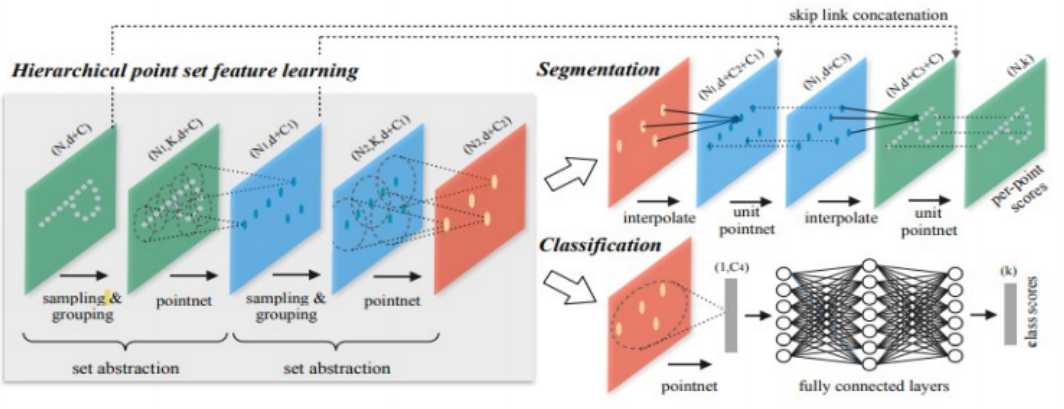


Figure 1.2: PointNet++ Architecture:
Set abstraction level consist of 3 layers they are sampling, grouping and pointnet layer. PointNet is used as local feature learner. Heirarchical feature learning is applied to obtain both local and global features. This architecture can be applied for both classification and segmentation task.

III.PU-Net: Point Cloud Upsampling Network[3]

The PU-Net architecture uses a network with a joint loss function at a patch level for the up-sampled points to provide a uniform distribution on the underlying surface. The algorithm used here is to first learn the multi-level features per point and then the point set is expanded via a multi-branch convolution unit implicitly in feature space. The expanded feature is then divided into a multitude of features which are finally reconstructed into an up-sampled point set.

Datasets used: Training: SHREC15 ,Testing: ModelNet40 and ShapeNet

Advantages: PU-Net learns geometry semantics of point-based patches from 3D models and then applies the learned knowledge to upsample a given point cloud.

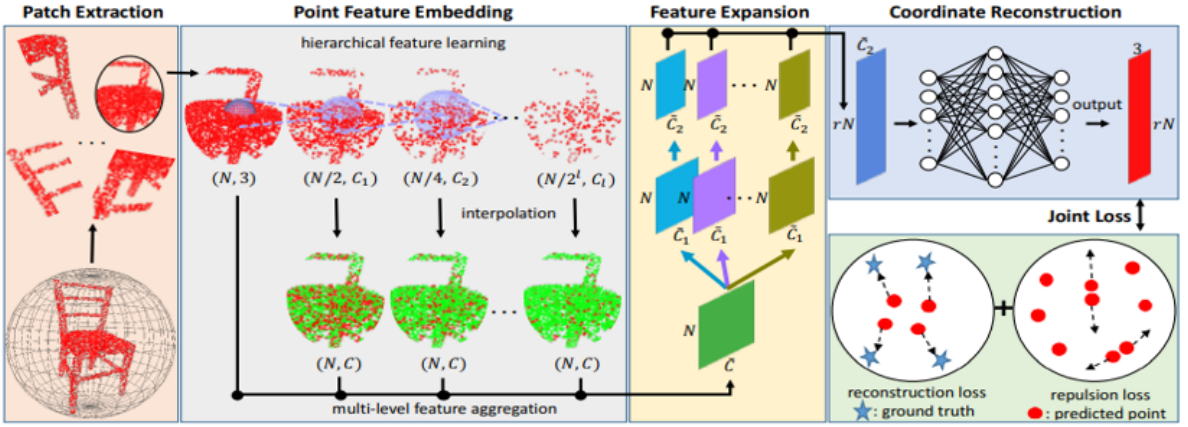


Figure 1.3: PU-Net Architecture

Input contains n points and output obtained has $r \times n$ points where r is the sampling rate. With interpolation we restore different level features for N points, with the help of convolution we reduce all level features to fixed dimension C. Reconstruction loss and repulsion loss is applied while training of PU-Net.

1.4 Problem statement

Refinement and Up-sampling of point cloud 3D representation of data have gained wide popularity lately, owing to the development of various depth scanning devices. However, the data obtained from these devices need to be pre-processed for better representation of underlying geometry. Upsampling the sparse point clouds is one such processing method. We plan to process a deep learning method to process unstructured point cloud data and to use deep learning to achieve various task involving refinement and up-sampling of the point clouds.

1.5 Application in societal context

Point clouds are being used for several purposes, they are used to create 3D CAD models for manufactured parts, animation, rendering, mass customization applications. They are also used for metrology and quality inspection and for a multitude of visualization

Up-sampling of point clouds is useful for better representation, reconstruction, rendering, and analysis of 3D models. The up-sampling and refinement process is required because it leads to shape completion of the problem of estimating the complete geometry of objects from partial observations which is very important in many vision-based robotics applications.

1.6 Project planning

Gantt chart is a project management tool which helps in planning and scheduling of projects of all sizes and useful for simplifying complex projects.

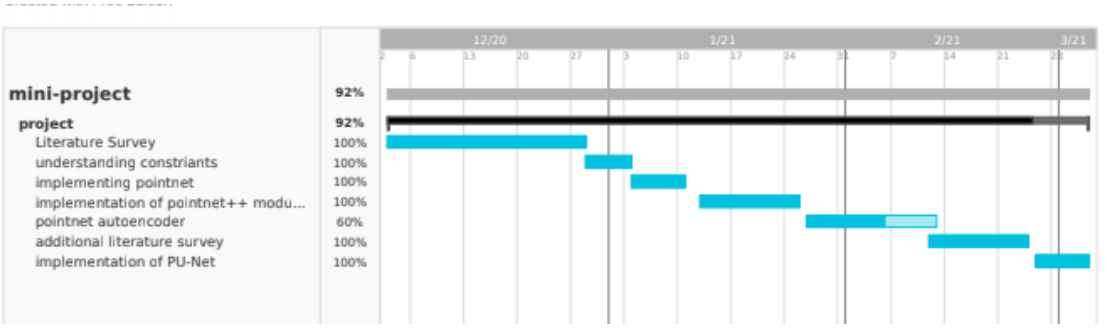


Figure 1.4: Gantt chart showing timeline of the project

1.7 Organization of the report

In Chapter 1 which is the current chapter, we give brief description about our project topic, Refinement and Upsampling of point clouds, motivation for choosing the topic, the literature survey done by the team, formation of problem statement, project planning and applications related to topic. Chapter 2, focuses on designing the system, its functional block diagram. It also concentrates on alternate approach the team was able to generate and selection of final design which gives optimal solution. Chapter 3, concentrates on implementation of the final design and final system architecture, which is, PU-net and PointNet autoencoder and their algorithms as well as flow charts. Chapter 4, briefs on optimization techniques that has been used to improve the final design and it's justification. Chapter 5, is all analyzing the results obtained which are the images of the input point cloud and output point cloud. Chapter 6, provides the conclusion and briefs on the future scope of the project.

Chapter 2

System Design: Upsampling of Point Clouds

This chapter is about system design in which, design alternatives and final design are included.

2.1 Functional block diagram for upsamplig of point cloud.

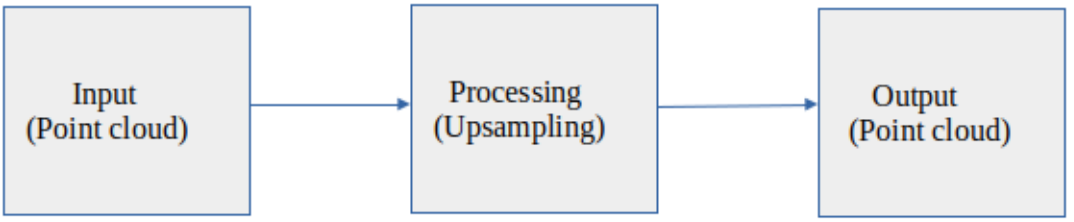


Figure 2.1: Functional Block Diagram for Upsampling of Point Cloud

2.2 Design alternatives for upsampling of point cloud

We worked on two main design alternatives and they are:

I.PointNet Autoencoder: The pointnet autoencoder is an up-gradation of pointnet architecture. The pointnet architecture successfully tackles the point clouds and is able to represent the point cloud in the form a feature vector. This feature vector contains all the information about the neighbouring points.Manipulating this vector might be fruitful in getting up-sampled point clouds. We have progressively up-convoluted the feature vector to get a bigger feature vector. This feature vector has been multiplied in a feature space and the features are split and the no of points might increase. At end this feature vector is split into N'×3 where N' is the no

of points in the upsampled point cloud.

II.PU-Net Upsampling: The PU-Net architecture uses pointnet++ as the frontal part of the network to get both the global and feature vectors from the 3D objects which is very much essential in learning the information about the underlying surface.In order to group features from different scales pointnet++ has 2 methods,MSG and MRG. We use MRG(Multi-resolution Grouping) method which allows the architecture to concatenate the features according to their densities.

2.3 Final design chosen for upsampling of point cloud

The final design that we have chosen is PU-Net upsampling, PointNet Autoencoder failed to capture the geometry when the data is sparse and also overlapping of the points occurs at the corners, Pu-Net is able to capture the geometry even if the data is sparse and it does the uniform sampling of the point clouds which is missing in case of PointNet Autoencoder.

Chapter 3

Implementation Details for Upsampling of Point Clouds

In this chapter we provide implementation details that include final system architecture, algorithm, flowchart and the block diagram of the proposed data.

3.1 PointNet autoencoder

3.1.1 Specifications and final system architecture

Autoencoder is an unsupervised learning Artificial neural network, that learns to efficiently compress the data and encode the data. And it also reconstructs back the data from encoded representation.
Autoencoder consist of 3 parts:

1.Encoder: The encoder part reduces the input dimension and compress the data to have encoded representation.We use PointNet architecture till max pool layer as encoder here.

2.Bottleneck layer: It contains the compressed representation of the data.Max pool layer is a bottleneck layer in this deign.

3.Decoder: It learns how to reconstruct the original data from the encoded representation.To reconstruct the data we use fully connected layer.

Architecture: Before feeding point cloud to the model we sample the point cloud to convert from $n \times 3$ dimension to $m \times 3$, where m and n are number of points in a point cloud and $m > n$(we have chosen number of samples i.e m as 2048 for our model). This sampled input is given to data-dependent spatial transformer network that canonicalize the data, and this output is given to MLP that will increase the dimension from $n \times 3$ to $n \times 64$, $n \times 128$ and so on till $n \times 1024$. Now the max pooling layer takes maximum value from each feature and forms a feature vector having dimension 1×1024. And this output is given to fully connected layer i.e decoder that converts 1×1024 dimension to 1×2048,1×4096 and so on till 1×6144 and finally reshape it to $1 \times m \times 3$ i.e the upsampled point cloud.

3.1.2 PointNet autoencoder algorithm

Result: Upsampled point cloud.
for *Each point cloud in the dataset* **do**
    Sample the point cloud.
    Perform input and feature transform.
    Convert $n \times 3$ dimensions to $n \times 1024$ using MLP.
    Perform max pooling to obtain 1×1024 dimensional feature vector
    Input the feature vector to fully connected layers, to obtain 1×6144 dimensional vector
    Reshape to obtain the upsampled output.
    Calculate chamfer distance.
end

3.1.3 PointNet autoencoder flow chart

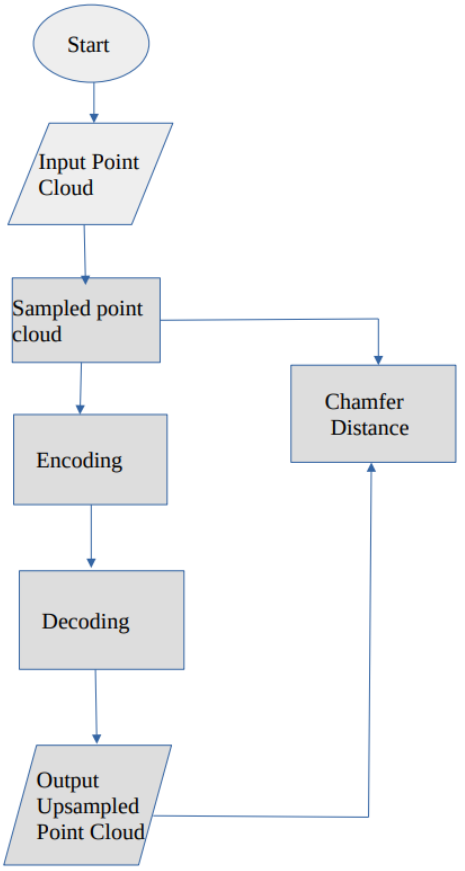


Figure 3.1: PointNet autoencoder flow chart

3.1.4 PointNet autoencoder block diagram

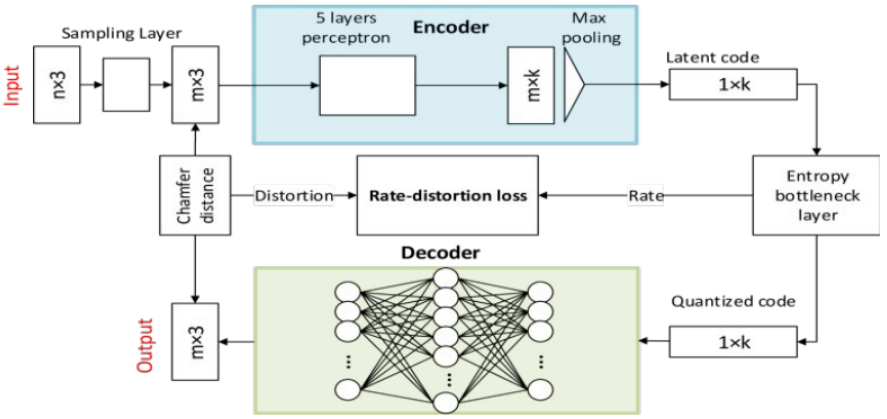


Figure 3.2: PointNet autoencoder block Diagram

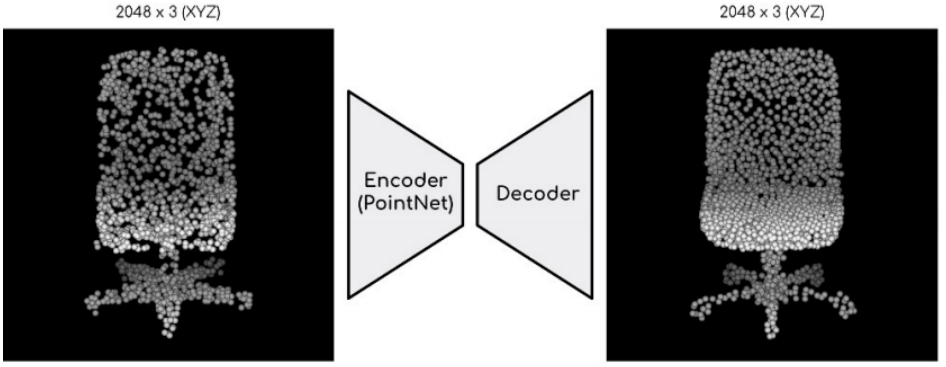


Figure 3.3: Upsampled Point Cloud

3.2 PU-Net

3.2.1 Specification and final system architecture:

Working with point clouds directly is challenging due to the irregularity and sparseness of the point cloud. Pu-Net is a data-driven point cloud upsampling network. This upsampling network is built upon the principles of pointNet++, In the case of PointNet upsampling the network did not capture the local structure induced by metric space, So in this with the help of Point-Net++ architecture hierarchical feature learning happens and due to which local structure is captured efficiently, and this leads to a better result.

It has mainly four components:
1.Patch extraction:

•Sets of 3D objects are collected as information for training. A vast area of shapes including smooth surfaces and shapes with sharp edges is contained in the sample.
•This is done so considering the network to upsample a point cloud, the knowledge of local geometry must be captured.
•This allows the training of network and the knowledge of geometry semantics to be approached patch-based.
•M points are randomly selected on the surface of these objects, a face patch is grown for each selected point making sure any point on the patch is within the given geodesic distance (d).
•To randomly generate N points, Poisson disk sampling is used as the referenced ground truth point distribution on the patch.
•Both local and global context lead to a smooth and uniform performance in our upsampling task. As a result, we varied d. sizes, allowing us to extract patches of points from the previous items of different sizes and densities.

2.Point feature embedding:
we consider two feature learning strategies mentioned below and these are used to learn both global and local features.consider two feature learning strategies mentioned below and these are used to learn both global and local features. Hierarchical feature learning:
•Extraction of local and global features has been shown to be efficient by progressively collecting features of increasing scales in a hierarchy. As a result, we use PointNet++'s recently proposed hierarchical feature learning mechanism as the very frontal part of our network.
•We use a relatively small grouping radius in each level to adopt hierarchical feature learning for point cloud upsampling, since generating new points typically requires more of the local context than high-level recognition tasks.

3.Feature expansion:
•Increasing the number of features in the feature space after the point feature embedding part. Since points and features are interchangeable, this is equivalent to increasing the number of points.
•We emphasise that since the function in the embedding space has already encapsulated the relative spatial information from the neighbourhood points through efficient multi-level feature aggregation, we do not need to specifically consider it when conducting this feature expansion process.
•Separate convolutions can be applied to the r feature sets to perform this feature expansion process. It can also be implemented using clustered convolution, which is more computationally effective.

4.Coordinate reconstruction:

•The 3D coordinates of output points from the extended function with the scale of rN × C˜2are reconstructed in this section.
•Regress the 3D coordinates on the function of each point using a series of completely connected layers, with the final output being the upsampled point coordinates rN × 3.

3.2.2 PU-Net algorithm:

Result: Upsampled point cloud.
for *Each point cloud in the dataset* **do**
  Read point cloud.
  Randomly select M points in the point cloud.
  Grow surface patches around these M points.
  Apply hierarchical Feature learning to obtain local and global geometry from the patchs.
  Expand the number of features in the feature space.
  3D coordinate reconstruction of the output points from expanded feature.
  Apply Joint loss function to obtain reconstruction loss and perform refinement.
end

3.2.3 PU-Net flow chart:

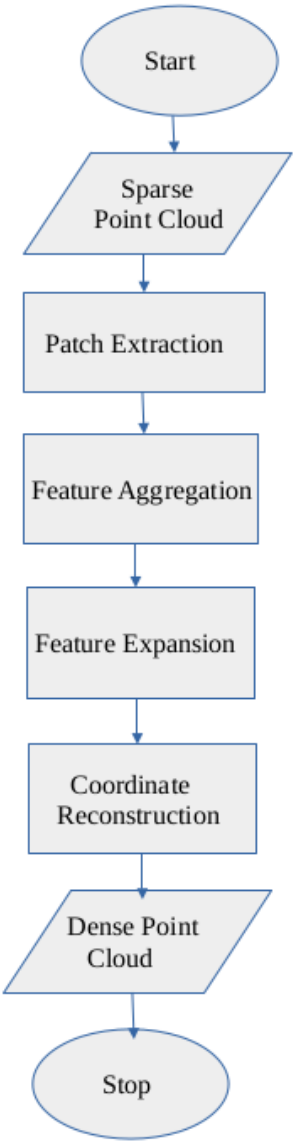


Figure 3.4: PU-Net flow chart

3.2.4 PU-Net block diagram:

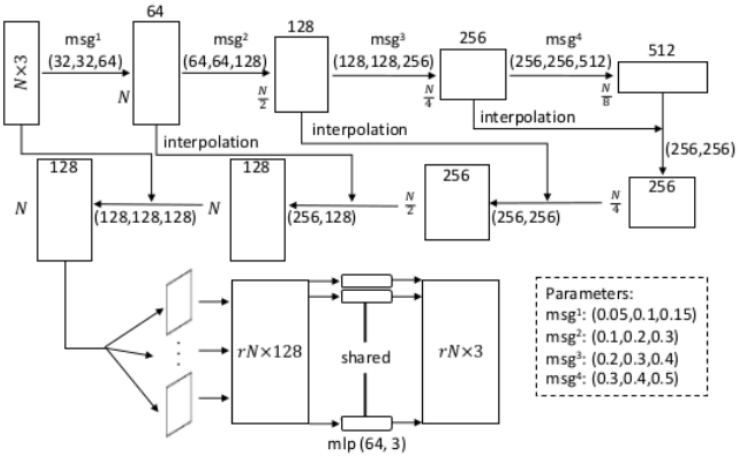


Figure 3.5: PU-Net Block Diagram

Chapter 4

Optimization for Point Cloud Upsampling

This chapter explains about optimisation, Types of optimisation, and about the optimisation that we have chosen to obtain the reconstruction loss.

4.1 Introduction to optimization

Optimization is making use of resources efficiently and effectively. Optimization is a must in making decisions and assessing physical and logical processes. Optimization helps in improving the system's performance. The motive behind optimization is to attain the best functional design. This involves optimizing characteristics of the system in use (time and energy consumption).

4.2 Types of optimization

•Continuous Optimization.
•Discrete Optimization.
•Global Optimization.
•Constrained Optimization.
•Derivative-Free Optimization.
•Linear Programming.
•Non-differentiable Optimization.

4.3 Selection and justification of optimization method

Upsampling the point cloud is not just increasing the number of points, The network must also be able to capture the geometry of the point cloud. To do this we use loss functions called the chamfer distance and earth mover's distance, But chamfer distance found to give better result than earth mover's distance so we use chamfer distance as our loss function.
Chamfer distance is the evaluation metric for two-point clouds, The two-point cloud here is a randomly sampled and upsampled point cloud. CD takes the distance of each point cloud, For each point in the point cloud CD finds the nearest point in the other set and it sums the square of the distances.CD helps the upsampled points to stay close to the parent points and it make sure points do not clutter with each other.

Chapter 5

Results and Discussions

• We have worked with 2 datasets SHREC15 and Modelnet40:
1. SHREC15: It contains .obj formats of non-rigid objects and non-rigid human statues.
2. Modelnet40: This dataset contains .off format 3D object representations of 40 different class objects like bathtub, chair, closet, etc
• We convert these .obj and .off formats into point clouds by uniformly sampling points all over the surface of the objects.
•As there is no different train and test data set available we sub-sample the existing point cloud and use it as input This chapter is about results and discussions on our design.

5.1 PointNet autoencoder

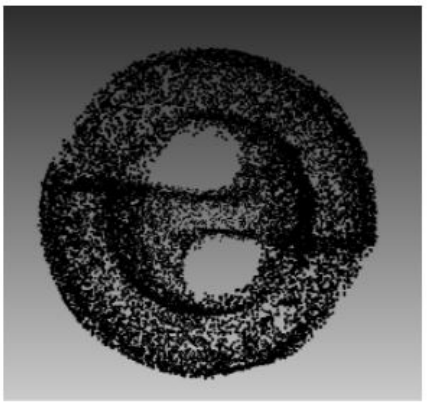
Result: The dataset used here is ModelNet40, the number of epochs that we have trained is 20.The output point cloud has points overlapped at the corners.

Figure 5.1: Upsampled Point Cloud using Autoencoder

5.2 PU-Net

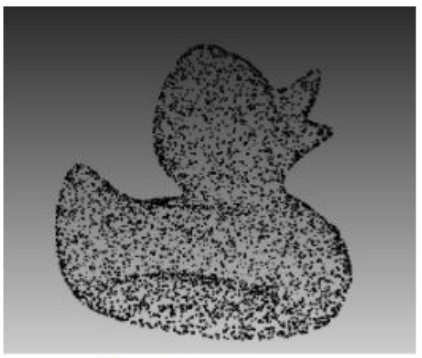
Results: The data set used is shrec15 for training and ModelNet40 for testing.The output obtained is upsampled, and the sampling ratio is 4.The model is trained for 250 epochs.

Input point cloud
No of vertices: 5000

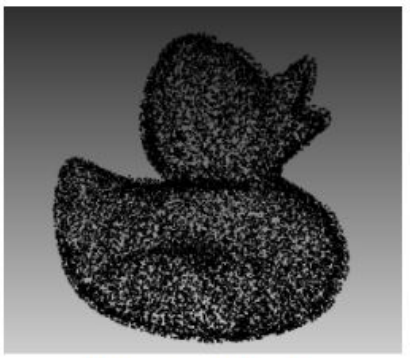


Output point cloud
No of vertices:20,000

No of epochs=250
Up-sampling ratio=4

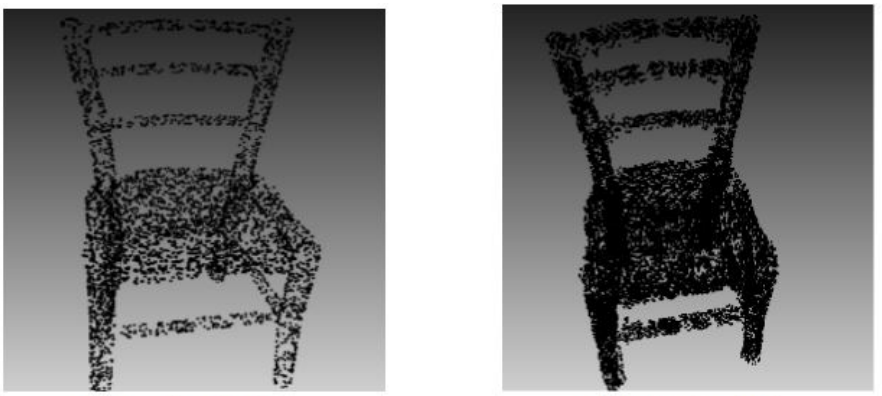


Input point cloud
No of vertices: 5000



Output point cloud
No of vertices:20,000

No of epochs=250
Up-sampling ratio=4



Input point cloud
No of vertices: 5000

Output point cloud
No of vertices:20,000

No of epochs=250
Up-sampling ratio=4

Loss function	Loss value
Total Loss	0.1491
Weighted Emd Loss(reconstruction loss)	0.1281
Repulsion Loss	0.0210

Figure 5.2: Loss Function

5.3 Result Analysis

In case of PointNet autoencoder the output obtained is not upsampled, and it did not capture the geometry and the points are overlapped at the corners. This is due to the number of epochs that we trained is very less(we trained only for 20 epochs) and also the error function that we used was the mean square so instead, we have to use chamfer distance and also we have to train the model for at least 80-100 epochs.
In case of PU-Net the Loss function used here is the joint re-construction loss function which is a combination if 2 loss function that ensures that points are not too far and also not too close to the parent points,the constituent loss functions are:
1. Reconstruction Loss: which helps the up-sampled points to stay close to the parent points.
2. Repulsion Loss: This helps the points not to clutter with each other.
Using PU-Net the output obtained is upsampled and it performed better than PointNet autoencoder.

Chapter 6

Conclusions and Future Scope

This chapter is about the final words on our design.

6.1 Conclusion

Upsampling of the point cloud is challenging due to the irregularity and sparseness of the data. The model to which we input the point cloud must be invariant to certain geometric transformations and also it should be invariant to permutations of the input set. PointNet architecture helps us to achieve these constraints.Incase of pointNet autoencoder,The fully connected layer at the output helps us to obtain upsampled output, and chamfer distance helps us to propagate back if the geometry is not captured properly.PU-Net which uses PointNet++ architecture achieves better results in upsampling of the point cloud.

6.2 Future scope

We would like to work on more robust models such as PU-GAN and PCN which are more robust, we would also like to improve our pointnet autoencoder model by using different loss functions.

Bibliography

[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[2] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[3] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.