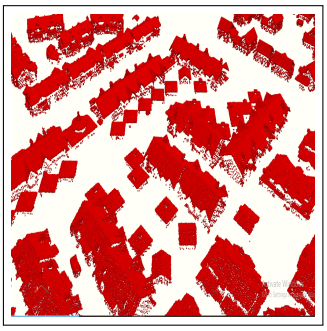
DEPARTMENT OF
ENGINEERING DESIGN
INDIAN INSTITUTE OF
TECHNOLOGY
MADRAS
CHENNAI-600 036

Attention-based Deep Learning Approaches towards Classification, Retrieval and Shape Completion of ALS Roof Point Clouds



A thesis

Submitted by

Dimple A. Shajahan

For the award of the degree

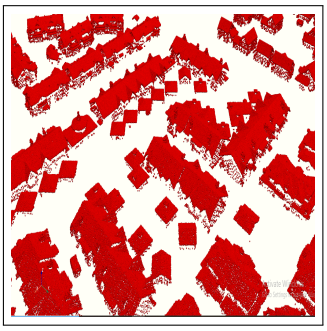
Of

DOCTOR OF PHILOSOPHY

July, 2021

DEPARTMENT OF
ENGINEERING DESIGN
INDIAN INSTITUTE OF
TECHNOLOGY
MADRAS
CHENNAI-600 036

Attention-based Deep Learning Approaches towards Classification, Retrieval and Shape Completion of ALS Roof Point Clouds



A thesis

Submitted by

Dimple A. Shajahan

For the award of the degree

Of

DOCTOR OF PHILOSOPHY

July, 2021

QUOTATIONS

The aim (of education) must be the training of independently acting and thinking individuals who, however, see in the service to the community their highest life problem -

ALBERT EINSTEIN

DEDICATION

To my beloved

THESIS CERTIFICATE

This is to undertake that the thesis titled, ***Attention-based Deep Learning Approaches towards Classification, Retrieval and Shape Completion of ALS Roof Point Clouds*** submitted by me to the Indian Institute of Technology Madras, for the award of **Ph.D.** is a bonafide record of the research work done by me under the supervision of **Dr. M. Ramanathan**. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Chennai 600 036 **Dimple A. Shajahan**

Date:22/07/2021

Dr. M. Ramanathan

Research Guide

Professor

Dept. of Engineering Design

IIT-Madras, 600 036

LIST OF PUBLICATIONS

The publications arising out of the work mentioned in this thesis are given as follows

1. **REFERRED JOURNALS BASED ON THE THESIS**
   (a) **Dimple A. Shajahan, Vaibhav Nayel, Ramanathan Muthuganapathy**
   Roof Classification from 3D LiDAR Point Clouds using Multi-view CNN with Self-Attention
   IEEE Geoscience and Remote Sensing Letters., Vol-17, pages=1465-1469, (2020).

   (b) **Dimple A. Shajahan, Mukund Varma T, Ramanathan Muthuganapathy**
   Point Transformer for Shape Classification and Retrieval of Urban Roof Point Clouds
   IEEE Geoscience and Remote Sensing Letters,doi =10.1109/LGRS.2021.30614222021 (2021).

2. **JOURNALS TO BE SUBMITTED BASED ON THE THESIS**
   (a) **Dimple A. Shajahan, Mukund Varma T, Ramanathan Muthuganapathy**
   Point Completion Transformer for Shape Completion of ALS Roof Point Clouds
   IEEE Transactions on Geoscience and Remote Sensing.

3. **PRESENTATIONS IN CONFERENCES**
   Dimple A. Shajahan, Vaibhav Nayel, Ramanathan Muthuganapathy
   Roof Classification from 3D LiDAR Point Clouds using Multi-view CNN with Self-Attention
   32nd Conference on Graphics, Patterns and Images - SIBGRAPI 2019 October at Rio De Janeiro, Brazil.

ACKNOWLEDGEMENTS

This thesis is the culmination of my five years of work at IITM, and there are numerous people I wish to thank for contributing to this research.

First and foremost, my deepest gratitude goes to my guide and mentor, Prof. M. Ramanathan, who has been a most reliable supervisor in guiding me throughout the research. I am also thankful to him for accepting me as his Ph.D. student and directing me to a novel and innovative research area in Computer Graphics. It would not have been possible to complete my doctoral thesis without his invaluable contributions and constant support. His intuitive thinking, positive attitude, thoughtful guidance, critical comments, and thesis correction have helped me shape up and project my works with maximum perfection. I would be indebted to him for his patience and offering the highest degree of freedom to work at my own pace and order.

I would also like to thank the anonymous reviewers of my research manuscripts for the suggestions to improve the quality of writing.

I gratefully acknowledge the support extended to me by Dr. Asokan T., HoD, Engineering Design, and my DC committee members Dr. G. Sarvana Kumar, Dr. Kavitha Arunachalam, Department of Engineering Design, and Dr. Meghana Nasre, Department of Computer Science & Engineering for their review and valuable suggestions for improving the quality of the research. I also acknowledge Prof. Srikanth Vedantam, Prof. Nilesh J. Vasa, former Heads of the Department of Engineering Design, to provide me with all the necessary facilities. I also thank Dr. Palaniappan

Ramu in my department, Dr. N.S. Narayanaswamy, Department of Computer Science, for their unstinted support during my course work. I thank all other faculties and non-teaching staff of the Engineering Design department of IIT Madras.

I cannot quantify the contributions and support of my friend Mukund Varma .T. for helping me towards innovative research and framing the structure of the manuscript. The series of online discussions with him during this pandemic time has motivated me to keep up with the current trends in deep learning, which has enhanced my research. I want to thank him from the bottom of my heart for working with me. I am also immensely thankful to Vaibhav Nayel, Mohammad Ajwahir, and Kushan Raj for their contributions to my research.

I am deeply thankful to all my peers at Advanced Geometric and Computing Lab (AGCL), IIT Madras, Lakshmi Priya, Manoj Kumar, M.V. L.Bharadwaj, Safeer Babu, and Amal Dev for their beneficial technical discussions and for helping me broaden my knowledge and outlook. I would also like to thank Jiju Peethambaran for inspiring me to select Urban Reconstruction as my research topic through his wonderful works in this area. I am also grateful to Subhasree. M. for the help and research directions offered to me during my contact programme. I am also thankful to my peers in other labs, Mahindra, Divya, Radhika, and Varghese, for discussing various topics related to machine learning. I cannot forget the help extended to me by my CSE friends Vipin N S, Nada Abdul Majeed, and Nikhita during the coursework of my Ph.D. programme.

I express my sincere thanks to the CCE chairman and all the staff members of CCE, especially Hema .R, for their advice and timely actions.

I cannot forget the support extended to me by the Principal of my parent institution, TKMCE, Dr. T.A Shahul Hameed, to grant me sufficient leave and exempt me from many duties to complete my research works. I also thank my colleagues in TKMCE, Nadera Beevi, Amal Azad, Saibi. R, and all faculty in my department for continuously supporting me throughout my research journey.

Last but not least, my family members were always supportive and giving me the freedom to pursue my research in the best way. My late father was always inspirational to me for education and in life. I always feel the blessings of my late grandfather in all aspects of my education.

And above all, I would like to thank the Almighty for blessing me with an opportunity to pursue my Ph.D. in a premier institute like IIT Madras and giving me physical and mental strength throughout my doctoral studies in all due respects to complete this thesis successfully.

ABSTRACT

KEYWORDS: ALS Roof; Point Clouds ; Deep learning; Attention; Classification; Retrieval; Shape Completion.

The 3-D digital representations of building models from Airborne Laser Scan (ALS) point clouds have many applications in multiple domains such as Geographic Information Systems (GIS), Remote Sensing, Archaeology, Photogrammetry and, Computer Vision. In these applications, especially in urban modeling, frequent updation of building models is necessary for reliable and up-to-date information. The existing methods based on geometry cannot meet the demands for large-scale urban data and are not suitable for complex-shaped buildings in the real world. A recent trend is to apply machine learning methods that give better performance compared to geometric approaches. However, classical machine learning methods have several limitations, such as extracting handcrafted features and not being scalable to large volumes of data. These methods are either manual or semi-automatic and are also time-consuming, compute-expensive. Therefore, there is a high demand for fully automatic approaches for processing ALS roof point clouds, giving better performance.

The advent of deep learning has led to significant improvements across various 3-D point cloud shape analysis tasks. Still, its effectiveness is not yet fully explored in remote sensing and GIS, especially for ALS building point clouds. Since the roof is the most informative part of the building from an airborne scan, 3-D modeling of buildings based on various roof styles is significant, and prior knowledge about roof style is

advantageous for many applications. Automatic roof-top classification, retrieval, and shape completion using ALS building point clouds are relevant in this context, and performing these tasks with high accuracy is a great challenge. Most existing deep learning methods for 3-D point clouds focus on aligned, synthetic point clouds. ALS roof point clouds, on the other hand, is more challenging as it is a real dataset containing noise, outliers, missing points, sparsity, occlusions. Hence, there is a vital requirement for deep learning methods that are robust to all these issues.

Inspired by the success of attention-based methods (like transformers) in Natural Language Processing (NLP), our study focuses on incorporating attention to standard networks, introducing full-blown attention-based architectures for classification, retrieval, and shape completion tasks of ALS roof point clouds. We perform detailed robustness tests on real, unaligned datasets containing various imperfections, and our proposed methods are highly robust to these issues while still being time and memory-efficient. These results also point out that attention-based methods might be an even more natural fit for point cloud processing, as attention in its core is a set operation: implying that it is invariant to permutation and cardinality of the input elements, making it ideal for point clouds. This thesis has successfully shown the effectiveness of deep learning methods for ALS roof point cloud processing in various tasks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

3-D	Three Dimensional
ALS	Airborne Laser Scan
LiDAR	Light Detection and Ranging
CNN	Convolutional Neural Networks
MVCNN-SA	Multi-view CNN with Self-Attention
SAN	Self Attention Network
PT	Point Transformer
PCT	Point Completion Transformer
MLP	Multi-Layer Perceptrons
SDP	Scalar Dot Product
MAP	Mean Average Precision
MHA	Multi-Head Attention operation
CD	Chamfer Distance
FPS	Farthest Point Sampling
GLU	Gated Linear Unit
MVCNN	Multi-view CNN
GVCNN	Group-view CNN
SVM	Support Vector Machines
RANSAC	Random Sample Consensus
PCN	Point Completion Network
ASCN	Attentional ShapeContextNet

PFN	Point Fractal Network
GPU	Graphics Processing Unit
PAT	Point Attention Transformer
RNN	Recurrent Neural Networks
LSTM	Long Short-Term Memory
DGCNN	Dynamic Graph CNN
2-D	Two Dimensional
GIS	Geographic Information Systems
NLP	Natural Language Processing
SOTA	State-of-the-Art
LoD	Level of Detail

CHAPTER 1
INTRODUCTION

1.1 RESEARCH BACKGROUND

In recent years, there is an enormous demand for three dimensional (3-D) digital representations of building models across multiple domains such as Geographic Information Systems (GIS) [Musialski *et al.* (2013); Zhou and Neumann (2008)], Remote Sensing [Musialski *et al.* (2013)], Archaeology [Remondino (2011)], Photogrammetry [Musialski *et al.* (2013); Yan *et al.* (2014)] and, Computer Vision [Axelsson *et al.* (2018); Musialski *et al.* (2013); Lafarge and Mallet (2011, 2012); Verma *et al.* (2006)]. These building representations can be used to recreate an entire city model, which has multiple applications in urban modeling [Poullis and You (2009)]. Airborne laser scanning (ALS) point clouds are known to be an appropriate data source for urban modeling as it covers a large area quickly and directly provides spatial coordinates of various objects in the urban scene [Chen *et al.* (2017)]. An ALS captures the overhead view of an urban scene consisting of various urban objects (e.g., trees, buildings, cars, trucks, poles, etc.), where buildings are the most prominent objects for urban modeling. Figure 1.1 visualizes a complete ALS scan for an urban scene in Dales dataset [Varney *et al.* (2020)] (where the building points are highlighted in red, vegetation points in green, car/truck points in yellow and ground points in blue color). The individual building point clouds are then segmented and extracted out for urban modeling tasks. These building point clouds mainly contain points of roof-tops, as the roof is the most informative part of a building [Chen and Lin (2016); Chen *et al.* (2017)].

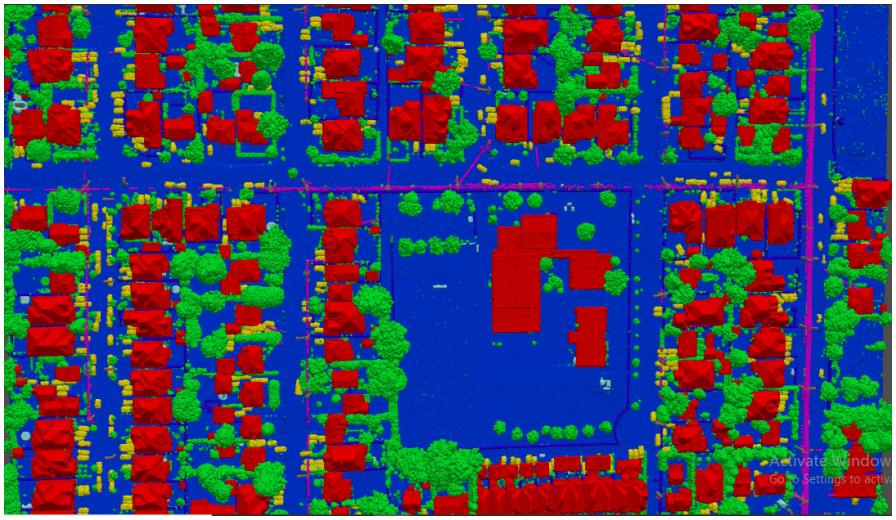


Figure 1.1: ALS point cloud of an urban scene [1](buildings:red, vegetation: green, cars/truck: yellow, and ground: blue).

Figure 1.2 illustrates an urban scene with only ALS building point clouds while Figure 1.3 visualizes the roof point clouds. Since this thesis mainly studies the processing of ALS roof point clouds, the other objects present in the scene are not relevant. Therefore, any mention of ALS roof point cloud directly refers to the segmented roof-tops from the scanned point cloud of an urban scene.

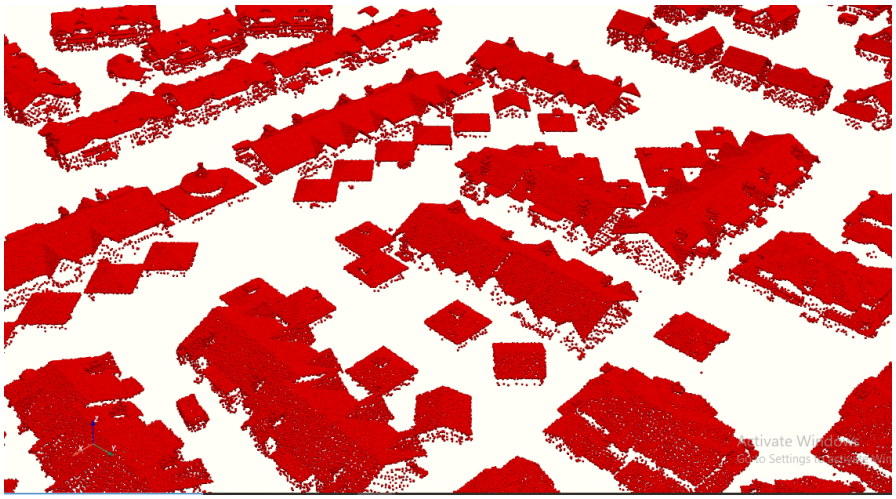


Figure 1.2: Urban scene with ALS building point clouds only [2]

[1] This figure is created using an input tile for an urban scene from Dales dataset.

[2] This figure is created using another input tile from Dales dataset.

Figure 1.3: An ALS building roof point cloud [3]

A 3-D building model can also be obtained using other acquisition techniques such as aerial photogrammetry, satellite images, synthetic aperture radar, extrusion from 2-D footprints, etc. Compared to these approaches, ALS has high precision, reliability, depth information and is not easily affected by changes in the environment [Chen *et al.* (2014*a*); Chen *et al.* (2017)]. However, the accurate analysis of an ALS building point cloud is challenging due to various imperfections present in these point clouds like sparsity, missing shapes, noisy points, etc. [Chen *et al.* (2017)]. Due to the large-scale applications of analyzing roof-tops in urban modeling, there is a vital requirement for advanced methods which facilitate strong performance and high speeds.

The existing research works for ALS building roof modeling employ several strategies of geometric methods that provide reasonably good results [Lafarge and Mallet (2012); Jung *et al.* (2017); Wu *et al.* (2017); Awrangjeb *et al.* (2018)] but with many limitations [Wichmann (2018); Wang *et al.* (2018)]. Since most of these methods do not utilize the roof style information, multiple geometric and topological constraints of the roof shapes

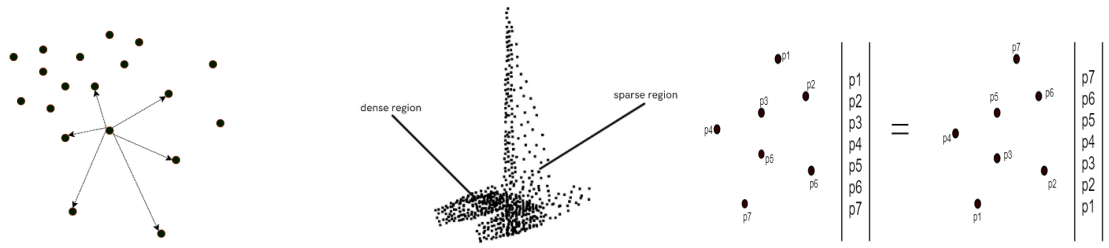
[3] This figure is from a random section of urban scene taken from an input tile in Dales dataset.

are assumed. These constraints do not generalize well and limit geometric methods to a smaller domain [Zhang *et al.* (2014)] (e.g.: dataset-specific). Additionally, these methods are strongly affected by natural variations [Gkeli and Ioannidis (2018)] present in the point set, which is significantly pronounced in real datasets like ALS roof scans.

A recent trend is to apply machine learning methods and experiments suggest that they give better performance and generalisation capacity compared to geometric approaches [Zhang *et al.* (2014)]. Some of these works include, for classification [Axelsson *et al.* (2018); Mohajeri *et al.* (2018); Zhang *et al.* (2014); Castagno and Atkins (2018)], retrieval [Liu *et al.* (2018); Chen *et al.* (2017); Chen and Lin (2016)]. However, these methods are still not scalable to large volumes of data or a variety of tasks as they rely on hand-crafted features [Zhang *et al.* (2014); Castagno and Atkins (2018); Zhang and Zhang (2018)] and are computationally complex even for simple classification tasks due to high-dimensional data [Castagno and Atkins (2018)]. Most of the existing techniques retrieve or reconstruct standard and relatively simple 3-D roof models and do not account for superstructures (i.e., dormers and chimneys) [Dehbi *et al.* (2021)]. Therefore, current methods are not capable of generalizing to diverse and complex roof shapes and are highly compute-expensive [Wichmann (2018); Wang *et al.* (2018)]. Hence, there is an inevitable requirement for efficient methods that can overcome the problems mentioned above.

Recently, deep learning methods (a special class of ML algorithms) have been proven to be extremely effective across various domains like computer vision [Lu and Shi (2021)], computer graphics [Vanegas *et al.* (2012)], and robotics [Guo *et al.* (2020); Bello *et al.*

(2020); Ahmed *et al.* (2018)]. However, the direct application of deep learning on 3-D point clouds is not easy due to the inherent nature of point clouds i.e they are unstructured (not gridded), irregular (non-uniform) and unordered (not ordered) [Bello *et al.* (2020)]. These properties are figuratively illustrated in Figure 1.4.



(a) Unstructured (b) Nonuniform (c) Unordered

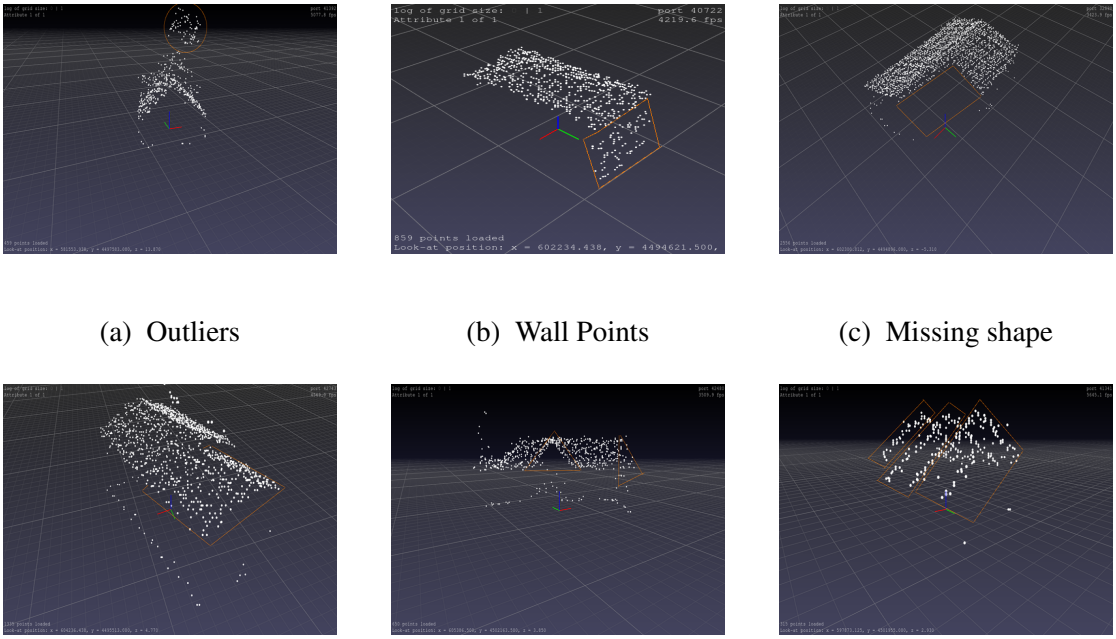
Figure 1.4: Characteristics of a point cloud

Convolutional Neural Networks (CNN) was a crucial reason for the grand success of deep learning in computer vision, and it has realized remarkable achievements in image classification [Shen (2019). However, it isn't easy to directly apply CNNs to analyze 3-D point clouds as they assume a uniform input distribution, not the case in point clouds. Therefore, researchers have tried to convert raw point clouds into more regular representations like images, voxels.

In voxel-based representations, 3-D points are converted to volumes which inherently does not allow modeling of complex shapes, leads to unnecessary artifacts, etc. Also computing high-resolution voxels leads to large computational costs and specialised hardware [Guo *et al.* (2020)]. In view-based methods, point clouds are projected onto planes that capture multiple views of the point cloud, and existing deep learning-based image feature extraction techniques are applied to these images. These generated

views are relatively low dimensional compared to voxels and are robust to holes, noise, etc. Multi-view methods showcase better performance than voxel-based methods as multiple views help capture rich geometric information compared to 3-D voxels, even though the latter contains depth information. However, view-based methods are strongly affected by self-occlusion, the number of viewpoints, and require large compute, storage requirements [Ahmed *et al.* (2018)]. Recent research has attempted to directly apply deep learning on raw xyz coordinates of a given point cloud; such methods are also called point-based methods. This does not lead to any loss of information as there is no conversion from one form of representation to another. However, these methods need to strictly ensure order invariance property to work effectively for a given input point cloud [Charles *et al.* (2017)]. However, due to the advantages of point-based methods recent research has been focused in this direction [Lu and Shi (2021); Guo *et al.* (2020)]. It is important to note here that view-based methods are still state-of-the-art in point cloud representation, and recent research in point-based methods has tried to reduce the gap.

Most existing deep learning methods for point clouds focus on aligned, synthetic point clouds. However, ALS roof point clouds are more challenging as they are obtained from real scans and contain noise, outliers, missing shape, sparsity, and occlusions. Noise generally refers to points lying with a slight deviation from the overall shape of the point cloud. In contrast, outliers are points lying outside the average distribution by a considerable variation. The missing shapes are regions missing in the point cloud due to incomplete scans, presence of occlusions, etc.



(a) Outliers (b) Wall Points (c) Missing shape

(d) Connected Roof (e) Complex shape (f) Sparse and Noisy

Figure 1.5: Natural variations in roof instances in RoofN3D dataset

Figure 1.5 showcases some of these challenges in real point cloud scans found in the RoofN3D dataset [Wichmann *et al.* (2018)] - a large ALS dataset specific to three classes of roofs. In turn, these challenges demand new solutions for processing real point clouds as current methods have limited capabilities to automatically extract semantic information [Zhang *et al.* (2014), Zhang and Zhang (2018)]. Hence, we focus on introducing efficient deep learning methods that perform well on real point clouds like ALS roof point clouds and synthetic point clouds obtained from CAD models.

We choose three tasks that are commonly used to benchmark the performance of deep learning networks for point cloud processing and are extremely challenging and relevant to building modeling. This research first addresses whether it is possible to predict the roof type based on a given input roof point cloud. This is extremely useful to determine

the roof style which is used in various downstream tasks. Next, we attempt to solve the task of roof retrieval, i.e., given an input point cloud, retrieve the closest matching roof shape from a large database. This allows for usage of existing roof models from a database instead of noisy, incomplete point clouds acquired from a scan [Chen *et al.* (2017)]. As mentioned earlier, point clouds obtained from large-scale 3-D scans contain missing regions, which are not desirable. Therefore, the final research problem we attempt to address is the shape completion of partial point clouds using deep learning. As mentioned earlier, we aim to introduce efficient and highly performant methods in real and synthetic datasets. To summarise, in this research, we investigate the following questions.

Using Deep Learning:

1. is it possible to predict the roof style of ALS buildings?
2. is it possible to retrieve ALS building roofs of a particular roof-style efficiently?
3. given an incomplete ALS roof point cloud, is it possible to generate the complete shape of the roof efficiently?

Due to the success of deep learning in image analysis tasks, we start by verifying the effectiveness of view-based methods for shape classification and retrieval of building roof point clouds. Further, we also explore how information from multiple views can be efficiently aggregated. Recent research has shown the effectiveness of attention in various Natural Language Processing (NLP) tasks. Our experiments suggest that attention, being a set operation, is appropriate for point cloud processing, and we introduce a fully attention-based network for classification and retrieval. Further, we attempt to solve a more challenging problem of roof shape completion based on a

similar motivation. This research concludes that deep learning can indeed be utilized for ALS roof point cloud processing and shows improved performance and efficiency. Additionally, we also explore a new design space of utilizing fully attention-based models for point cloud understanding and analyze its effectiveness across various tasks.

1.2 PROBLEM STATEMENT

This thesis aims to explore deep learning-based approaches for deriving ALS roof point cloud representation. The goal is to investigate innovative deep learning architectures for various shape analysis tasks on ALS building roof point clouds.

1.2.1 Classification of ALS roof point clouds

Given a collection of labelled roof point clouds $(x_1, y_1), ...(x_n, y_n) \in X \times Y$ where x_i refers to a point cloud in the database and y_i its corresponding label. Classification is the process of predicting the label y for an input point cloud $x \in X$.

The classification task is implemented using two different deep learning approaches viz.:

1. a view-based method incorporated with attention techniques.
2. a point-based, fully attentional strategy.

View-based Classification of ALS Roof Point Clouds

We start our experiments by applying view-based methods for roof classification in which the point clouds are converted to multiple views and then classified based on the derived shape vector obtained from all the views. However, existing view-based

methods do not consider relationships between the views and cannot effectively utilize the view information for aggregation. In Chapter 3, a novel architecture called Multi-view CNN with Self Attention (MVCNN-SA) is used to determine the relevance of each view and combine them based on their relative importance. The view-based classification pipeline is shown in Figure 1.6.

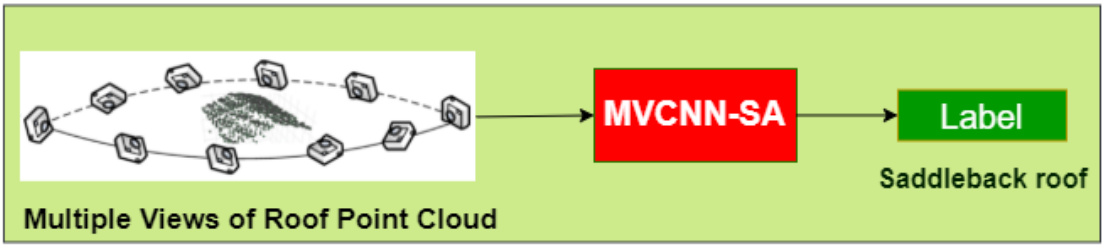


Figure 1.6: View-based classification of ALS roof point clouds

Point-based Classification of ALS Roof Point Clouds

Due to the increasing popularity of point-based methods in 3-D point cloud processing, a novel point-based method called Point Transformer is introduced for ALS roof classification in Chapter 4. In this work, we validate that attention is apt for point clouds as it is a set operator, i.e., cardinality and order invariant. Since a different design space is explored from existing point-based methods, the performance is evaluated on benchmark synthetic datasets. Additionally, we simulate various corruptions on these point clouds like sparsity, partial shape removal to evaluate the model's robustness. The point-based classification pipeline used in this work is shown in Figure 1.7.

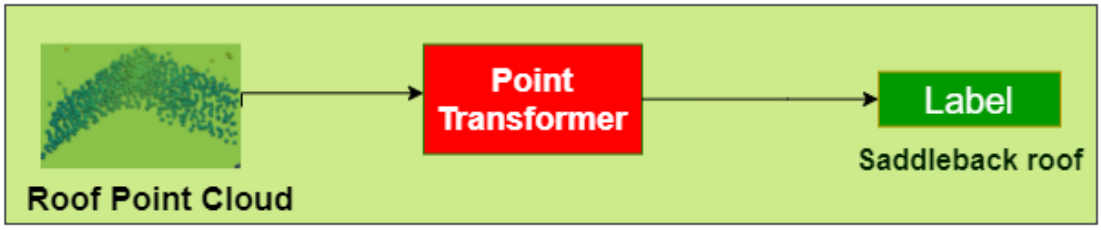


Figure 1.7: Point-based classification of ALS roof point clouds

1.2.2 Retrieval of ALS roof point clouds

Retrieval is the process of returning the closest matching roof point cloud x_i from a large database for an input query point cloud $x \in X$. The feature extracted from the query shape is compared with the features stored in the point cloud database, and the corresponding feature distances are calculated. The point clouds in the database are sorted based on the calculated feature distances and returned as output. Unlike classification, in retrieval, the model must capture fine details present in the point cloud, including intraclass geometric differences, to retrieve the closest shape. The pipeline for shape retrieval is shown in Figure 1.8. The feature representation derived by the proposed view-based and point-based methods for classification can be used for retrieval.

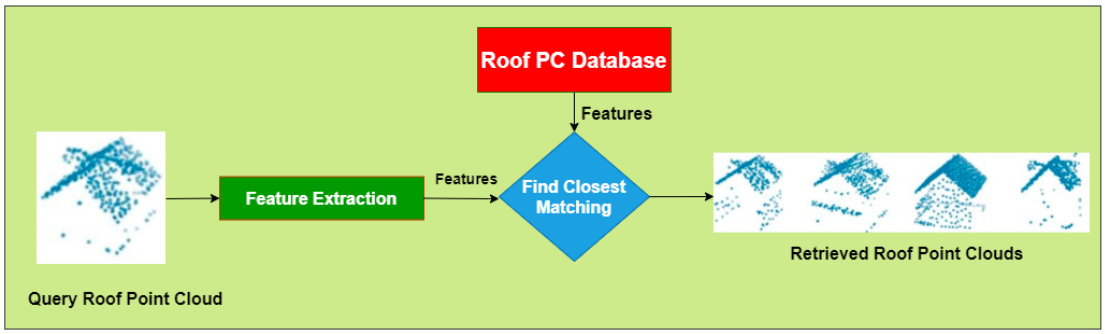


Figure 1.8: Retrieval of ALS roof point clouds

1.2.3 Shape completion of ALS roof point clouds

Shape completion is the estimation of the complete roof geometry from a damaged partial roof point cloud, or in other words, the goal of 3-D shape completion is to predict a complete shape Z for a given partial shape input point cloud X. The pipeline for shape completion is shown in Figure 1.9.

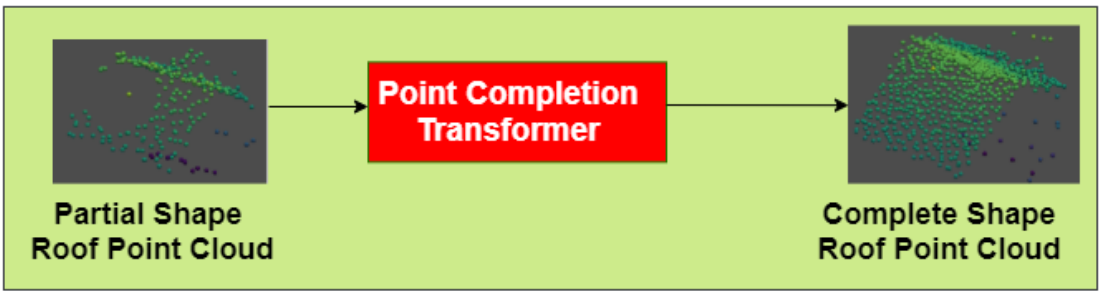


Figure 1.9: Shape Completion of ALS roof point clouds

Recent research has started focusing on 3-D shape completion with sufficient success. These methods emphasize learning the global, local geometric characteristics from the partial input shape to generate a complete point cloud. However, most existing deep networks cannot capture intricate details of the input partial point cloud as they tend to average multiple shapes. This generates genus-wise distortions, which leads to noise and loss of local geometric information. For an effective shape completion, the model must generate a complete point cloud coherent with the partial input shape without losing local geometric information.

We introduce a novel, fully attention-based deep learning approach, Point Completion Transformer (PCT), in Chapter 5 for complete shape reconstruction of ALS roof point clouds. The method includes a novel local attention operation, followed by encoder-

decoder connections to guide the completion process. Since we are exploring a different design space from existing point-based methods, the performance is also evaluated on benchmark synthetic datasets for 3-D point cloud shape completion.

1.3 MOTIVATION

With the recent advances in deep learning for various shape analysis tasks of point cloud such as classification, retrieval, and shape completion, the technology now seems mature enough to process 3-D point clouds of buildings acquired by an ALS. Current state-of-the-art geometric and machine learning methods perform these tasks on relatively simple 3-D building models from ALS point clouds, but they may not be suitable for all complex shape roof buildings in real-world environments [Wichmann (2018); Wang *et al.* (2018)]. This leads to a necessity for exploring these problems with new approaches like deep learning.

Deep learning methods are known to be highly efficient for feature extraction and have not yet been widely explored in the field of remote sensing, especially in urban modeling from ALS data due to the non-availability of large and suitable datasets [Wichmann *et al.* (2018)]. As deep learning approaches have a lot of properties such as very high generalization capability, scalability, ability to inherit robustness to various levels of point corruptions [Charles *et al.* (2017); Taghanaki *et al.* (2020)], they guarantee better performance in the real point clouds processing tasks.

1.3.1 Applications

The demand for ALS digital building models is increasing, and it has a significant role in various applications in GIS, remote sensing, photogrammetry, and computer vision. These building models can be used in many applications in urban and regional planning, surveying, generation of 3-D city models, city mapping, city navigation, wireless telecommunication and virtual information systems for tourists, environment simulations of pollutants, flood, and noise propagation. There are a wide variety of applications associated with 3-D classification, retrieval, and shape completion of ALS building roof point clouds which are described below:

Classification and Retrieval of Roofs

GIS provides accurate information for urban mapping, but details of building roof structures are often missing or incomplete in these maps. Identifying the roof geometry type and storage of those details in an easily accessible format for retrieval later can be used for many purposes. One use case is small unmanned aircraft systems, or drones can exploit this information to improve navigation accuracy and perform safe contingency landings, particularly in urban regions. Another use of roof classification is for estimating the solar potential by calculating the available surface area of the roof. Classification and retrieval of roof types play a significant role in determining the financial aid, insurance, and cadastral in some countries, as these are levied based on the roof type of the residing building. Roof-tops are often used as emergency rescue points or exits during natural calamities or incidents due to security threats. Therefore, identifying the roof style often helps in disaster relief operations. In GIS,

for generating a city-scale region, scene construction is the most time-consuming stage. Since many building models are already available publicly, it is feasible to reuse the existing resources instead of reconstructing each building. Hence, 3-D building model retrieval is highly relevant and useful for the scenarios mentioned above.

Shape Completion of Roofs

The automatic roof shape completion method can improve the quality of ALS roof point clouds and reduce manual labor for point cloud acquisition. Additionally, they are highly relevant for tasks including building energy modeling, reducing errors in registration, disaster simulation, real-time simulations for training and visualization, evaluation of the built environment, and improving the quality of existing building models. The quality of buildings reconstructed by existing automatic approaches is sufficient for most visualization purposes. However, these reconstructed models might not be accurate for critical analysis and simulations and need to be manually corrected. This process is usually very time-consuming, especially for recreating large cities, so efficient shape completion methods are required. These tasks have a significant role in archaeological buildings and heritage preservation in the event of partially scanned roofs due to occlusion or damage.

Change detection is another crucial task where shape completion has great relevance. The reconstruction step can be considered an essential prerequisite for 3-D change detection, particularly for detecting informal buildings or extensions and for the update of the 3-D map database. 3-D reconstruction helps detect the changes in the roof topology and structure details of buildings quickly, and this information can be used

to update urban map databases. The latest automatic shape completion methods help identify new, completely demolished, and unchanged buildings in a city. This also provides the state and local government officials to verify if the identified changes are authorized or not.

1.4 CONTRIBUTIONS

This thesis is perhaps the foremost research in applying deep learning-based methods for shape analysis tasks of ALS building roof point clouds. The research in this thesis contributes innovative deep learning architectures which show high performance and generalization capacity across various functions like classification, retrieval, and shape completion for both real point clouds like ALS roof scans and synthetic point clouds obtained from CAD models. The proposed methods surpass the state-of-the-art (SOTA) results in ALS roof point cloud classification, retrieval, and shape completion tasks.

The main contributions of the presented research can be summarized as follows:

1. Successfully shown that deep learning methods can be used for the effective representation of ALS roof point clouds.
2. Introduced novel view pooling operations in a view-based method for aggregating multiple views of a point cloud based on the relevance of the views.
3. We validate that attention is a more appropriate operation for point cloud processing and introduce a fully attentional network for shape classification and retrieval.
4. Further, we attempt to solve a more challenging problem - shape completion using a fully attentional model which can encode better local information.
5. Unlike other works, we also focus on analyzing the robustness of our proposed methods to unseen point corruptions on the input point cloud.
6. The proposed methods are computationally efficient, making them suitable for real-time applications.

1.5 ORGANIZATION OF THE THESIS

This thesis is organized into six chapters. After this introductory chapter 1, it is structured as follows: Chapter 2 presents an extensive literature survey of works related to ALS building roof classification, retrieval, and shape completion. It provides an overview of different approaches and techniques concerned with the automatic classification, retrieval, and reconstruction of 3-D building models from ALS data. Further, a novel view-based approach for attention-based classification and retrieval is discussed in Chapter 3. In Chapter 4, an innovative deep learning method using full attention for classification and retrieval tasks is described. Chapter 5 discusses another fully attention-based architecture explicitly designed for reconstruction. Chapter 6 concludes the thesis along with directions towards future work.

CHAPTER 2

LITERATURE SURVEY

The knowledge of the roof type and its geometry is very significant for various applications in urban modeling, indicating that shape analysis tasks such as classification, retrieval, and shape completion of ALS roof point clouds are relevant, well-studied research subjects. These techniques broadly fall into the categories of geometric or classical machine learning methods and are limited by their capacity to work on complex roof shapes and noisy scans. Deep learning has proven to be very successful across various domains. Still, to the best of my knowledge, no work explores the applications of the same across multiple tasks utilizing ALS roof point clouds. This section is split into two portions - Section. 2.1 describes existing works specific to ALS roof modeling while Section. 2.2 discusses existing literature for deep learning to process general 3-D point clouds.

2.1 ALS BUILDING ROOF MODELING TECHNIQUES

One of the primary tasks in ALS roof modeling is reconstruction using which other sub-tasks like classification, segmentation, structuring, hypothesis generation can be inferred [Awrangjeb *et al.* (2018)]. The existing strategies for ALS building roof reconstruction are classified into two main categories: model-driven and data-driven [Vosselman and Maas (2010)]. Model-driven or top-down (parametric) approaches require a predefined catalog (grammar-based) of basic roof shapes (e.g., flat, gable, and hip), and the most appropriate model is then selected from this available library to best

fit the input point cloud [Henn *et al.* (2013); Vanegas *et al.* (2012); Lafarge and Mallet (2011)]. The advantage of such model-driven methods is that the final roof shape is always topologically correct and relatively robust to noise, missing data, sparsity since we are directly reusing predefined roof models [Dorninger and Pfeifer (2008); Gkeli and Ioannidis (2018)]. However, these methods require prior information about the roof style and can only reconstruct simple roof shapes (i.e., useful for tasks that do not require a high level of detail (LoD)) as these complex or arbitrary roof shapes are not included in the catalog [Haala and Kada (2010)]. One possible solution is to combine the simple building templates in the catalog and reconstruct complex roof shapes that are not defined in the template, thereby avoiding the need to expand the catalog with additional models [Wichmann (2018)].

In data-driven or bottom-up (non-parametric) approaches, simple geometric primitives such as points, lines etc are aggregated together to form the roof planes [Verma *et al.* (2006); Sampath and Shan (2010)]. These data-driven methods are categorised based on the adopted techniques [Gkeli and Ioannidis (2018)] such as segmentation [Rottensteiner *et al.*; Kada and Wichmann (2012); Tarsha-Kurdi *et al.* (2008)], plane fitting [Fischler and Bolles (1981); Omidalizarandi and Saadatseresht (2013)], filtering and thresholding [Alexa *et al.* (2003)], and other learning-based methods [Makantasis *et al.* (2015); Alidoost and Arefi (2016)]. A popular data-driven approach to reconstruct the roofs is to reassemble the planar patches derived by segmentation algorithms. The most common segmentation algorithms used for segmenting building roof planes are region growing [Rottensteiner *et al.*; Kada and Wichmann (2012)] and RANSAC [Tarsha-Kurdi *et al.* (2008)]. After segmentation, the geometric and

topological relationships of the resulting segmented planes [Rottensteiner *et al.*; Kada and Wichmann (2012); Sohn *et al.* (2008)] are used to extract building shape cues such as intersection, step lines, outlines, and other surface primitives. Finally, roof models are reconstructed based on the extracted building modeling cues and refined using several regularisation and optimization operations [Zhou and Neumann (2010); Jung *et al.* (2017)].

Existing building reconstruction methods can reconstruct realistic 3-D roof models from ALS point clouds if the input point clouds are of high quality with sufficient density, noise-free, and complete [Awrangjeb *et al.* (2018); Jung *et al.* (2017); Wu *et al.* (2017)]. If the input data is of low quality, the roof superstructures such as the chimney, dormers are usually considered noise and ignored during the reconstruction process, thereby reconstructing only coarse and basic roof models. Though several strategies have been proposed to solve this problem, most of them are not applicable in practice for a fully automatic large-scale reconstruction [Wichmann (2018)]. Therefore, there is a vital requirement for methods that can deal explicitly with the automatic reconstruction of small roof structures and have the ability to deal with low-quality roof point clouds. A class of data-driven methods called learning-based methods are specifically developed to generalize better to diverse and complex input data, but their application in ALS roof building modeling has not yet been fully explored.

2.1.1 Classification of ALS building roof point clouds

Existing research works for ALS roof style classification primarily rely on a set of pre-defined rules to identify certain roof styles. Hence, they suffer from low classification

performance due to the use of heuristic rules and the assumptions regarding the geometry of these roofs [Zhang *et al.* (2014)]. While learning-based methods are meant to generalize better to a wide variety of roof types, there exist very limited works in ALS roof classification [Zhang *et al.* (2014); Castagno and Atkins (2018); Guptha and Bohare (2019)]. Zhang et al. [Zhang *et al.* (2014)] use a random forest classifier to train a bag of words feature extractor from the input point cloud. Then, a synthetic codebook is generated manually instead of learning from the sample data, and the proposed method achieves better classification performance in the given datasets. Castagno et al. [Castagno and Atkins (2018)] uses a multi-modal architecture utilising both satellite images and Light Detection and Ranging (LiDAR) data as input. Pre-trained CNNs extract features from the input satellite images and projected LiDAR images passed to support vector machines (SVM) or random forest classifiers to predict the roof shape. However, these classical machine learning algorithms like SVM, logistic regression, and decision trees suffer from computational complexities due to the high-dimensionality of GIS data.

Deep Learning is another type of learning-based method which has shown great efficiency in various domains. Sarthak et al. [Guptha and Bohare (2019)] have attempted to perform roof classification using a point-based deep learning approach called PointNet. While their work directly extends an existing architecture for roof classification, such methods can perform the classification task with great efficiency and accuracy. Therefore, there is a scope for utilizing more advanced techniques in Deep Learning for ALS roof classification, which can directly lead to better performance.

2.1.2 Retrieval of ALS building roof point clouds

The retrieval of 3-D building roof models from model databases or the internet is a need of the hour for efficient urban scene reconstruction and real-world tasks [Chen *et al.* (2017, 2014*b*)]. This emphasizes the concept of data reuse and reduces the overall reconstruction cost, i.e., available roof models in a database having similar geometric shapes are reused rather than reconstructing ALS roof point clouds whenever they are acquired. Existing methods for roof model retrieval primarily use input queries as polygon models [Akgul *et al.* (2009)] while a few recent works have attempted to use point clouds [Chen *et al.* (2017, 2014*b*)].

The core idea in retrieval is to build a compact, efficient encoding of the building roof model, which can be compared to retrieve the closest matching shape. Chen et al. [Chen *et al.* (2017)] have proposed a view-based method for 3-D building model retrieval using ALS point clouds. However, view-based methods yield reasonable retrieval results as they do not capture strong local features and are affected by self-occlusion [Chen and Lin (2016)]. Another work [Chen *et al.* (2014*b*)], proposes to encode the input point cloud using low-frequency spherical harmonic functions (SHs) [Chen *et al.* (2014*b*)]. However, this decreases the ability to distinguish objects with similar geometric shapes, leading to ambiguity in shape description. Therefore, we investigate new strategies for developing more efficient methods for a consistent and accurate representation of building roof point clouds. This can be done by proposing deep learning methods which can encode better local geometric information and using the derived feature representation for fully automatic retrieval.

2.1.3 Shape Completion of ALS building roof point clouds

Shape completion is the process of predicting a complete roof shape given an input partial point cloud. ALS scans contain multiple imperfections, and the scanned point clouds can have missing regions due to incomplete scans, occlusions, etc. A hybrid approach for reconstruction of ALS roof point clouds involving deep learning and geometric methods has been developed to reconstruct lightweight building models with a level of detail 2 (LOD2) [Zhang and Zhang (2018)]. In this approach, the ALS building point clouds are first classified by deep reinforcement learning and then reconstructed by geometric methods. The reconstruction approach integrates the edge-aware resampling algorithm and 2.5-D dual contouring for building reconstruction. Although this approach is capable of generalizing and reconstruct complex roof shapes precisely, semantic information is lost. Additionally, this method is not effective for large-scale datasets.

To the best of our knowledge, there exists no published work with results for ALS roof shape completion using deep learning methods. Sarthak et al. [Guptha and Bohare (2019)] have made an attempt to try shape completion of ALS roof point clouds using a point-based deep learning approach called Point Completion Network (PCN). This work has shown that shape completion can be performed on ALS roof point clouds successfully, revealing that more advanced deep learning techniques can perform better.

2.2 DEEP LEARNING FOR GENERAL 3-D POINT CLOUDS

Recent progress in deep learning and the emergence of 3-D shape datasets such as ModelNet [Wu *et al.* (2015)], ShapeNet [Chang *et al.* (2015)] have led to advancements in 3-D point cloud representation. Many deep learning approaches for effective feature representation are used for various 3-D point clouds processing tasks such as classification, retrieval, and segmentation. These approaches can be roughly categorised into voxel-based [Qi *et al.* (2016); Maturana and Scherer (2015)], view-based [Su *et al.* (2015); Feng *et al.* (2018); He *et al.* (2018); Kanezaki (2016)] and point-based methods [Charles *et al.* (2017); Qi *et al.* (2017); Li *et al.* (2018*b*); Liu *et al.* (2019*c*,*a*); Wang *et al.* (2019)].

Voxel-based methods convert 3-D points into volumes which leads to unnecessary artifacts, cannot model complex shapes effectively, and require huge compute resources for high-resolution voxelization. View-based methods project the input point cloud into multiple views, and existing deep learning-based networks extract view-feature vectors which are then used to create the final shape descriptor. These methods perform better than voxel-based methods but require large compute storage requirements and are affected by self-occlusions, number of viewpoints. On the other hand, point-based methods directly apply deep learning methods on raw xyz information, which leads to no loss in information. The following subsections discuss deep learning methods to process general 3-D point clouds for various tasks like classification, retrieval, and shape completion.

2.2.1 Classification of 3-D point clouds

View-based methods [Su *et al.* (2015); Feng *et al.* (2018)] utilise a dense convolutional neural network to derive view-feature vectors from multiple views of the given point cloud. Due to the continuous progress in CNNs, these methods showcase high performance when compared to the methods which directly utilize 3-D coordinates [Su *et al.* (2018); Jiang *et al.* (2019); Kanezaki (2016)]. However, existing view-based methods do not focus on the effective aggregation of view features to generate an overall shape representation. Multi-view Convolutional Neural network (MVCNN) [Su *et al.* (2015)] treats all views to be equally important and performs a naive max-pooling operation which can lead to loss of information. Therefore, we need to look at better strategies to account for view-wise relationships and combine these representations into a rich shape descriptor.

While point-based methods are more challenging and under-perform when compared to view-based methods, recent research has been continuously exploring new ideas to reduce the performance gap between point-based and view-based techniques. Point clouds are naturally non-uniform, unstructured, unordered, making it difficult to directly apply deep learning networks while maintaining order-invariance. A pioneering point-based method, PointNet [Charles *et al.* (2017)] addresses this problem by utilising symmetric operations to derive a feature representation for point clouds. PointNet consists of a stack of multilayer perceptron (MLP) layers followed by a max-pooling operation to derive a global feature representation. Since PointNet does not derive any local information, Qi et al. [Qi *et al.* (2017)] proposed an improved version PointNet++ to capture local and global information. PointNet++ is a hierarchical network that

applies PointNet recursively on a nested portion of the point cloud and extracts local features at different scales. PointNet preserves the order invariance property more strictly when compared to PointNet++. Recent research, including SO-Net, PointCNN, DensePoint, and graph-based DGCNN, has focused specifically on deriving better local information. SO-Net [Li *et al.* (2018*a*)] uses an improved hierarchical learning of local features which showcases improved performance when compared to Point Net++, PointCNN [Li *et al.* (2018*b*)] uses a χ-transformation matrix to get an order-independent feature and simulate convolution-like operation on the input point cloud. DensePoint [Liu *et al.* (2019*c*)] learns densely contextual representation by utilising multi-scale and multi-level information from the input point cloud and showcases strong performance gains. Graph-based networks like DGCNN [Wang *et al.* (2019)] construct a graph in the feature space, which is updated after every layer based on neighborhood group of points. Although these methods yield high classification accuracies compared to PointNet, they are drastically affected by unseen point corruptions [Taghanaki *et al.* (2020)].

The attention mechanism was first introduced in NLP to allow models to focus on the important parts of the input text and build suitable context. They have been shown to improve performance when used in complementary with existing architectures. Similarly, for point cloud processing, there have been attempts to use attention to capture better feature representations and obtain high performance. L2G [Liu *et al.* (2019*b*)] proposes an auto-encoder for learning local information from point clouds similar to PointNet++ with self-attention. Point2Sequence [Liu *et al.* (2019*a*)] captures fine-grained contextual information and aggregates multi-scale features from each local

region using attention. Attentional ShapeContextNet (ASCN) [Xie *et al.* (2018)] uses a stack of sequential attention blocks to derive point features and is then max-pooled to create a global feature representation which results in loss of information. Set Transformer [Lee *et al.* (2019)] adopts a similar strategy like ASCN but introduces an attention-based pooling operation to overcome the loss of information. However, these two methods are excessively parameterized, leading to the sparsity of attention matrices and over-fitting. Point Attention Transformer [Yang *et al.* (2019)] introduces a rich point-wise embedding, followed by a Group Shuffle Attention (GSA) to capture relations between points. It also uses a learnable Gumbel Subset Sampling (GSS) for down-sampling, which, unlike farthest point sampling, is permutation invariant, differentiable, and trainable for end-to-end learning of hierarchical features. These above-mentioned methods (except L2G and Point2Sequence) try to utilize transformer blocks but under-perform when compared to other SOTA methods. Therefore, we investigate better ideas to construct an efficient and effective method for shape classification while still being robust to unseen transformations.

2.2.2 Retrieval of 3-D point clouds

The accurate retrieval of 3-D point clouds is challenging as there exist diverse shapes with complex local geometries. Deep learning has been widely used for 3-D shape retrieval in recent years. Generally, these approaches can be roughly categorized into three classes: model-based, view-based, and metric-based approaches. The model-based approaches directly learn shape features from 3-D data formats, such as polygon meshes or surfaces [Boscaini *et al.* (2016); Xie *et al.* (2017)], voxel grid [Maturana

and Scherer (2015)] and point clouds [Charles *et al.* (2017); Qi *et al.* (2017)]. The main limitations of these methods lie in the restriction of shape representation or high computational complexity, especially for the voxel-based methods. PointNet learns to capture global information, which leads to lower retrieval performance. PointNet++, on the other hand, derives stronger local information and hence offers a high retrieval score compared to PointNet. More improved works such as PointCNN [Li *et al.* (2018*b*)], DGCNN [Wang *et al.* (2019)], DensePoint [Liu *et al.* (2019*c*)] introduce specific modifications to capture local information and hence exploit the same for effective shape retrieval. Since existing attention-based networks learn to capture global information due to the inherent property of the attention operation, its retrieval results are only comparable to that of PointNet. Similarly, features extracted from multi-view methods [Su *et al.* (2015); Feng *et al.* (2018); He *et al.* (2018)] can be used for retrieval. The amount of information captured in these view-based methods is directly related to the number of views, absence of any occlusions, etc. Unlike classification, view-based methods do not perform as well as point-based methods in the shape retrieval task as direct access to the raw point cloud helps derive better geometric details.

2.2.3 Shape Completion of 3-D point clouds

Shape completion is a challenging problem and has a lot of applications in computer graphics [Anguelov *et al.* (2005); Yuan *et al.* (2018)], computer vision [Dai *et al.* (2018); Hou *et al.* (2019)] and robotics [Sung *et al.* (2015); Engel *et al.* (2014)]. Recently, deep learning methods [Xie *et al.* (2020); Yuan *et al.* (2018); Zhang *et al.* (2020); Tchapmi *et al.* (2019); Yang *et al.* (2018)] have become popular for 3-D

shape completion with the aid of synthetic datasets like Completion3D [Tchapmi *et al.* (2019)], ShapeNet part dataset [Chang *et al.* (2015)].

Most of the pioneering learning-based methods for shape completion are based on volumetric grids [Dai *et al.* (2017); Han *et al.* (2017); Stutz and Geiger (2020)] or view-based projections [Park *et al.* (2019)] that leverage 3-D/2-D convolution operations resulting in high computational cost or loss of geometric information. To avoid these limitations, Yuan et al. proposed PCN [Yuan *et al.* (2018)], which directly maps the partial input point cloud to a complete shape. PCN is the first encoder-decoder-based deep learning architecture explicitly designed for 3-D shape completion. PCN uses a PointNet based encoder and a decoder with stacks of fully connected layers with certain properties from folding networks. These fully connected layers capture the global shape while the folding network captures the local information. However, the model fails to reconstruct objects consisting of multiple disconnected parts and very thin structures such as wires. TopNet [Tchapmi *et al.* (2019)] is another encoder-decoder-based architecture that explores a hierarchical tree-based decoder where every node is responsible for generating separate portions of the point cloud. A constraint of this decoder is that the tree needs to be sufficiently large, which is difficult to handle, and if smaller size, it can limit the possible topologies learned. GRNet [Xie *et al.* (2020)] is another interesting work that converts the 3-D points into grids and proposes a unique feature sampling layer to extract features from neighboring points. Additionally, they also introduced a differentiable "Gridding Loss" to ensures better penalization based on the model's ability to reconstruct local geometries. It is important to note that most of the existing works can only reconstruct an overall global shape and causes loss of local

geometric information. For example: Given four distinctly unique chairs in the dataset, autoencoders tend to "average" these shapes and produce a common structure that can minimize loss against all the samples. To overcome this problem, Point Fractal Network (PFN) [Huang *et al.* (2020)] predicts only the missing shape and fuses its output with the partial input to create the complete point cloud. PFNet uses an adversarial network where the generator consists of a stack of PointNet++ blocks to encode the partial input shape. However, the distribution of points in the predicted and partial shapes need not be similar and thus can lead to distortions. Hence, we need to propose an improved strategy to capture strong local geometric cues from the partial input shape and generate a complete point cloud coherent to the input shape.

CHAPTER 3

CLASSIFICATION AND RETRIEVAL OF ALS ROOF POINT CLOUDS USING MVCNN WITH SELF-ATTENTION

3.1 INTRODUCTION

Classification and retrieval of ALS roof point clouds are useful for various applications, such as urban modeling. Performing these tasks with high accuracy is a challenge due to the natural variations present in the data, such as irregularity, sparsity, noise, occlusion, and outliers. Deep Learning-based methods have proven to have high generalization capacity and can extract discriminative features from the input point cloud. Most existing works primarily focus on deriving shape representations for clean, aligned point clouds. Hence there is a strong requirement for methods that are robust to various input imperfections.

To convert these point clouds into a regular format, they are transformed into 3-D voxels or a collection of images (typically multiple views). Due to ever-growing research in image recognition, view-based methods still outperform other techniques in point cloud processing. Multi-view methods project a 3-D point cloud into multiple views, then passed on to a Deep Neural Network to extract view-wise features. The view-wise features are then aggregated to generate a concise shape descriptor for various downstream tasks like classification, retrieval, etc. Multi-view convolutional neural network (MVCNN) [Su *et al.* (2015)], Group-view convolutional neural network (GVCNN) [Feng *et al.* (2018)] are popular multi-view methods which give high performance scores and achieve state-of-the-art performance. The efficiency of these

methods is not yet evaluated on ALS roof point cloud scans and other similar real datasets. The emergence of various deep convolutional neural networks like ResNet [He *et al.* (2016)], ImageNet [Deng *et al.* (2009)], VGG [Simonyan and Zisserman (2015)] has enabled efficient extraction of features from each view image. However, all these views do not contribute equally towards representing the given shape, and naive pooling techniques can lead to the loss of information.

This work shows how learning the relative importance of view-descriptors results in a better overall shape representation, leading to an increase in performance across all tasks. Similar approaches have been used for the identification of salient features from multiple instance images for object detection as in [Han *et al.* (2015), Zhang *et al.* (2016), Zhang *et al.* (2017), Cheng *et al.* (2016)]. The proposed method - Multi-view CNN with Self-Attention (MVCNN-SA) incorporates an attention block to derive relative importance weights for each view, which are then used to create a pooled representation of the object. Additionally, this leads to robustness to varying view-point changes.

The key contributions of the work are as follows:

- First work to study the effectiveness of deep learning methods for ALS roof point cloud processing.
- Introduced an adaptive view pooling method based on the relative importance of each view to fuse information efficiently.

3.2 PRELIMINARIES

3.2.1 Multi-view CNN

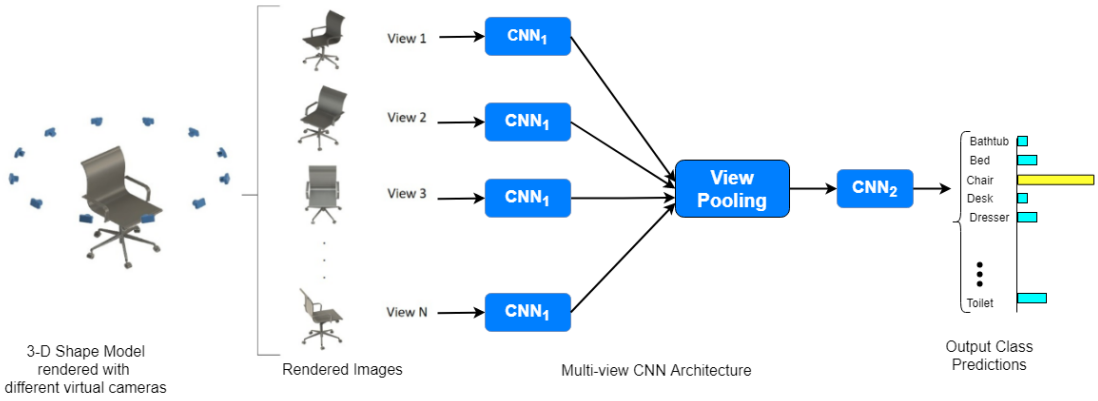


Figure 3.1: Architecture of Multi-view CNN [Su *et al.* (2015)]

Multi-view convolutional neural network (MVCNN) [Su *et al.* (2015)] shown in Figure 3.1, consists of four main components - virtual cameras to render multiple view images of a given point cloud, CNN_1 to derive view wise representations, view-pooling block to generate a single shape descriptor, CNN_2 to enhance the shape representation before classification. Subsection. 3.2.2 discusses in detail the procedure to capture multiple views. CNN_1 uses a standard convolutional backbone VGG, pretrained on the ImageNet dataset, and extracts the feature vectors from the penultimate layer. The network parameters of CNN_1 are shared across all the views, which is followed by an element-wise max-pooling operation in the view-pooling block. CNN_2 consists of a stack of fully connected layers followed by a softmax-classification layer. The authors from the original paper evaluate the model on the ModelNet40 dataset and outperform existing voxel-based methods like 3-D ShapeNet by almost 12% in classification and 21% in shape retrieval. However, the method considers all views to be equally

important, which can affect the quality of the shape descriptor to a large extent. Under such circumstances, it is essential to further investigate the relationships between the views adaptively.

3.2.2 Rendering of Views in MVCNN

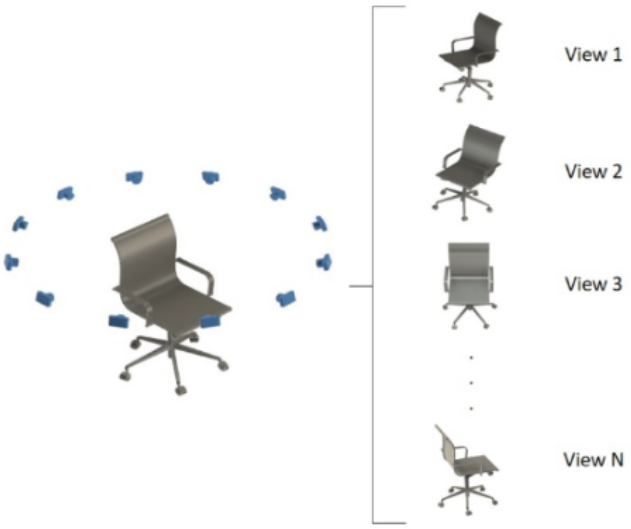


Figure 3.2: Capturing of views (illustrated using the first camera setup)

The 3-D models are publicly available as polygon meshes, which are collections of vertices, edges, and faces that define the shape of a polyhedral object. MVCNN uses the Phong reflection model [Phong (1975)] to generate the rendered views of polygon meshes. The mesh polygons are rendered under a perspective projection, and the pixel color is determined by interpolating the reflected intensity of the polygon vertices. Shapes are uniformly scaled to fit into the viewing volume. MVCNN experiments with two camera setups for capturing the views of a 3-D polygon mesh. Viewpoints (virtual cameras) must be set up for rendering each mesh to create a multi-view shape

representation. For the first camera setup, the input shapes are assumed to be upright oriented along a consistent axis. Most available 3-D models satisfy this requirement. In the first experiment, 12 rendered views are created by placing 12 virtual cameras around the mesh every 30 degrees, as shown in Figure 3.2. The cameras are elevated 30 degrees from the ground plane, pointing towards the centroid of the mesh. The centroid is calculated as the weighted average of the mesh face centers, where the weights are the face areas. For the second camera setup, no assumption is made about the consistent upright orientation of shapes. In this case, views are rendered from several more viewpoints since it is unknown beforehand which ones yield good representative views of the object. The renderings are generated by placing 20 virtual cameras at the 20 vertices of an icosahedron enclosing the shape. All cameras point towards the centroid of the mesh. Then four rendered views are generated from each camera, using 0, 90, 180, 270 degrees rotation along the axis passing through the camera and the object centroid, yielding 80 views.

3.2.3 Retrieval in MVCNN

In MVCNN, the shape descriptor derived as the output of the max pool function is directly used as the feature vector for 3-D shape retrieval. For two 3-D shapes, X and Y, x and y are the shape descriptors extracted by MVCNN. Then Euclidean distance between these two 3-D shapes is used in shape retrieval. The distance metric formula is defined as: $d(X, Y) = ||x - y||_2$, where $||.||_2$ represents the l_2 distance norm between x and y. Since MVCNN is fine-tuned for classification, retrieval performance is not directly optimized, and thus MVCNN further employs a low-rank Mahalanobis metric

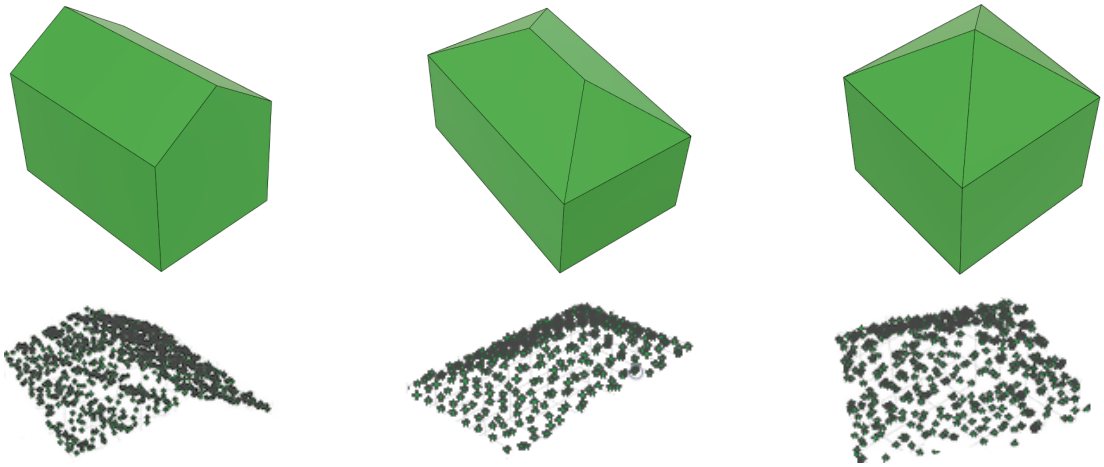
to improve the retrieval performance. In the proposed work, we do not consider such optimization methods to directly gauge the model's capacity to learn efficient shape representations.

3.2.4 Dataset

The main challenge in the direct application of deep learning for ALS building roof classification was the absence of a publicly available dataset that provides distinct classes for buildings. However, many urban applications such as facility management and preservation of townscapes require a fine subdivision of the building class, for example, to distinguish between different roof types or to recognize specific roof structures. In order to close this crucial gap, Wichmann et al. [Wichmann *et al.* (2018)] have recently developed a publicly available dataset- RoofN3D, which can be used to train deep neural networks for different tasks in the context of 3-D building reconstruction. This dataset covers a large area in New York City (NYC) and includes rich semantic information. Provided by the U.S. Geological Survey (USGS), these point clouds were collected from airborne LiDAR from 08/2013 to 04/2014 and cover an area of 1,009.66 sq. km. The average density of the point clouds is about 4.72 points/m^2. The spatial reference of the point cloud is NAD83(2011)/UTM zone 18N with the unit in meters.

RoofN3D consists of ALS 3-D point clouds corresponding to the roofs of buildings of New York City and has 118,073 instances with three building types, i.e., pyramid, saddleback (gable), and two-sided hip. Each instance in the dataset contains the point cloud of a single roof. Many building point clouds contain outliers such as points of

trees, walls, or projections. Some point clouds also have missing points, making this a representative real-world dataset. One limitation of this dataset is that it contains only three categories of roofs. RoofN3D is still a work in progress and is expected to be improved, updated, and extended in the future with more roof categories. The training data is available at https://roofn3d.gis.tu-berlin.de. The individual building types and their corresponding roof point clouds are shown in Figure 3.3.



(a) Saddleback (b) Hip (c) Pyramid

Figure 3.3: Building roof types and corresponding roof point clouds in RoofN3D

3.2.5 Initial Experiments

Based on our experiments, we have partitioned the dataset as 60% for training, 25% for validation, and the remaining 15% for testing for the proposed work. Since RoofN3D is an imbalanced dataset, the proportion between the classes was maintained in all these subsets via stratified sampling. Stratified sampling aims at splitting a data set so that each split is similar with respect to something. In a classification setting, it is often chosen to ensure that the train and test sets have approximately the same percentage of

samples of each target class as the complete set. The exact details of the split in the RoofN3D dataset are mentioned in Table 3.1.

Table 3.1: Class wise distribution of RoofN3D dataset

Class	Instances	Train	Validation	Test
Saddleback	89057	53434	22264	13359
Two-sided Hip	26830	16098	6707	4025
Pyramid	2186	1311	547	328
Total	118073	70843	29518	17712

Due to the challenges present in the RoofN3D dataset discussed in subsection 3.2.4, we first evaluate the feasibility of applying deep learning by experimenting with a standard convolutional network - ResNet. We build 20 multi-view images for each point cloud in the RoofN3D dataset and train the model to perform single view classification. The results obtained (shown in Figure 3.9) clearly indicate the feasibility of such an approach. This can be further improved by deriving a shape descriptor from all the views of a given point cloud, as discussed in this chapter.

3.3 METHODOLOGY

The pipeline for ALS roof shape classification as shown in Figure 3.4 has two main phases: rendering of views and classification of views. Section. 3.3.1 discusses the procedure used for view-rendering, and the network architecture for classification is described in section. 3.3.2. The overall architecture can be broken down into the following major components:

1. Multiple views of a given point cloud are rendered and fed to a feature extraction network in MVCNN-SA to generate view-wise feature vectors.
2. These view-wise representations are then passed through a self-attention network to capture relative feature importance and create a pooled shape descriptor.
3. The shape representation is then passed through a classification layer to predict the roof shape.

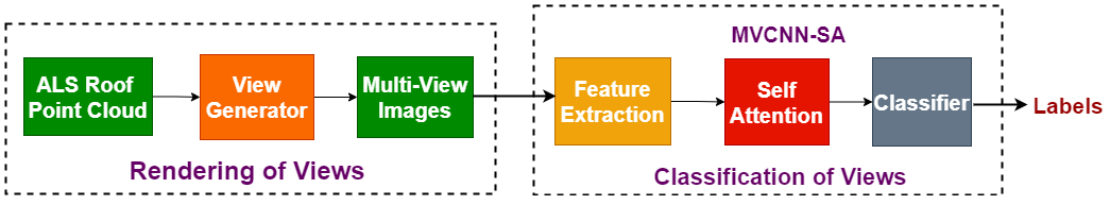


Figure 3.4: Pipeline of View-based Classification with two phases: Rendering of views and Classification

3.3.1 Rendering of Views

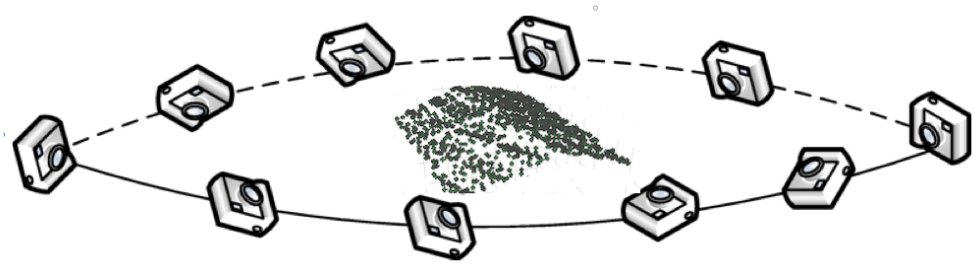


Figure 3.5: Rendering of views from ALS roof point cloud

The technique used for capturing 2-D views of ALS roof point clouds is similar to the 1^{st} camera setup, as in section 3.1 in [Su *et al.* (2015)]. In this case, the roof point clouds are assumed to be in an upright orientation along a specific axis (e.g., the z-axis). The view-generator takes 2-D projections from different angles to capture the geometry of the object. The virtual cameras are placed around the point cloud mesh

so that the camera viewpoints are separated by intervals of angle θ in the same plane as shown in Figure 3.5. The elevation of each camera above the horizontal plane is $30°$, pointing towards the centroid of the point cloud. The centroid is calculated as the weighted average of the mesh face centers, where the weights are the face areas. Phong shading [Phong (1975)] in Blender [Blender Online Community] was used to generate the rendered views of point clouds and the images are captured with depth information. The whole setup, with lighting, is similar to the approach used in [Su *et al.* (2015)]. We set $\theta = 36°$ which yields 10 views for an object. Additionally, to test for robustness, we randomly change the camera angles at each viewpoint and evaluate the classification performance in Section 3.4.

3.3.2 Classification

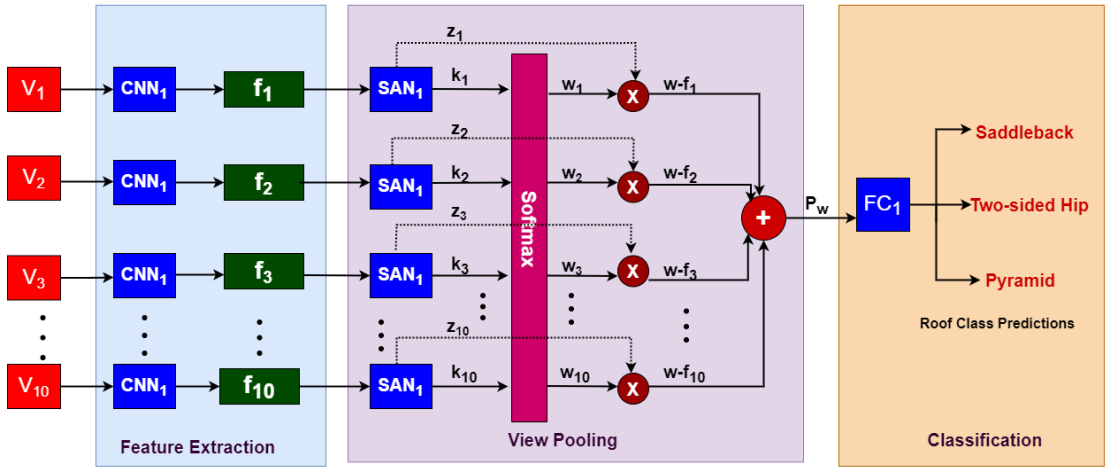


Figure 3.6: Architecture of MVCNN-SA comprising three parts: Feature Extraction, View-Pooling and Classification.

The architecture of the proposed MVCNN-SA is shown in Figure 3.6. Similar to the architecture of MVCNN, MVCNN-SA also consists of three main parts. The first part

is used for feature extraction and consists of CNN blocks, where all branches of the network share the same parameters in CNN_1. The second part is for view-pooling and has the self-attention network (SAN) blocks in which all SANs share the same weights. This is followed by the third part, which has only a fully connected layer FC_1 for classification.

For each 3-D roof point cloud, let $V = \{v_1, v_2, ..., v_{10}\}$ be the set of rendered images where each v_i is passed to CNN_1. The feature set $F_M = \{f_1, f_2, ..., f_{10}\}$ consists of feature maps f_i extracted from these views. In our experiments, we have used a ResNet-18 architecture for CNN_1. The feature maps f_i are passed to an SAN which generates a key-value pair (k_i,z_i) for a particular view i as shown in Eqn. (3.1) and (3.2).

$$z_i = Z(f_i) \, \epsilon \, R^L \tag{3.1}$$

where z_i is the output of the value network and L is the size of the value vector.

$$k_i = K(f_i) \, \epsilon \, R^L \tag{3.2}$$

where k_i is the output of the key network.

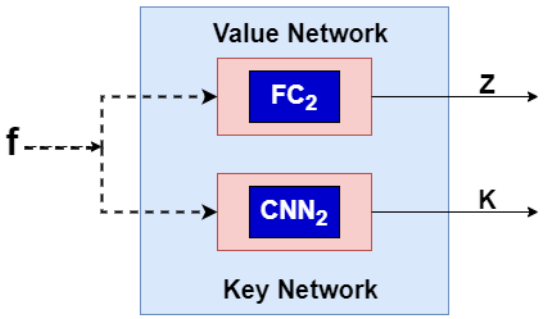


Figure 3.7: Architecture of Self-Attention-Network (SAN).

The architecture of the SAN is shown in Figure 3.7 and consists of two parts - a value network Z containing a fully connected layer FC_2 and a key network K containing a CNN_2 network. CNN_2 consists of three convolutional layers followed by a fully connected layer whose outputs represent un-normalized self-attention weights. These are called keys and are passed through a softmax function for normalization, producing w_i as shown in Eqn. (3.3).

$$w_{ij} = \frac{e^{k_{ij}}}{\Sigma_i \, e^{k_{ij}}} \tag{3.3}$$

where w_{ij} is the attention matrix given by the softmax function. The purpose of the softmax function is to ensure that the weights for a particular feature of the value vector across views sum to one. These weights are multiplied with the outputs from the value network to get the scaled view features which are then aggregated together to get the view pool as shown in Eqn. (3.4).

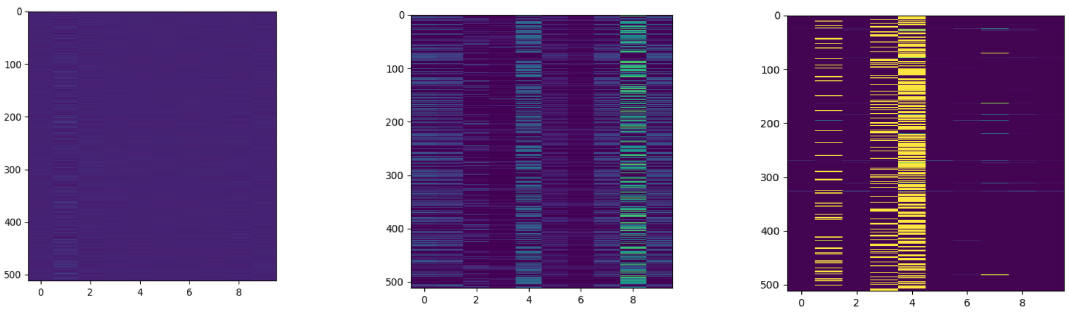
$$P_w = \Sigma_i \, w_i \odot z_i \, \epsilon \, R^L \tag{3.4}$$

where P_w is the aggregated view pool. This view pool is then fed to a fully connected layer with softmax activation for classification. We term this view-pooling method as feature-wise attention. The entire network is trained in an end-to-end fashion using the cross-entropy loss function. In feature-wise attention, the weights $w_i \in R^L$ are vectors such that the j^{th} element of each vector sum to one. The scaling of each z_i is performed using a Hadamard product, as shown in Eqn. (3.4). It assigns different importance

to each feature in a view vector, allowing the network to pay attention to the salient features in every view. An auxiliary entropy term is included in the loss function to study the effect of changing attention map distributions. The entropy is calculated as in Eqn. (3.5)

$$S = -\Sigma_i \, w_i \cdot log w_i \tag{3.5}$$

A positive entropy penalty encourages the attention map to be concentrated, while a negative penalty causes the attention map to become more diffuse. The attention matrix is analyzed, and three possible configurations are recognized in it. A diffused attention map is one in which all views contribute almost equally to the prediction and is shown in Figure 3.8a. The intermediate attention map is one in which some views contribute more to the prediction and is plotted as shown in Figure 3.8b. The last one is the concentrated attention map in which one or two views dominate the other views in the output and is shown in Figure 3.8c.



(a) Diffused (b) Intermediate (c) Concentrated

Figure 3.8: Three types of Attention Maps showing view-wise feature importance

Another method for view pooling was experimented with, other than feature-wise attention, which we termed as view-wise attention. In view-wise attention, the attention

weights $w_i \in R$ are real numbers summing to 1. For each view, the value vectors z_i are scaled by the attention weights w_i and summed to generate the view pool. This can be interpreted as giving the same importance to every feature of a view. The experiments have shown that the higher degree of freedom in feature-wise attention allows the network to achieve better classification performance than view-wise attention. Figure 3.6 can be used to represent both methods: feature-wise attention or view-wise attention, with the only difference being the dimensionality of the keys k_i and weights w_i.

3.3.3 Parameter Setting

We train the model MVCNN-SA for 75 epochs with a training time of 32 hours and a testing time of 0.3 hours. We use Adam optimizer with an initial learning rate of 0.0001. A decay of 0.1 is used for every 30 epochs, along with a momentum of 0.9. Cross-Entropy Loss with Softmax activation is used for classification in the output layer. The softmax activation function takes an N-dimensional real vector as input and outputs an N-dimensional real vector with values in the range (0, 1) that add up to 1. It is usually used as the last layer before calculating loss. Softmax uses Eqn.(3.6) to calculate the probability of each class prediction in the input vector.

$$\text{Softmax}(x_j) = \frac{\exp(x_j)}{\sum_{k=1}^{N} \exp(x_k)} \quad \forall j \in 1.....N \tag{3.6}$$

Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross entropy loss calculates the

distance between the output of the deep neural network and the original distribution. This loss increases as the predicted probability diverge from the actual label. Cross entropy loss for multiclass classification is calculated as a separate loss for each class label per observation and sum the result, is shown in Eqn.(3.7).

$$\text{Cross Entropy Loss, } \; CE_{Loss} = -1/n \sum_{i=1}^{n} t_i log(p_i) \tag{3.7}$$

where

- n - number of classes
- p_i - predicted probability for i_{th} observation
- t_i - truth value for i_{th} observation
- log - the natural log

Since a high-class imbalance is present in the RoofN3D dataset, for training our network with N instances per batch, we sampled each instance in the batch by the inverse frequency of the class. Thus, instances in larger classes have a lower probability of being selected. Due to population differences of the classes, the sampled batch is equally distributed, which can also help improve prediction accuracy. Suitable transformations for dataset augmentation are applied to the images during samplings, such as random resized crop, random horizontal flip, and normalization with mean and standard deviation.

3.3.4 Retrieval

The feature vectors from the penultimate MLP layer are extracted using the trained classification model for MVCNN-SA to evaluate the shape retrieval scores. The most relevant shapes in the test set (Roof point cloud database) are sorted for each query by cosine distance, and the Mean Average Precision (MAP) score is computed.

3.4 RESULTS AND DISCUSSION

The measures commonly used to evaluate the performance of 3-D shape classification and retrieval are used for evaluating the performance of the proposed work and are discussed in detail below. We first discuss the evaluation metrics for 3-D shape classification followed by metrics used for retrieval.

1. **Classification Accuracy**:

   An intuitively appealing measure is classification accuracy, which is defined as the ratio of correct predictions to total predictions made. A correct classification means that the learned model predicts the same class as the original class of the test case. Alternatively, the accuracy of a classification model on a test set (as shown in Eqn.(3.8)) may be defined as follows:

$$Accuracy, A = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.8}$$

   - True Positives (TP) means the predicted value matches the actual value. That is, the actual value was positive, and the model predicted a positive value.
   - True Negatives (TN) means the predicted value matches the actual value. That is, the actual value was negative, and the model predicted a negative value.
   - False Positives (FP) means the predicted value was falsely predicted. That is, the actual value was negative, but the model predicted a positive value.
   - False Negatives (FN) means the predicted value was falsely predicted. That is, the actual value was positive, but the model predicted a negative value.

2. **Precision** :

   The precision is calculated as the ratio between the number of positive samples correctly classified to the total number of samples classified as positive. The precision measures the model's accuracy in classifying a sample as positive and is shown in Eqn.(3.9).

$$Precision, P = \frac{TP}{TP + FP} \tag{3.9}$$

3. **Recall** :

   The recall is calculated as the ratio between the number of positive samples correctly classified as positive to the total number of positive samples which is shown in Eqn.(3.10). The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples detected.

$$Recall, R = \frac{TP}{TP + FN} \tag{3.10}$$

4. **F1_score** :

   F1 Score is the harmonic mean between precision and recall which is shown in Eqn.(3.11). The range for F1 Score is [0, 1]. It tells how precise is the classifier as well as how robust it is.

$$F1\_score = \frac{2 \cdot P \cdot R}{P + R} \tag{3.11}$$

5. **Confusion Matrix** :

   A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the model. Classification accuracy alone can be misleading if the dataset has an unequal number of observations in each class or has more than two classes. Therefore, the performance of a classifier is also evaluated via the confusion matrix, which displays the number of correct and incorrect predictions made compared with the actual classifications in the test set.

6. **Mean Average Precision**:

   The metric Mean Average Precision (MAP) is employed to quantify the retrieval performance. The Mean Average Precision is defined as follows in Eqn.(3.12):

$$MeanAveragePrecision, MAP = \frac{1}{N} \sum_{i=1}^{N} AP@i \tag{3.12}$$

- N - number of queries in the test set
- Average Precision for a given query k, $AP@k = \frac{1}{GTP}\sum_{k}^{n} P@k$
- GTP - total number of ground truth positives
- n - total number of interested shapes,
- $P@k$ refers to the precision for a value k.

Intuitively, MAP is considered the area below the precision-recall graph. A perfect retrieval algorithm has MAP = 100% and a higher value indicates better results.

Before the experimentation of MVCNN-SA, the existing deep learning architecture ResNet-50 [He *et al.* (2016)] was used for the classification of ALS roof point clouds. This method used all views of the point cloud individually and resulted in lower accuracy (shown in Figure 3.9) compared to the proposed method. After experimenting with two types of pooling mechanisms used in MVCNN-SA mentioned in section 3.3.2, it was found that MVCNN-SA with feature-wise/view-wise pooling performs better than MVCNN with max/average pooling. In our experiments, we have used a MVCNN architecture similar to MVCNN-SA (with Resnet-18 for CNN_1 and without a CNN_2 after view-pooling). Sarthak et al. have applied the PointNet method on RoofN3D and have obtained the results for classification of roof types [Guptha and Bohare (2019)]. The comparison of results of MVCNN-SA with MVCNN and the PointNet is shown in Figure 3.9. The results of MVCNN-SA have surpassed the results of MVCNN and Sarthak et al. [Guptha and Bohare (2019)] on the RoofN3D dataset.

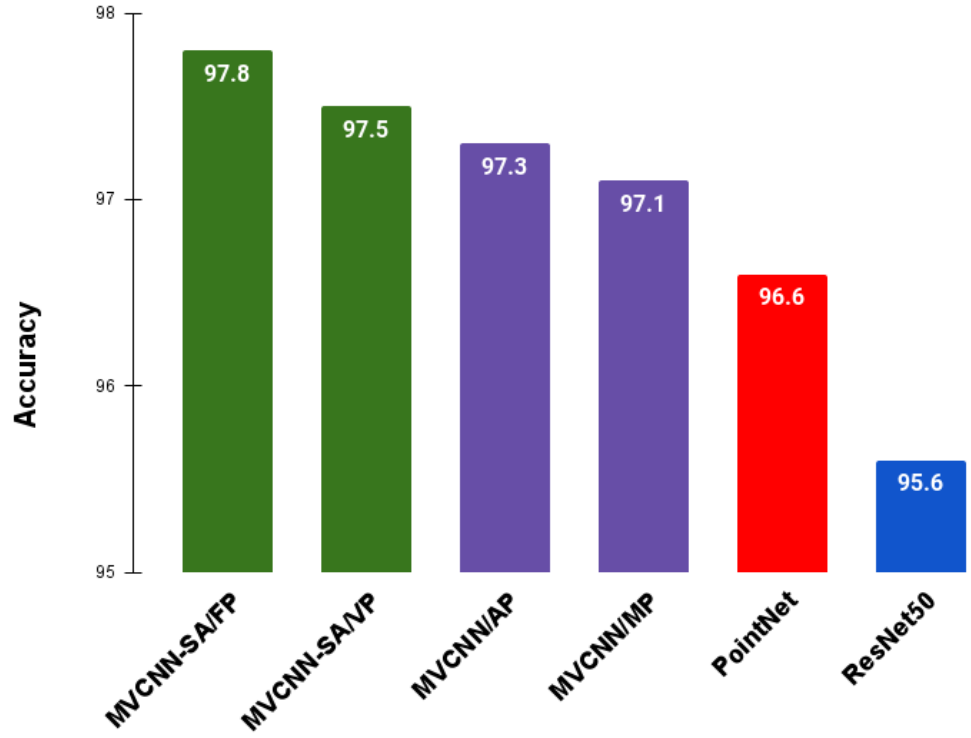


Figure 3.9: Test Accuracies for various models (MVCNN-SA/FP - MVCNN-SA with Feature-wise Pooling, MVCNN-SA/VP - MVCNN-SA with View-wise Pooling, MVCNN/AP - MVCNN with Average Pooling, MVCNN/MP - MVCNN with Max Pooling).

Table 3.2: Confusion Matrix of MVCNN-SA/FP trained on RoofN3D

	Saddleback	**Two-sided Hip**	**Pyramid**
Saddleback	13128	78	7
Two-sided Hip	199	3902	35
Pyramid	32	45	286

The confusion matrix for MVCNN-SA with the feature-wise pooling approach is shown in Table 3.2. The results show that the proposed approach performs well in classifying all the roof types. Errors mostly come from classifying the pyramid class, which is under-represented in the dataset.

The comparison of precision, recall and F1-score values of MVCNN-SA with Sarthak et. al (i.e, PointNet results [Guptha and Bohare (2019)]) is shown in Table 3.3.

Table 3.3: Comparison of classification results of proposed model with Sarthak et. al

Class	**MVCNN-SA**			**PointNet**		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Saddleback	0.98	**0.99**	**0.99**	**0.99**	0.97	0.98
Two-sided Hip	**0.97**	0.94	**0.96**	0.92	**0.95**	0.93
Pyramid	**0.87**	0.79	**0.83**	0.69	**0.83**	0.75

Further, we observed that the concentrated and diffused attention maps showed poor accuracy. A diffused attention map might give importance to views with irrelevant or redundant information, leading to poor performance. A similar argument holds true for concentrated attention maps, as they might ignore views with important information. We found that the intermediate attention map, which has an entropy coefficient close to zero, produces a balance between these two behaviors and also gives the highest accuracy as shown in Table 3.4.

Table 3.4: Comparison of Accuracy based on entropy coefficients.

Entropy Values	1	10^{-3}	10^{-5}	0	-10^{-5}	-10^{-3}	-10^{-1}
Accuracy for original views [%]	97.26	97.74	**97.76**	97.65	97.54	97.22	97.14
Accuracy for transformed views[%]	96.5	97.28	97.26	**97.4**	97.24	97.15	97.18

We experimented with a change in the camera's elevation, varying randomly between

(-30,30) degrees, and captured the transformed views for training, validation, and testing. The performance obtained for the transformed views was not drastically lower compared to results on the original dataset, as shown in Table 3.4. This may be because even if some of the cameras shifted to places where they could not capture much information, other views would compensate for this. In the final experiment, we captured 20 views where $\theta = 18°$ to verify that increasing the number of views does not cause significant gains in performance, which was also found to be true in [Su *et al.* (2015)].

Table 3.5: Classification Accuracies and Retrieval Scores of MVCNN-SA and PointNet

Method	**Accuracy(%)**	**MAP**
PointNet	96.57 [Guptha and Bohare (2019)]	90.8
MVCNN-SA	**97.76**	96.94

The comparison of classification accuracies and retrieval scores of MVCNN-SA with the PointNet on RoofN3D is shown in Table 3.5. The model parameters, size and forward pass time for various models are shown in Table 3.6.

Table 3.6: Model Parameters, Size and Forward Pass time

Method	**#Params (M)**	**Size (MB)**	**Time (ms)**
ResNet-50	23.51	94.4	18.98
PointNet	3.5	40	25.3
MVCNN[ResNet-18]	11.17	44.8	22.11
MVCNN-SA	**15.34**	**61.5**	**27.2**

3.4.1 Experimental Setup

All the experiments were run on a computing node running Ubuntu 16.04 with an Intel(R) Xeon(R) CPU E5-2630 V3 @ 2.40GHZ, 80 GB RAM, and an NVIDIA GTX 1080 Ti GPU. PyTorch 1.0 was used for deep learning. Blender software 2.79-b-Linux was used for generating the rendered views from ALS roof point clouds.

3.5 SUMMARY

An adaptive importance-learning algorithm was proposed to discover efficient image descriptors from multiple views of ALS roof point clouds. The information across all views is fused to avoid loss of information in the shape descriptor. The feature-wise attention approach was more effective than view-wise attention, max pool, and average pool. Using this novel approach, we were able to surpass the results by the PointNet method on the RoofN3D dataset. However, the performance of our model could still be improved from the current results with hyperparameter tuning and changing the architecture of the self-attention network.

CHAPTER 4

CLASSIFICATION AND RETRIEVAL OF ALS ROOF POINT CLOUDS USING POINT TRANSFORMER

4.1 INTRODUCTION

The recent research in deep learning for 3-D point clouds is more focused on point-based methods despite the huge success of view-based methods for classification and retrieval. Point-based representation does not require conversion of the point cloud to any other representation, making it more efficient in terms of computational cost, space, and time than view-based or voxel-based representations. View-based and voxel-based methods lead to unnecessary artifacts during conversion and can also lead to loss of geometric information. To alleviate this problem, point-based methods directly utilize point-wise features such as coordinate (X, Y, Z) information, and hence they are increasingly becoming popular these days. Existing point-based methods utilize convolutions that have some limitations, as they assume a uniform input distribution and cannot learn long-range dependencies. Recent works [Xie *et al.* (2018); Liu *et al.* (2019*a*,*b*); Lee *et al.* (2019); Yang *et al.* (2019)] have shown that adding attention in conjunction with these methods improves performance. This reality raises a question: can attention layers completely replace convolutions? This research, therefore, proposes a fully attentional model - Point Transformer, for deriving a rich point cloud representation for complete and partial ALS roof point clouds.

This chapter describes a study that explores the possibility of completely replacing convolutional layers with attention blocks. The motivation for this study is that

convolutional kernels get activated at specific regions, and similarly, attention layers focus on specific regions of the point cloud. We also hypothesize that attention-based methods might be an even more natural fit for point cloud processing, as attention in its core is a set operation: implying that it is invariant to permutation and cardinality of the input elements, making it ideal for point clouds. The proposed method - Point Transformer (PT) outperforms other state-of-the-art models in a real dataset - RoofN3D [Wichmann *et al.* (2018)], and to ensure a fair comparison with previous works, the model is evaluated on a widely used benchmark dataset for point cloud classification- ModelNet40. Despite all the progress in the 3-D point cloud classification, it is difficult to find a single architecture that consistently performs better when evaluated on transformed test sets. Almost all methods drastically fail when evaluated on real datasets with unseen input transformations. Therefore, it is highly essential to innovate a novel point-based architecture that is consistent and robust to unseen input transformations in the test set. To determine the model's robustness to various unseen point corruptions (noise, sparsity, occlusion, etc.), it is tested on a dataset RobustPointSet [Taghanaki *et al.* (2020)] and the model outperforms other methods.

The key contributions of the proposed work are as follows:

- Novel approach to show self-attention can completely replace convolutions in point cloud processing.
- The proposed iterative transformer (with weight-sharing) can hierarchically learn complex information with a significantly lesser number of parameters.
- A learnable grouping mechanism with a routing loss is introduced to group semantically similar points and derive region/group-wise feature vectors.
- Focus on the model's robustness to unseen corruptions apart from classification performance on benchmark datasets.

- Our specific design considerations ensure that the model is highly time and memory-efficient, making it ideal for real-time use cases.

4.2 PRELIMINARIES

4.2.1 Attention

Recently, Natural Language Processing, Computer Vision, and Robotics have been revolutionized by the attention mechanism. Attention is the process of selectively focusing on the most relevant part of the input information instead of focusing on an entire input. There are different types of attention mechanisms used, and self-attention is a widespread technique among them. Self-attention, also known as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence. An attention function can be described as mapping a query and a set of key-value pairs to an output. Calculating self-attention can be primarily done in a few steps. In the first step, the key vector K, the query vector Q, and the value vector V are derived from the input. The second step in calculating self-attention is to multiply the Query vector Q with the key vector K. In the third step, the obtained score is divided by the square root of dimensions of the key vector (d_k), for example: if the dimension of the key vector is 64, then the score will be divided by 8. The reason behind that is, if the dot products become large, it can cause the self-attention scores to be very small after normalization. In the fourth step, the softmax function is applied to all self-attention scores calculated with respect to the query. In the fifth step, the value vector is multiplied by the attention matrix calculated in the previous step to aggregate the derived representation based on inter-dependencies. These steps are

summarised in Eqn.(4.1), where Q, K, V are the query, key, and value vectors.

$$Attention(Q, K, V) = Softmax(QK^T/\sqrt{d_k})V \tag{4.1}$$

As seen above, unlike traditional sequential models such as Recurrent Neural Networks (RNN) and Long short-term memory (LSTM), the attention operation does not depend on the input sequence order. This makes it a natural fit for point cloud processing.

4.2.2 Transformers

The Transformer architecture is an extension of the attention operation, specifically using a Multi-Head Attention operation (MHA). Rather than computing the attention once, the MHA operation computes it multiple times (heads). This helps the transformer jointly attend to different information derived from each head. The output from each of these heads is concatenated before projecting onto a final output dimension. A transformer layer also contains a residual connection followed by a layer normalization. To summarise, the main properties of a transformer are:

- Transformers, in its core, uses the attention operation, which is a set operator making it ideal for point cloud processing, i.e., they are inherently cardinality and order invariant.
- The attention operation derives three feature vectors Q, K, V using individual MLPs from the input sequence. The dot product of Q and K is used to compute the relationships between the input points and indicates the most important portions of the input sequence. These weight values are used to aggregate the V vector to derive an enhanced representation.
- Unlike convolutions and shared MLP layers, the attention operation computes relationships between the input sequence to derive a rich feature representation and can model long-range dependencies.

There are some limitations in existing transformer methods:

- In NLP, the length of the input sequence is relatively low when compared to point clouds (>1000). Hence, transformers cannot be directly applied for point cloud processing.
- Due to the large sequence length, the attention matrices are relatively sparse and can model unwanted relationships, leading to over-fitting.

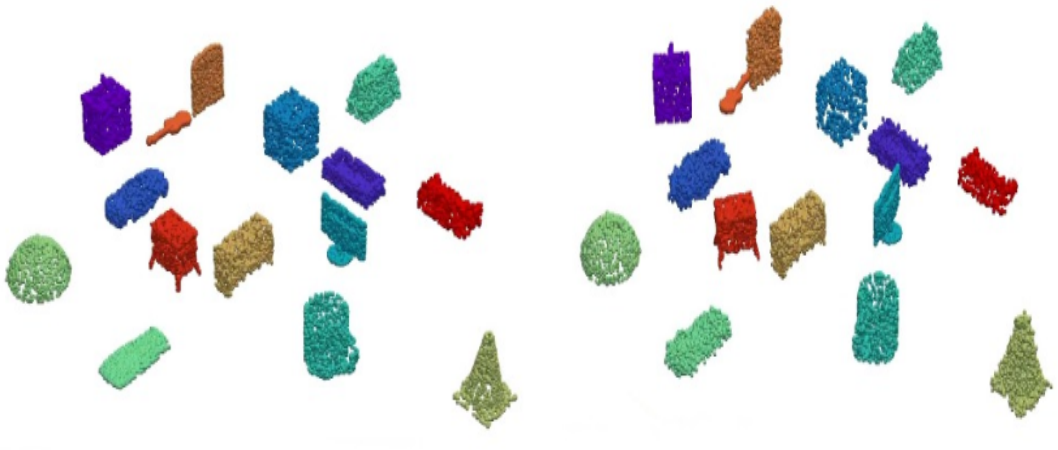
Therefore, we need to make suitable changes to achieve high performance across various point cloud processing tasks.

4.2.3 Dataset

The model's shape classification and retrieval performance are evaluated on a large-scale urban dataset of roof point clouds - RoofN3D [Wichmann *et al.* (2018)]. The dataset is uniformly split into 95,632 train, 10,627 validation, and 11,814 test samples for this study. The RoofN3D dataset was already detailed in Section 3.2.4 in Chapter 3.

Being a new approach and to ensure a fair comparison with previous works, the model's performance is also evaluated on a widely adopted benchmark for 3-D point cloud classification - ModelNet40 [Wu *et al.* (2015)]. It contains 12,308 CAD models of 40 categories split into 9,843 train and 2,468 test samples. For accurate comparison with other works, we use the pre-processed ModelNet40 dataset provided by PointNet++ [Qi *et al.* (2017)] with the standard train-test split. Each point cloud is randomly sampled to 1024 points as input to the networks unless otherwise stated. The point clouds are centered at zero, and coordinates (x, y, z) are normalized between $[-1, 1]$. However, the objects in ModelNet40 are synthetic, aligned, complete, and noise-free, which does not

simulate real-world noise. In practice, deep learning models are often deployed to work with real-world point cloud scans obtained from 3-D sensors. An input point cloud can be transformed in several ways like rotation, translation, the addition of noise, etc. Some samples in the original dataset and transformed dataset can be visually seen in Figure 4.1a and 4.1b. This raises an interesting question: Can classification methods trained on clean, aligned, and noise-free datasets perform well on real-world data?



(a) Original Dataset (b) Transformed Dataset

Figure 4.1: A few samples of the original dataset representing ModelNet40 and modified dataset with different transformations representing RobustPointSet

Hence to evaluate the robustness of our proposed method, we use another benchmark dataset - RobustPointSet [Taghanaki *et al.* (2020)] which contains transformed samples of ModelNet40. These transformations include - addition of noise, sparsity, partial shape removal, rotation, translation, occlusions. Each transformation is described as follows:

- Noise: random noise from a normal distribution is added to each point in the

sample.

- Missing part: each object is split into eight axis-aligned chunks, and one is randomly deleted.
- Occlusion: Given an object, a random wall or a random part from another object is added to it. The random part is taken similar to the 'Missing part' set, while walls are points sampled from a plane added to the top, bottom, left, right, front, or back of the object.
- Sparse: 93% of the 2048 points of each object is randomly set to zero, i.e., each object has only 128 valid points in this set.
- Rotation: a random rotation applied on an arbitrary axis to each object.
- Translation: Each object is translated by a 3-D vector t sampled from the standard distribution.

It is important to note that we do not train the model on these transformations, and a model trained on a clean point cloud is evaluated on these unseen point corruptions. Additionally, we simulate a few corruptions on RoofN3D and evaluate the model's performance using a similar strategy. Our experiments suggest that the proposed method gives a high performance on ALS roof point clouds and synthetic benchmark datasets while also being highly robust to unseen transformations.

4.2.4 Existing Methods using Self-Attention

A few works utilize self-attention like Attention Shape ContextNet (ASCN), Set Transformers, and Point Attention Transformers (PAT). However, they attempt to apply self-attention for point clouds directly. ASCN uses a sequential stack of attention blocks to derive point features that are max pooled to create a global feature representation, however direct max-pooling results in loss of information. Set Transformers uses a similar idea to ASCN but introduces an attention-based pooling operation to prevent

loss of information. But it leads to overfitting due to over-expressive attention matrices. PAT uses a parameter-efficient group shuffle attention with a learnable Gumbel Subset Sampling method. All the above novel methods showcase lower performance when compared to other SOTA methods. We attempt to utilize the learning capacity of transformers and introduce a novel architecture for point cloud classification and retrieval.

4.3 METHODOLOGY

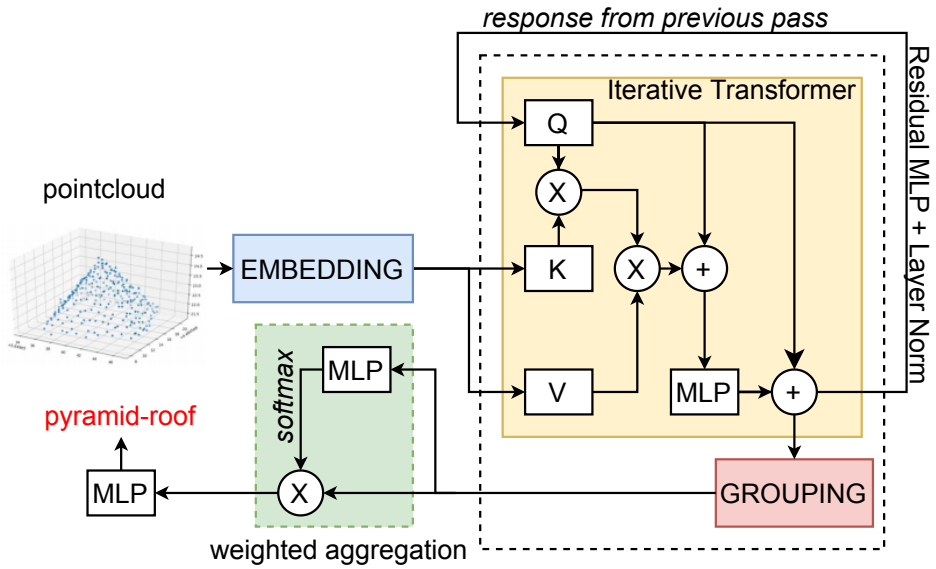


Figure 4.2: Architecture of Point Transformer
It comprises four modules: Embedding, Iterative Transformer, Grouping and Weighted Aggregation

Our proposed architecture - Point Transformer (PT) as shown in Figure 4.2 consists of the following modules:

- Embedding Module
- Iterative Transformer

- Grouping Module with Routing Loss
- Weighted Group Aggregation

4.3.1 Embedding Module

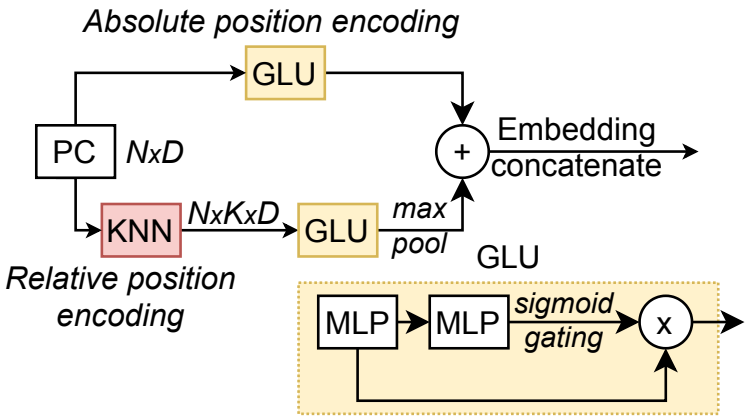


Figure 4.3: Architecture of Embedding Module

The input point cloud of size N represented as xyz coordinates contains information about the absolute position of every point. While this is quite expressive, every point can also be represented by its relative position. To avail both this information, an embedding module (as shown in Figure 4.3) is introduced containing two branches - one to encode the absolute position and the other to derive the relative position information. The absolute position details are encoded by feeding each point into a Gated Linear Unit (GLU) to propagate the required information selectively. In the other branch, k-nearest neighbors are computed to derive the relative position of each point. This is then passed through another GLU layer, followed by a max-pooling operation to create a suitable vector. The point representations from each branch are then concatenated to derive the enriched point-wise embedding. The embedding module can be made robust to variable

point cloud size by reducing the value of k based on the number of points. $k = k_0 * N/N_0$ where $k_0 = 32$, $N_0 = 1024$ are constants.

4.3.2 Iterative Transformer

Transformers have shown huge success in natural language processing and have recently displayed some potential in image processing. They are inherently order-invariant to a given sequence, making them ideal for point cloud processing. The transformer layer is characterized by the scalar dot product (SDP) attention operation given in Eqn.(4.2). The idea behind the attention mechanism is to learn complex relationships between the elements of a sequence and focus on certain parts of it. These relationships are represented by an NxN attention matrix computed as the dot product of query (Q) and key (K) vectors. This attention matrix is then multiplied with the value vector (V) to derive a rich feature representation. The Q, K, V vectors are obtained using individual Multi-Layer Perceptrons (MLP) from the input sequence. Rather than computing the attention once, a Multi-Head Attention (MHA) mechanism [Vaswani *et al.* (2017)] computes the SDP operation multiple times. This strategy helps the transformer jointly attend to different information derived from each head.

$$SDP(Q, K, V) = Softmax(QK^T/\sqrt{N})V \tag{4.2}$$

Existing methods utilize a stack of sequential transformers to learn an element-wise representation. However, given a large sequence, multiple MHA operations become very expressive, leading to overfitting [Lee *et al.* (2019)]. Deviating from these

methods, we introduce an Iterative Transformer (shown in Figure 4.4) to recursively improve the feature representation across M passes. In the first pass, the Q, K, V vectors are derived from the point embeddings, and the output is passed through a residual MLP layer with Layer Normalization. In the subsequent passes, the Q vector is updated using the output from the previous pass, and the K, V vectors are retained as the same. It is important to note that the weight parameters are shared across the passes and help learn complex features hierarchically. The above operation ensures that at every pass, the attention layers try to selectively refine the information learnt from the previous ones while keeping the number of parameters fixed irrespective of the value of M. The four passes of the Iterative Transformer are illustrated in Figure 4.5.

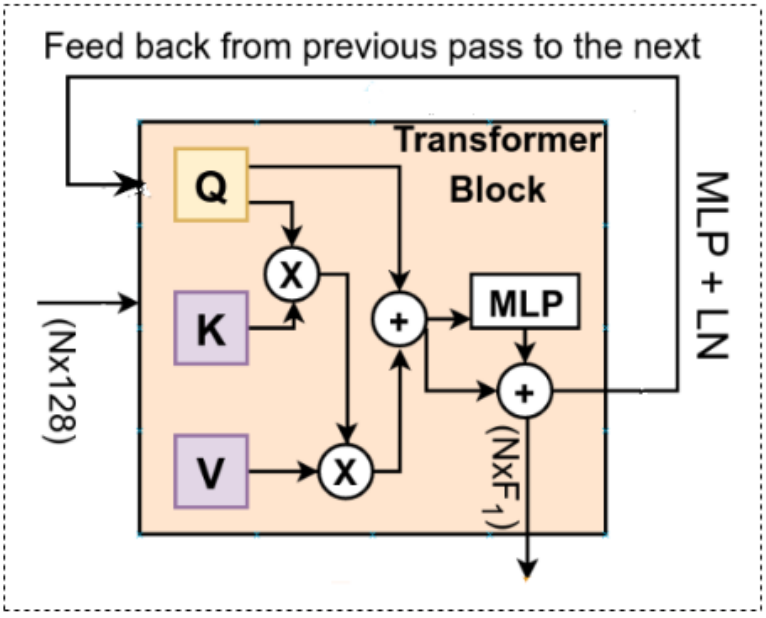


Figure 4.4: Architecture of Iterative Transformer

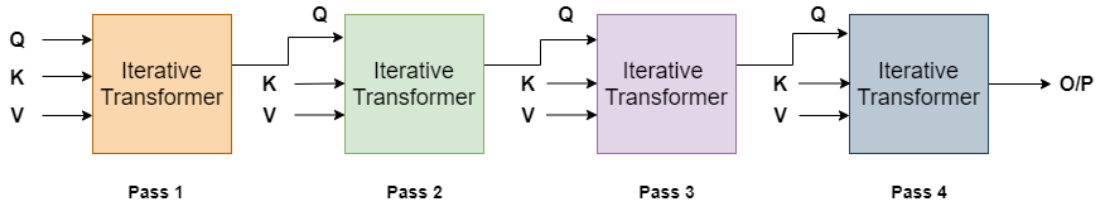


Figure 4.5: Four Passes of Iterative Transformer

4.3.3 Grouping Operation

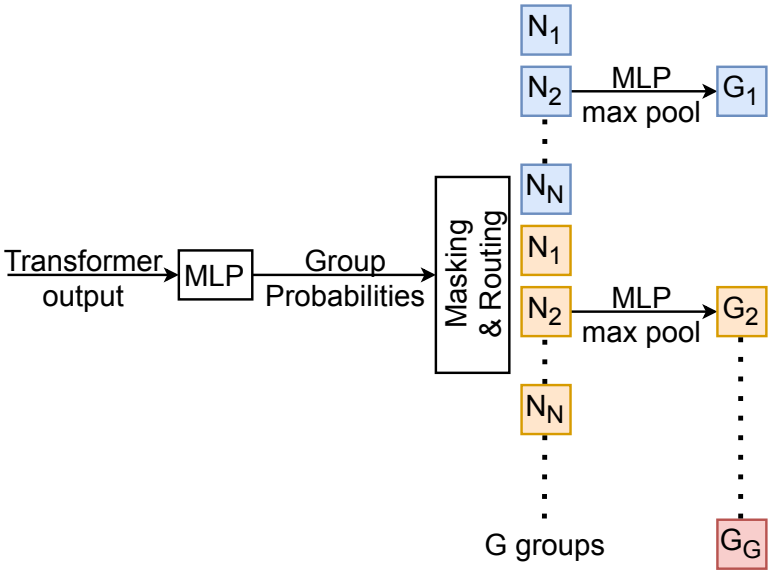


Figure 4.6: Architecture of Grouping Module

The architecture of the grouping module is illustrated in figure 4.6. We know that a point cloud is characterized by multiple groups of points together representing the complete shape. As an alternative to direct pooling of the derived point representation, we create region-wise or group-wise feature vectors using a learnable grouping operation. To split the points into groups, probability values are predicted for each point, determining the best group for it. The points are then hard-routed based on these probability values, which implies that a point can belong to only one group, and no residue of the same is passed to the others. In each group, a specific feature transformation is applied

to the points using an MLP layer followed by a max-pooling operation to create a group feature vector. It is imperative to ensure that the model explores all available groups for each point before converging to the best group for a point. Hence at times during training, the points are routed to the group corresponding to their second-highest probability. This strategy indirectly ensures that all the points are not routed to a single group. Since the model derives different features across passes, we perform the grouping operation at every pass to derive the final shape representation.

Routing Loss

During grouping, it is also essential to ensure a sufficient number of points in every group to capture region-specific information. The routing loss (given in Eqn.(4.3)) is introduced to penalize the model based on the distribution of points among the groups. Here N_g is the number of points routed to the g^{th} group, and N is the total number of points in the input point cloud.

$$RoutingLoss, R_L = \sum_{g=1}^{g=G} (\frac{N_g}{N})^2 \tag{4.3}$$

To provide an intuition, let us assume f_1 and f_2 to be the fraction of points routed to any two groups. If both these fractions of points are routed to a single group, then the value of routing loss will be higher as $(f_1 + f_2)^2 >= f_1^2 + f_2^2$ and similarly, this idea can be extended to G groups. It is important to note here that the model is still free to route any number of points to a group and that the routing loss, random routing are just measures to prevent the model from getting biased to a selective number of groups.

4.3.4 Weighted Group Aggregation

The group aggregation operation combines the group feature vectors obtained from all passes to create a point cloud representation. A weight matrix is derived using an MLP layer to select relevant information from these groups, normalized using a softmax operation. These weights are multiplied with the corresponding features and then added to derive the global feature vector. This operation is equivalent to $sum(softmax(ax + b) * x)$ where x denotes the input, a and b represent the weights and bias terms of the MLP layer respectively.

The derived global feature vector is passed onto two MLP layers to enhance it before the final classification layer. The model is trained to minimize the smooth cross-entropy loss [Wang *et al.* (2019)] along with the routing loss as mentioned above.

4.4 EXPERIMENTS AND RESULTS

We have conducted comprehensive experiments to validate the effectiveness of the Point Transformer. The details of the experiments are given below.

4.4.1 Experiment Setting

We use the following hyper-parameters to train Point Transformer for the classification task. The model derives 128 size vectors from the embedding module, subsequently fed to the transformer block with hidden size 128 and a grouping module to create group feature vectors of size 512. We choose M(number of passes) = 4 and G(number of groups) = 4 for all our experiments. The network is trained using an Adam optimizer

with a batch size of 18 and an initial learning rate of 0.001, which is reduced by a factor of 0.7 every 20 epochs, and the model converges in less than 100 epochs. All experiments are run on an NVIDIA GTX 1080Ti GPU. As shown in Figure 4.7 the training loss, the model has converged in about 70K steps. Figure 4.8 shows how the training and validation accuracies are varying with epochs. The highest validation score is obtained in the 34th epoch and didn't increase further. Hence we concluded that the model has converged around 40 epochs.

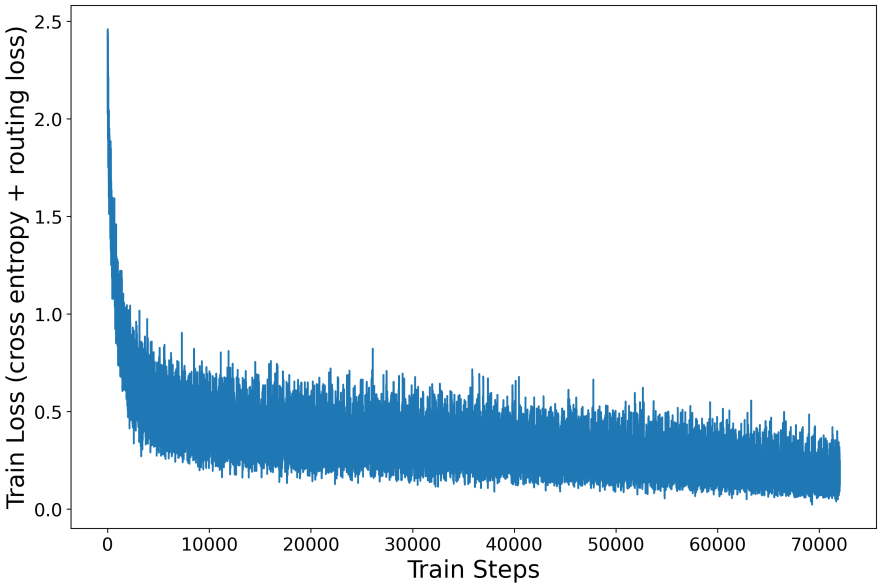


Figure 4.7: Train Loss Vs Steps.

4.4.2 Shape Classification and Retrieval

The RoofN3D dataset is uniformly split into 95,632 train, 10,627 validation, and 11,814 test samples. By default, 1000 points are selected for the RoofN3D dataset by uniform sampling and 1024 points by farthest point sampling for ModelNet40. During training,

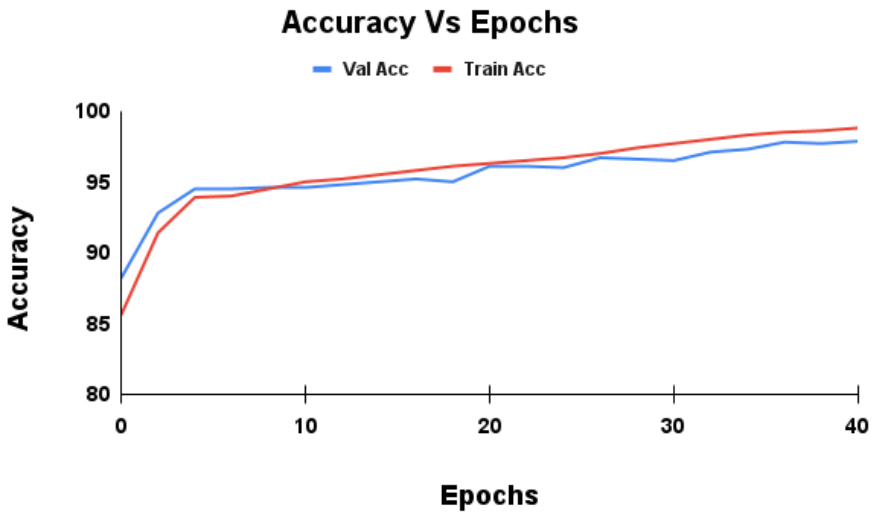


Figure 4.8: Train and Validation Accuracies Vs Epochs

the input is augmented by randomly scaling the shape in the range of $(0.8, 1.25)$ and randomly translating in the range of $(-0.1, 0.1)$. The feature vectors are extracted from the penultimate MLP layer using a trained classification model to evaluate the shape retrieval scores. The most relevant shapes are sorted for each query by cosine distance, and the Mean Average Precision (MAP) score is computed.

Table 4.1: Shape classification and retrieval results on RoofN3D

Method	**Accuracy(%)**	**MAP**
PointNet	96.57 [Guptha and Bohare (2019)]	90.8
PointNet++	97.03	94.3
PointCNN	97.39	94.37
SO-Net	97.78	-
MVCNN-SA [Shajahan *et al.* (2020)]	97.76	96.94
PT (Ours)	**97.86**	**97.34**

Tables 4.1 and 4.2 compare the shape classification and retrieval results of our

method with other methods on the RoofN3D and ModelNet40 datasets. Point Transformer performs better than all SOTA models in the RoofN3D dataset, signifying its effectiveness in real datasets with multiple imperfections. In ModelNet40, the model surpasses the performance of point-based methods such as PointNet++, PointCNN, and SONet, attention-based methods like L2G, Attention ShapeContextNet, and even graph-based methods like DGCNN. The model also outperforms other methods in the shape retrieval task.

Table 4.2: Shape classification and retrieval results on ModelNet40

(a) Shape Classification results

Method	**Accuracy(%)**
PointNet [Charles *et al.* (2017)]	89.2
ASCN [Xie *et al.* (2018)]	90.0
MVCNN [Su *et al.* (2015)]	90.1
Set Transformer[Lee *et al.* (2019)]	90.4
L2G [Liu *et al.* (2019*b*)]	90.6
PointNet++ [Qi *et al.* (2017)]	90.7
SO-Net[2k] [Li *et al.* (2018*a*)]	90.9
PAT [Yang *et al.* (2019)]	91.7
DGCNN [Wang *et al.* (2019)]	92.2
PointCNN [Li *et al.* (2018*b*)]	92.2
PCNN [Liu *et al.* (2019*c*)]	92.3
PT (Ours)	**92.5**

(b) Shape Retrieval results

Method	**MAP**
PointNet [Charles *et al.* (2017)]	70.5
ASCN [Xie *et al.* (2018)]	71.83
MVCNN [Su *et al.* (2015)]	79.5
PointCNN [Li *et al.* (2018*b*)]	83.8
DGCNN [Wang *et al.* (2019)]	85.3
PT (Ours)	**88.4**

Figure 4.9 shows samples retrieved for a given query shape for both datasets. In the roof retrieval case, complex substructures present in the query shape are retained in the

retrieved samples.

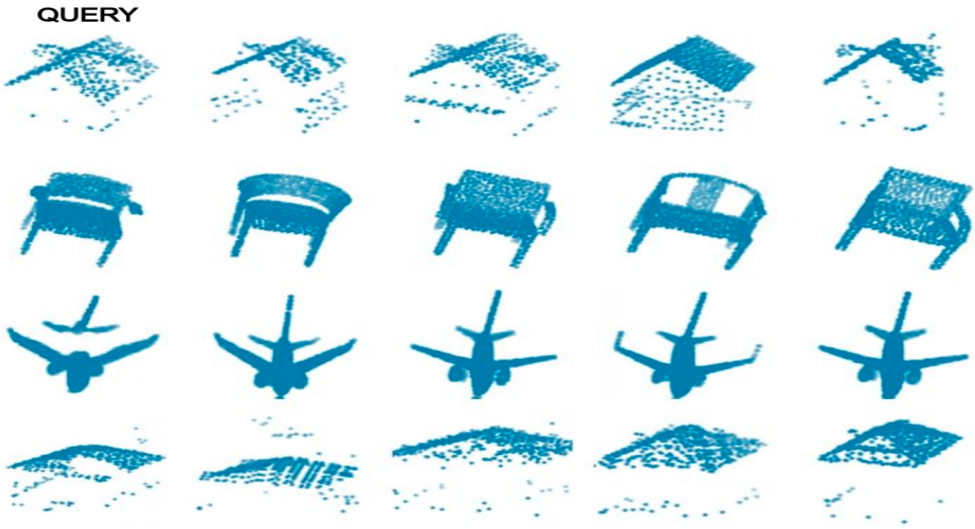


Figure 4.9: Shape retrieval using complete query (illustrated in top 2 rows) and partial query (illustrated in bottom 2 rows)

4.4.3 Test for Robustness

Most existing methods primarily focus on improving classification accuracy on clean, aligned datasets. It is equally important to test the model's robustness on unseen transformations to prove its applicability for real datasets. For this experiment, we follow the procedure given in RobustPointSet by training the model on 2048 points without any augmentation and test it for various unseen corruptions. Table 4.3 shows results for various models on these unseen test sets, and it is quite evident that our proposed method outperforms other techniques. Among the other methods, PointNet seems to be a robust model even though it does not achieve SOTA performance in the classification tasks. As seen from Table 4.3, most methods fail significantly in cases like Sparse, Noise, Rotation. Methods that rely on local information suffer greatly since the

notion of the neighborhood is changed drastically on corruption. During rotation, 3-D coordinate positions change and hence affect the performance of all models.

Table 4.3: Results for robustness tests in RobustPointSet (ModelNet40)

Method	**Orig**	**Noise**	**Trans**	**Missing**	**Sparse**	**Rot**	**Occ**	**Avg. Acc**
PointNet	89.06	74.72	79.66	81.52	60.53	8.83	39.47	61.97
PointNet++	91.47	14.90	91.07	50.24	8.85	12.70	70.23	48.49
DGCNN	92.52	57.56	91.99	85.40	9.34	13.43	78.72	61.28
PointMask	88.53	73.14	78.20	81.48	58.23	8.02	39.18	60.97
DensePoint	90.96	53.28	90.72	84.49	15.52	12.76	67.67	59.40
PointCNN	87.66	45.55	82.85	77.60	4.01	11.50	59.50	52.67
PointConv	91.15	20.71	90.99	84.09	8.65	12.38	45.83	50.54
RSCNN	91.77	48.06	91.29	85.98	23.18	11.51	75.61	61.06
PT (Ours)	**92.39**	**71.52**	**92.1**	**86.62**	**15.1**	**14.02**	**66.1**	**62.55**

Orig-Original, Trans-Translation, Missing- Missing points, Rot-Rotation, Occ-Occlusion, Avg. Acc-Average Accuracy

Table 4.4: Results for robustness tests in Roofn3D

Method	**Orig**	**Noise (75%)**	**Missing (50%)**	**Uniform removal (87.5%)**	**(75%) Partial shape retrieval (MAP)**
PointNet	96.57	95.07	83.22	75.52	83.89
PointNet++	97.03	75.55	77.73	70.02	83.45
PointCNN	97.39	76.32	80.43	73.60	86.5
PT (Ours)	**97.86**	**97.6**	**92.3**	**76.92**	**90.77**

Orig-Original, Noise(75%)-gaussian noise added to 75% points, Missing(50%)-50% points are removed from a single region, Uniform removal(87.5%)-87.5% points are uniformly removed

We extend these experiments to the RoofN3D dataset, and the results are shown in Table 4.4. For testing robustness in the shape retrieval task, partial point clouds are used as the query shape, and corresponding closest matching full shapes are retrieved. This further

validates the expressivity of our method despite being given incomplete input shapes.

4.4.4 Visualising the Model

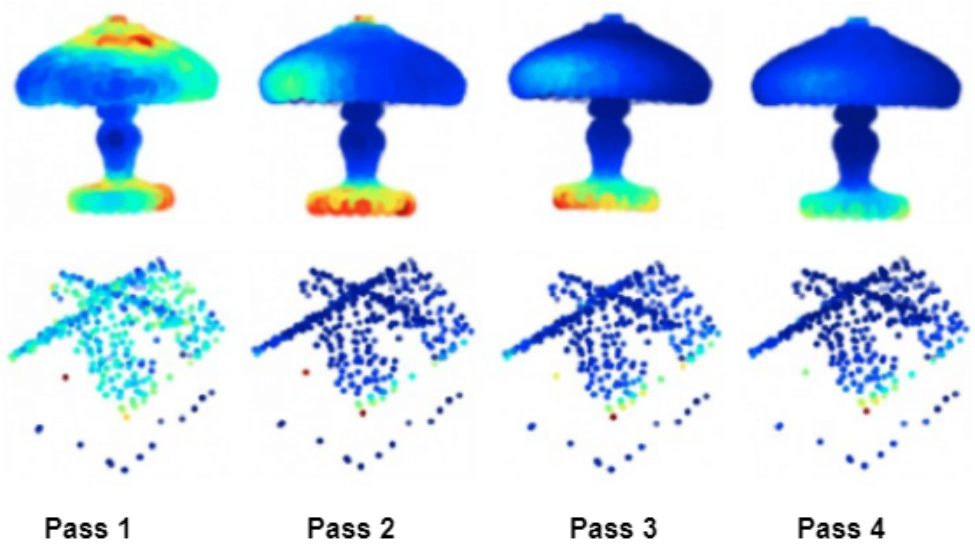


Figure 4.10: Average attention across passes (Red-High, Blue-Low Attention)
The attention is distributed across the model in the initial passes and converges to specific regions in the later passes.

To better understand and validate the operations in our method, we visualize the average attention maps and group formation. We compute the average attention by taking the mean across heads and for each point. As seen from Figure 4.10, the model distributes its attention across all points in the initial pass but later attends to specific regions, as it tries to improve the feature representation recursively.

Semantically similar points are grouped together describing a particular shape in the point cloud, as shown in Figure 4.11 where each group is given a different color.

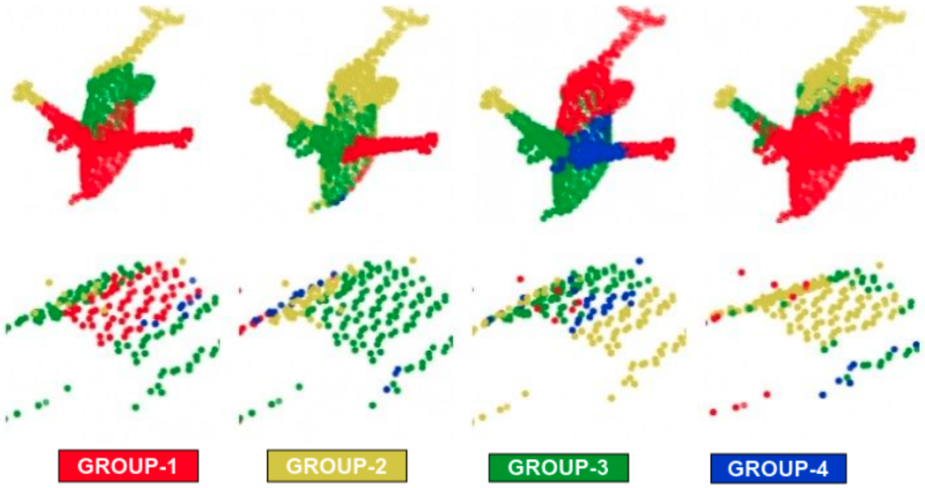


Figure 4.11: Group formation across passes

We also visualize the variation in average attention on increasing point corruptions (such as noise addition, reduced point density, partial shape removal), as shown in Figures 4.12 and 4.13. The interesting observation is that the model has focused on similar regions in all cases. Another interesting observation is that the model can identify noisy points and not pay attention to them.

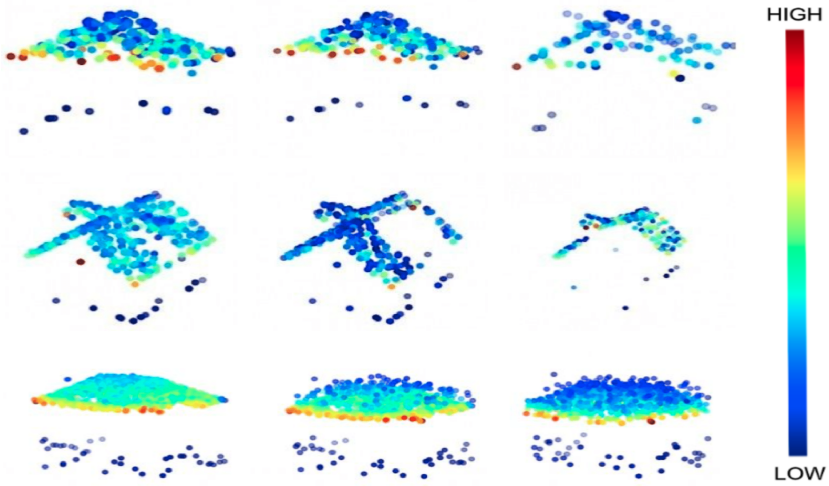


Figure 4.12: Attention Maps in RoofN3D (Red-High, Blue-Low) on various levels of Point Corruptions: reduced point density, partial shape removal, noise addition. (from top to bottom)

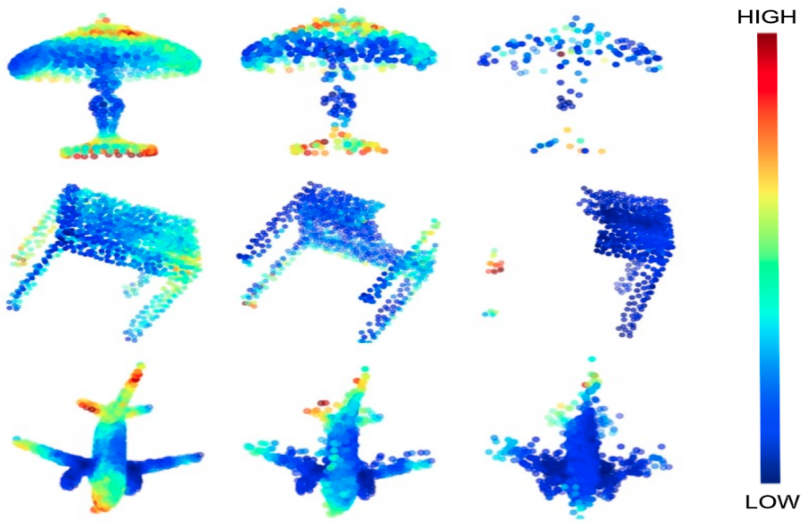


Figure 4.13: Attention Maps in ModelNet40 (Red-High, Blue-Low) on various levels of Point Corruptions: reduced point density, partial shape removal, noise addition. (from top to bottom)

4.4.5 Time and Space Complexity

Table 4.5: Model Parameters, Size and Forward Pass time

Method	**Params(M)**	**Size (MB)**	**Time (ms)**
MVCNN(12 views)	128.93	515.7	55.6
MVCNN-SA	15.34	61.5	27.2
PointNet	3.5	40	25.3
PointNet++ (MSG)	1.48	12	163.2
PointCNN	0.6 Li *et al.* (2018*b*)	6.9 Li *et al.* (2018*b*)	-
PT (Ours)	**1.03**	**4.1**	**10.9**

While view-based methods achieve high performance, they are very computationally expensive and less space-efficient than point-based methods, as shown in Table. 4.5.

Among the point-based methods, Point Transformer is a much smaller model with only 1.03 M parameters compared to other methods like PointNet, PointNet++ having 3.5M and 1.48M parameters, respectively. Apart from the number of parameters, our method has significantly lower computational time as it replaces all convolutions with fully connected layers. These results prove the efficiency of our model without compromising on performance.

4.5 SUMMARY

Inspired by the success of Transformers in Natural Language Processing and recently in image processing, we introduce a stand-alone self-attention model, Point Transformer, for efficient point cloud representation. We propose a novel recursive transformer, a learnable grouping operation, which makes our method more effective. Detailed experiments validate the effectiveness of our approach in shape classification and retrieval tasks. Further, we also focus on the robustness of our model on unseen input transformations, and it outperforms other state-of-the-art techniques. The model is also highly time and memory-efficient when compared to other methods. We hope to inspire future work to investigate the properties of point transformer networks and extend them for various 3-D point cloud processing tasks.

CHAPTER 5

SHAPE COMPLETION OF ALS ROOF POINT CLOUDS USING POINT COMPLETION TRANSFORMER

5.1 INTRODUCTION

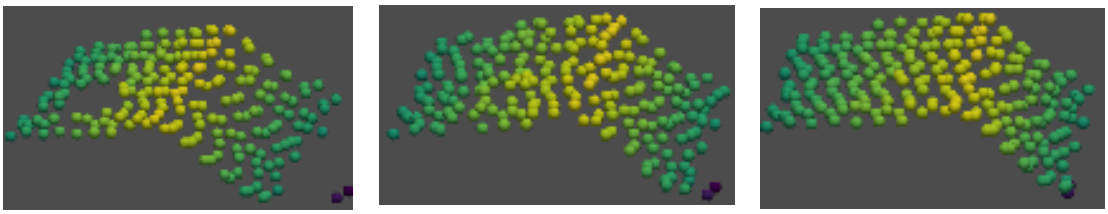


Figure 5.1: Shape completion of partial shape ALS roof point clouds.

The shape completion of ALS building roof point clouds is a challenging problem in urban modeling and a prerequisite for many real-world applications. This task is required to compensate for the structural, geometric, and semantic information loss due to several factors such as noise, occlusion, sensor issues and enhance the quality of incomplete scans of buildings. Shape completion of ALS roofs implies the estimation of the complete roof geometry from a partial input ALS roof point cloud, as illustrated in Figure 5.1. Recently, there have been a couple of research works [Xie *et al.* (2020); Yuan *et al.* (2018); Tchapmi *et al.* (2019); Yang *et al.* (2018)] which applied deep learning for 3-D shape completion using synthetic datasets like Completion3D [Tchapmi *et al.* (2019)], ShapeNet part dataset [Chang *et al.* (2015)]. However, the study focuses only on artificially created incomplete point clouds from aligned datasets. Therefore, there is a strong requirement for evaluating the performance of these models on naturally incomplete point clouds with other inherent corruptions. Additionally,

these methods introduce a loss of local geometrical information. For example: Given four distinctly unique chairs in the dataset, autoencoders tend to "average" these shapes and produce a common structure that can minimize loss against all the samples. To overcome this problem, PFNet predicts only the missing shape and fuses its output with the partial input to create the complete point cloud. However, the distribution of points in the predicted and partial shapes need not be similar and thus can lead to distortions.

In this work, we propose a fully-attentional network Point Completion Transformer which can maintain local structures present in the partial input point cloud and provides sturdy performance in real and synthetic datasets. Unlike PFNet, we propose reconstructing the complete point cloud using a locally guided encoder to capture vital local information. Further, these encoder representations guide the decoder, which can help reconstruct a complete shape coherent to the partial input shape. The proposed decoder reconstructs the complete point cloud at various resolutions, which helps incorporate hierarchies between the points. The key contributions of the proposed work are as follows:

- Fully attention-based deep learning method for shape completion of ALS roof point clouds.
- Novel local attention operation as an alternative for scalar dot product attention to capturing local geometric information.
- Multi-stage decoder used to hierarchically reconstruct complete point clouds at various resolutions.
- Chain of encoder-decoder connections to guide the decoder completion network based on the input partial point cloud and enables retention of local geometric information.

5.2 PRELIMINARIES

5.2.1 RoofN3D : Dataset for shape completion of ALS roof point clouds

The model's shape completion performance is evaluated on a large-scale urban dataset of ALS roof point clouds - RoofN3D [Wichmann *et al.* (2018)]. The dataset is uniformly split into a train set of 95,632, a validation set of 10,627, and test samples of size 11,814 for this study. The RoofN3D dataset was already detailed in Section 3.2.4 in Chapter 3. Partial point clouds are generated by randomly sampling a point and removing points around the given point within a specific radius. The generated partial point clouds are highly diverse - contains naturally existent missing shapes, simulated holes in roofs, partially removed superstructures, etc., and can be effectively used to evaluate shape completion methods.

5.2.2 Completion3D : Benchmark Dataset for 3-D point cloud shape completion

Figure 5.2: Partial Shape and Ground Truth Samples in Completion3D

Additionally, we also evaluate the model's performance on the Completion3D dataset, a sampled version of the original ShapeNet dataset containing eight different categories. Some samples in the Completion3D dataset are shown in Figure 5.2. It contains point clouds sampled to 2048 points and includes a diverse range of shapes and local geometries. The train/validation split is shown in Table 5.1 and the test split is unknown

and evaluated using the online leaderboard https://completion3d.stanford.edu/results.

Table 5.1: Completion3D data split

Class	Instances	Train	Validation
Airplane	3895	3795	100
Cabinet	1422	1322	100
Car	5777	5677	100
Chair	5850	5750	100
Lamp	2168	2068	100
Sofa	3023	2923	100
Table	5850	5750	100
Watercraft	1789	1689	100
Total	29774	28974	800

5.3 METHODOLOGY

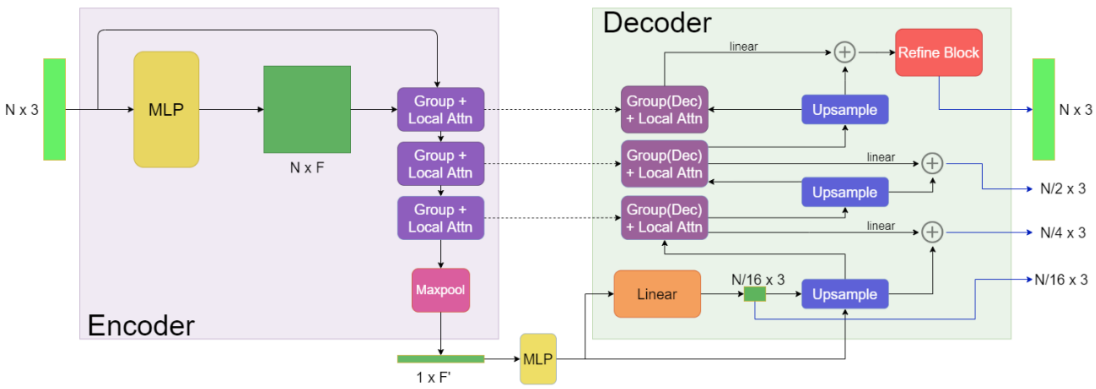


Figure 5.3: Architecture of Point Completion Transformer

The architecture of the Point Completion Transformer (PCT) is shown in Figure 5.3.

The following sections describe each component of the PCT network architecture.

5.3.1 Encoder

The encoder is used to derive a feature representation from the partial input point cloud. The partial input point cloud represented as $(N \times 3)$ is passed through two MLP layers, each followed by batch normalization and relu activation function. This builds a feature representation that is passed onto the stack of encoder blocks. Each encoder block consists of a local grouping layer followed by the local attention operation. After each encoder block, the point cloud is downsampled to capture multi-scale information. Additionally, the number of groups created in each block remains the same, which thereby increases the receptive field across the layers. This multi-scale, multi-resolution information is then max-pooled to derive a single global representation passed onto the decoder to generate the complete point cloud. Inspired by UNets [Ronneberger *et al.* (2015)], each level of the encoded representation is passed onto the decoder to guide the shape completion process. Each component in the encoder is explained in detail in the following subsections.

Local Grouping Layer

The core idea in the local grouping operation is to identify a discrete set of point centers in the point cloud and extract local features from the surrounding neighborhood around each center. This is done by first selecting N_1 centers from the partial input point cloud using Farthest Point Sampling (FPS), and for each point, we determine K nearest neighbors. This is used to index the incoming feature vectors at the current layer to create a representation of size $N_1 \times K \times F$. To capture local spatial information, it is concatenated with the distances between each center point and its K neighbors to create

a representation of size $N_1 \times K \times (F + 3)$. This is passed through an MLP layer to fuse spatial and encoded representations, which is then max-pooled along the K axis to create a rich feature vector of size $N_1 \times K \times F_1$. These two vectors are passed to the local attention block to further enhance the captured local geometric information. The architecture of the Local Grouping Layer used in the Encoder is shown in Figure 5.4.

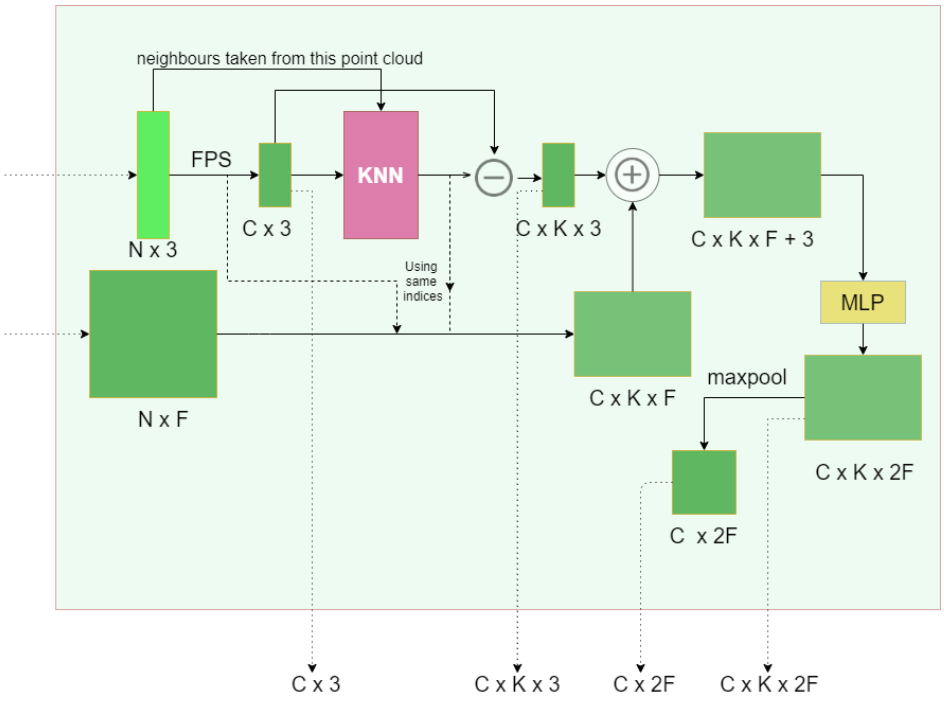


Figure 5.4: Architecture of the Local Grouping Layer

Local Attention Layer

The scalar dot product attention operation in Eqn.4.2 described in Section 4.3 in Chapter 4 is a global operation that computes relationships between the entire point set to produce an enhanced feature representation. This is not necessarily useful for the shape completion task as we need to retain local geometries present in the partial input shape

in the predicted complete point cloud. To do so, we modify the scalar dot product attention to compute relationships strictly within its local neighborhood to derive region-specific vectors. This is done by passing the derived center representations from the local grouping layers as Query (Q) vector while the corresponding neighborhood to derive the Key (K), Value vectors (V) vectors. To help the network adapt to the local structure, we supplement positional encodings (P) derived as the distance between each center point and the corresponding points in the neighborhood. Since the attention matrix (A) is relatively smaller ($K \times K$) when compared to the original scalar dot product attention ($N \times N$), we do not collapse the channel dimension while computing the attention weights enabling greater expressive power. The attention matrix is then passed through an MLP layer to further enhance it, which is used to aggregate the neighborhood information obtained via the value vectors. The overall operation can be summarised in equation Eqn.(5.1) where $f(.), \alpha(.), \beta(.), \gamma(.), \delta(.)$ are functions learnt by MLP layers.

$$\begin{aligned} P &= f(X_{neighbours} - X) \\ Q = \alpha(X_{center}), K = \beta(X_{neighbours}), V &= \gamma(X_{center}) \\ A &= \delta(Q - K + P) \\ O &= A \times (V + P) \end{aligned} \tag{5.1}$$

The architecure of the Local Attention Layer used in the Encoder is illustrated in Figure 5.5.

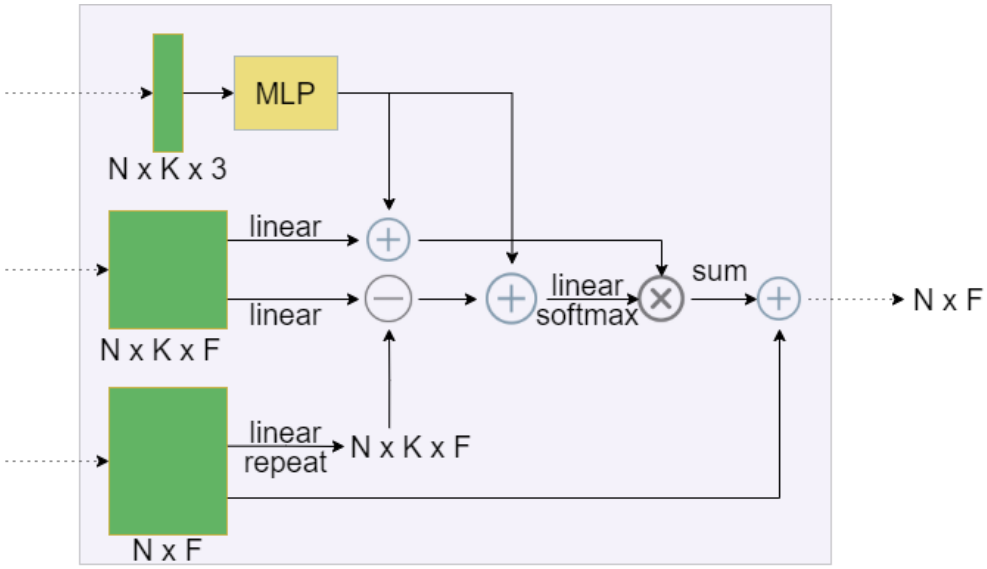


Figure 5.5: Architecture of the Local Attention Layer

As mentioned earlier, each encoder block consists of a local grouping layer followed by a local attention layer. The encoder contains multiple such blocks. The input representation is sequentially down-sampled before the final max-pooling operation to create a single shape descriptor passed onto the decoder for shape completion.

5.3.2 Decoder

Similar to the encoder, the decoder contains multiple stacks of decoder blocks, which take the encoded global shape vector to predict a complete point cloud. The core motivation in the design of the decoder is to initially create a sparse point cloud of the overall shape from the global shape vector, which is then upsampled to the final resolution guided by the encoder. At each up-sampling stage, we predict displacement vectors $(\delta x, \delta y, \delta z)$ with respect to the previous resolution, which enforces its characteristic hierarchical nature. The global representation obtained from the encoder contains information about the overall shape, which is then passed through

an MLP layer and reshaped to create the sparse point cloud. This is then passed onto multiple decoder blocks, containing an upsampling layer followed by a local grouping and local attention layer. The following subsections describe each component in detail.

Upsampling Layer

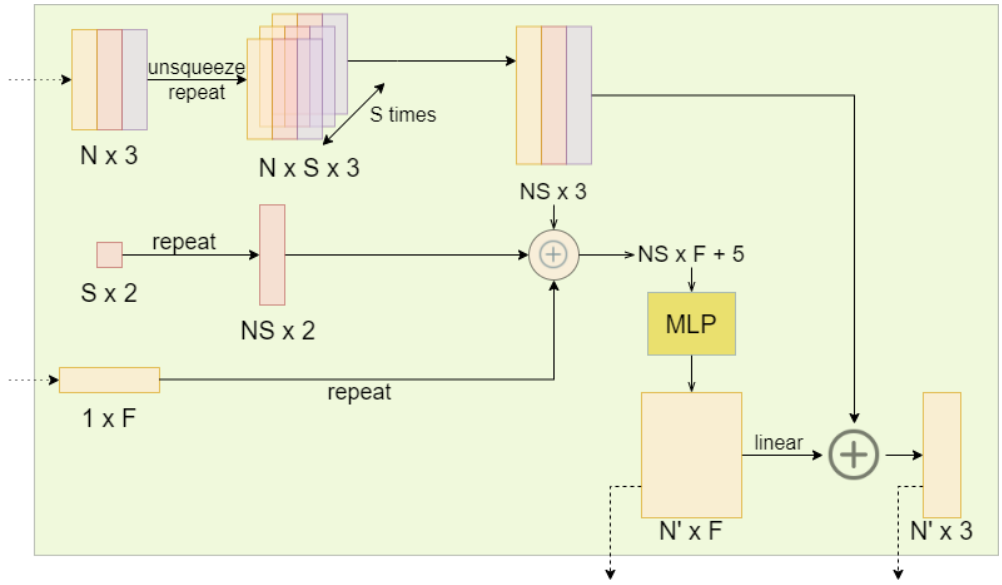


Figure 5.6: Architecture of the Upsampling Layer

Inspired by the PCN network, a simple folding operation is used for upsampling the points at each resolution. To do so, the global vector derived in the previous resolution is passed through an MLP layer and concatenated to set local features derived from the reconstructed points at the given decoder level. These point-wise feature vectors are then repeated M times (to upsample the point cloud by M) and concatenated with 2D grids sampled from a 2D plane. This feature representation is then passed onto an MLP layer to create displacement vectors for each point that is added to the reconstructed points in the previous resolution to create an upsampled point cloud. This is then passed

onto the local grouping and attention operation to encode the local geometries present in the partial input shape and create a high-quality complete point cloud. The architecture of the Upsampling Layer is shown in Figure 5.6.

Local Attention and Local Grouping Layer

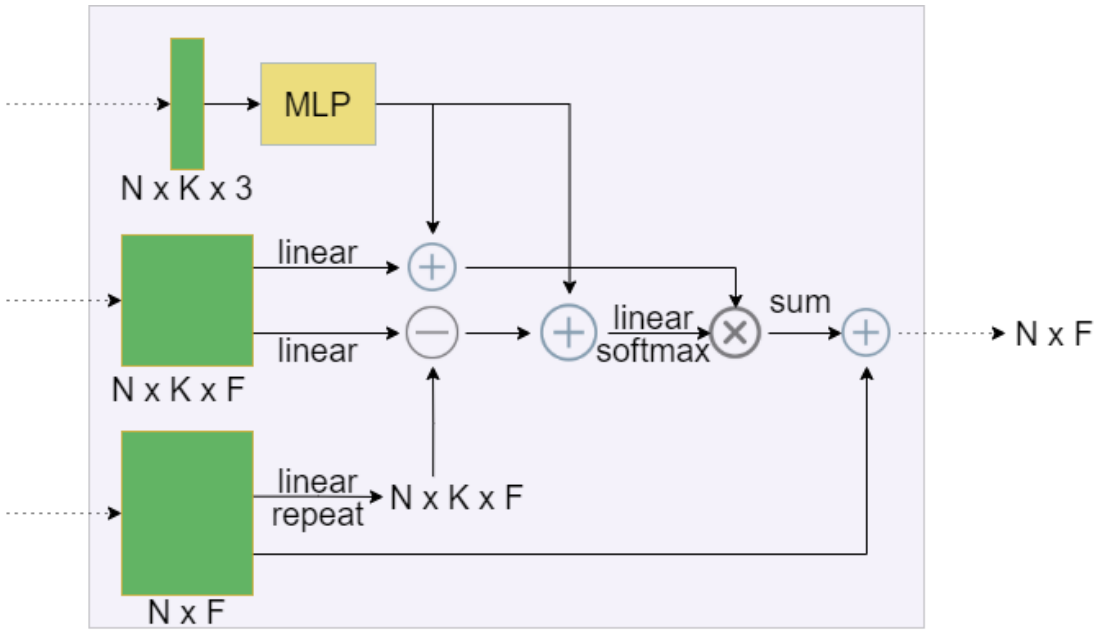


Figure 5.7: Architecture of the Local Attention Block in Decoder

The local attention and grouping layer are similar to the ones present in the encoder. The relationships are computed between the predicted complete shape and the encoded partial representation at the corresponding encoder level, unlike the encoder. These relationships provide local geometric cues from the partial input shape to guide the decoding process. This is done by simply considering the decoded complete point cloud at the given level as centers to the local neighborhood present in the encoded

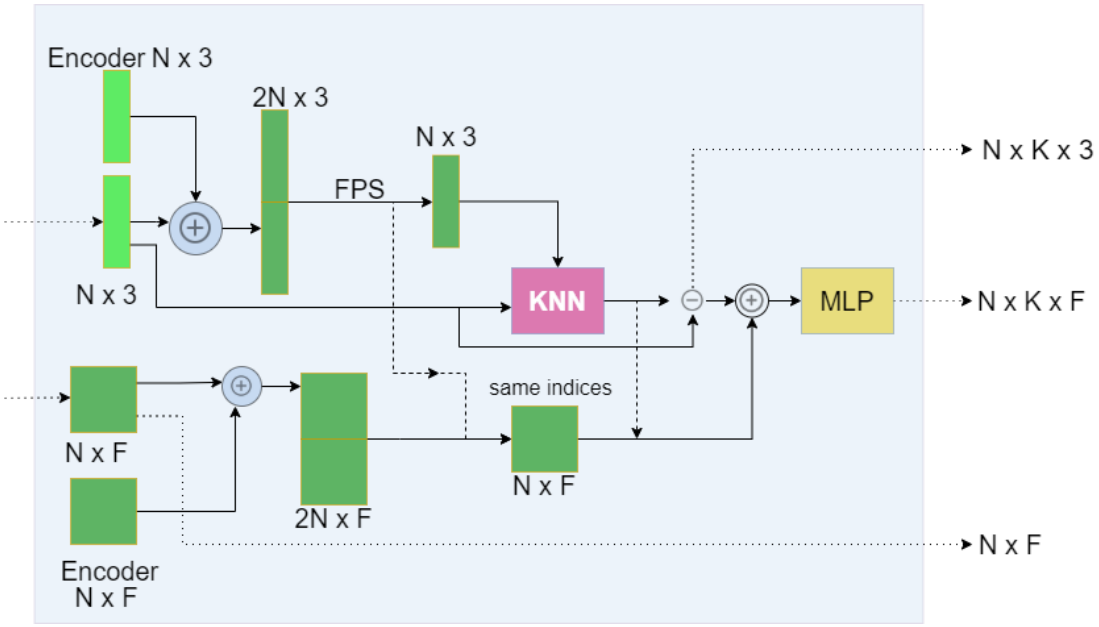


Figure 5.8: Architecture of the Local Grouping Layer in Decoder

partial shape representation. Since the decoder contains points that are not present in the encoder, the representations from the encoder and decoder (at the same level) are concatenated, and the FPS operation is performed to select a suitable set of point-wise representations covering the entire shape. This is used to derive the local neighborhood in the local grouping and attention operation. We refer to the subsection 5.3.1 for more details regarding each operation. This obtained enhanced representation is used to create displacement vectors which are then added to the incoming point cloud at that level to refine it. The architectures of the Local Attention Layer and Local Grouping Layer in the Decoder are illustrated in Figures 5.7 and 5.8.

As mentioned earlier, the decoder contains multiple stacks of decoder blocks, which hierarchically reconstruct the complete shape by increasing the resolution at each stage. Since the encoder-decoder strictly uses local geometric cues, there is a chance that some of these individual neighborhoods present in the completed shape are of different

distributions. Hence we pass this to a simple refinement network to uniformly distribute the points and create the final complete point cloud.

Refinement Network

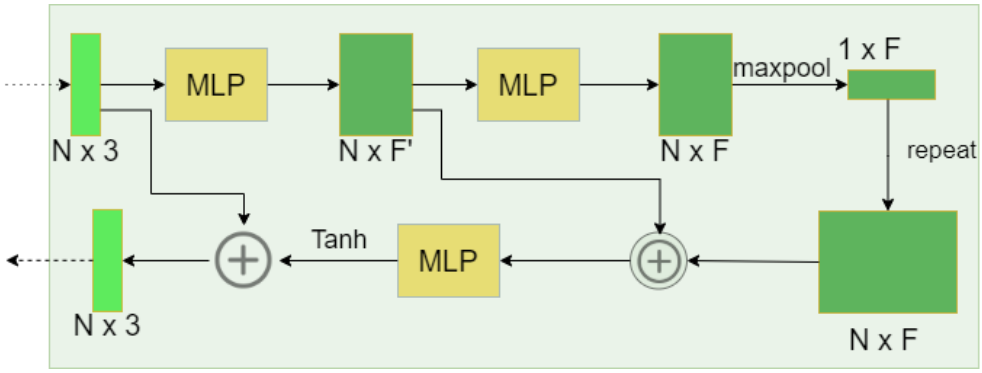


Figure 5.9: Architecture of the Refinement Network

The architecture of the Refinement Network in the decoder is shown in Figure 5.9. The refinement network consists of a stack of MLP layers used to uniformly redistribute the generated complete point cloud obtained from the decoder. The xyz coordinates of the generated points are passed through a set of MLP layers, then max-pooled to create a global representation. This global representation is concatenated with the output of the first MLP layer and then passed through another series of MLP layers to create a uniform point cloud.

The overall network is trained with an objective to minimize the chamfer distance between the predicted point cloud and ground truth complete shape. Additionally, we also penalize the generated points at each resolution to ensure higher shape completion quality.

5.3.3 Loss function : Chamfer Distance

The Chamfer Distance (CD) as shown in Eqn.(5.2) calculates the average closest point distance between the output point cloud S_1 and the ground truth point cloud S_2. The Chamfer distance loss is used to evaluate the quality of the reconstruction. For each point, the Chamfer distance (shown in Figure 5.10) finds the nearest neighbor in the other set and computes their squared distances, which are summed over both target and ground truth sets, which are shown in Eqn.(5.2). It is important to note that S_1 and S_2 need not be the same size to calculate CD.

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} min_{x \in S_1} \|x - y\|_2^2 \tag{5.2}$$

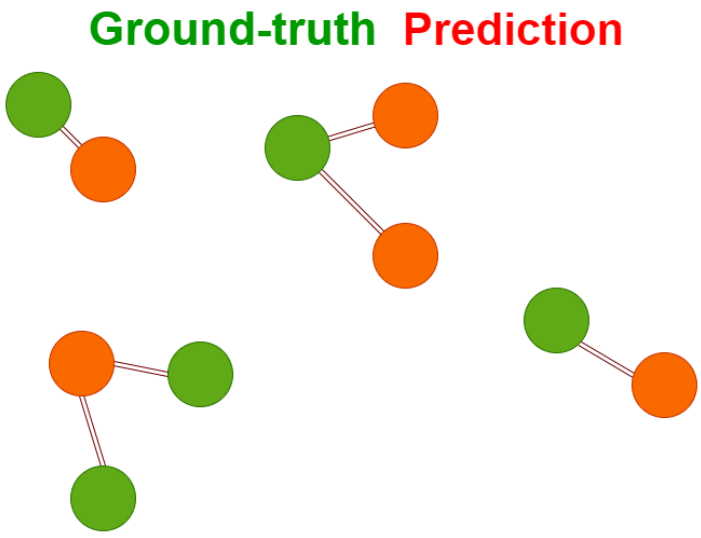


Figure 5.10: Chamfer Distance. The orange color points represent the reconstructed points while the green color points represent the ground truth

As mentioned above, we compute the chamfer distance at each resolution, summed together to compute the total loss. The chamfer distances at the lower resolutions are

scaled by a α factor. Therefore the total loss is given by Eqn.(5.3).

$$Loss = \alpha * CD_{resolution1} + 2 * \alpha * CD_{resolution2} + 4 * \alpha * CD_{resolution3} + CD_{finalresolution} \tag{5.3}$$

5.4 EXPERIMENTS AND RESULTS

5.4.1 Experiment Setting

The following hyper-parameters are used to train PCT for the shape completion task. The model derives vectors of size 64 from the embedding module, subsequently fed to the transformer block. These vectors are passed through grouping blocks which produce groups of 32 points, and the dimension of the hidden vectors is doubled at each pass. The global vector created is of 1×512 size. 64 points are then generated in the decoder, which is first quadrupled in the first pass. In the subsequent passes, the number of points is doubled. The network is trained using the Adam optimizer with a batch size of 12 and an initial learning rate of 0.0001. The learning rate is reduced by a factor of 0.7 every 40 epochs, and the model converges in less than 100 epochs. For the first 25 epochs $\alpha = 0.025$, which is then increased to 0.05 and finally after 50 epochs are increased to 0.1. All experiments are run on an NVIDIA GTX 2080Ti GPU.

As seen in Figure 5.11, the train loss has converged after 600,000 iterations. As seen in the other Figure 5.12, the train and validation chamfer distance values are very close to each other, implying that the model is neither underfitting nor overfitting.

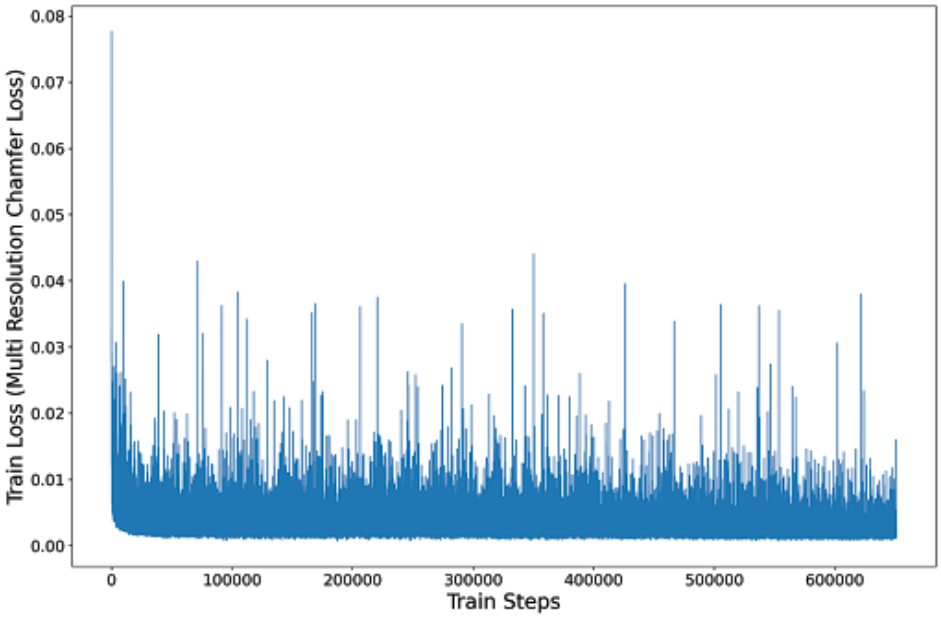


Figure 5.11: Training Loss Vs Training Steps

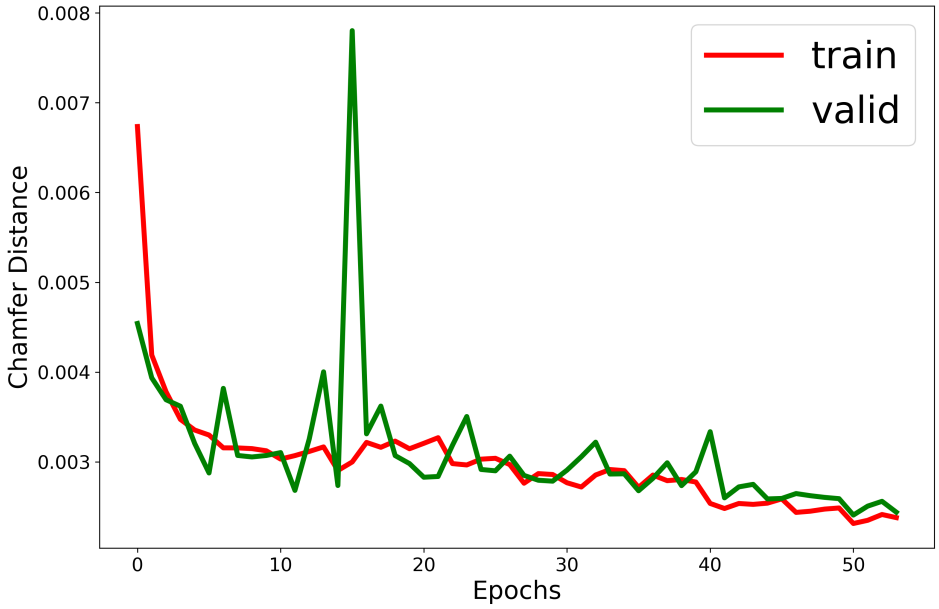


Figure 5.12: Chamfer Loss Vs Epochs

5.4.2 Results of Shape Completion in Completion 3D

For shape completion in RoofN3D, we select 1024 points. As mentioned earlier, partial point clouds are created by randomly sampling a point and removing its local neighborhood within a given radius. In Completion3D, we use the provided partial and complete shapes to train and evaluate our model. Table 5.2 describes results in the benchmark dataset Completion3D.

Table 5.2: Results of Shape completion in Completion3D

Method	**Chamfer Distance** (10^{-4})
VRCNet	8.12
PMP-Net	9.23
GRNet Xie *et al.* (2020)	10.64
TopNet Tchapmi *et al.* (2019)	14.25
PCN Yuan *et al.* (2018)	18.22
FoldingNet Yang *et al.* (2018)	19.07
PCT (Ours)	**12.33**

We want to point out that the given metric - chamfer distance is not ideal to evaluate the quality of reconstruction. Chamfer distance does not penalize the model based on its ability to retain local geometric information, and it is sufficient to predict the overall geometry. We strongly believe that this is why our model does not beat state-of-the-art results while still capturing vital local information. We provide visual evidence to show that the proposed method retains better local geometry when compared to other methods. Therefore, there is a solid requirement to introduce better evaluation metrics to accurately compare the reconstructed point cloud's completeness. However, in this work, we stick to benchmark evaluation metrics and leave this for future work.

As seen from the Figures 5.13 - 5.17 we can see that PCT retains detailed local geometric information in the predicted completed point cloud. Generally, most methods tend to reconstruct noisy points to the averaging property of the network and chamfer loss. However, since PCT strictly derives representations from local neighborhoods, only the model does not reconstruct any form of noise (as seen between the chair legs in Figure 5.15. Due to similar reasons, most methods also tend to reconstruct thin lines, planes, or corners poorly, and once again, PCT predicts samples with much higher visual quality.

Partial shape | Ground Truth | Completed shape

Figure 5.13: Shape completion of a stool in Completion3D

Partial shape | Ground Truth | Completed shape

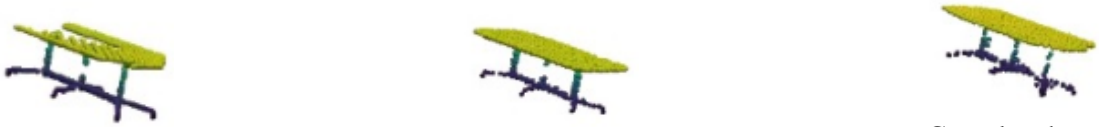
Figure 5.14: Shape completion of a boat in Completion3D

Partial shape | Ground Truth | Completed shape

Figure 5.15: Shape completion of a chair in Completion3D

Partial shape | Ground Truth | Completed shape

Figure 5.16: Shape completion of a car in Completion3D



Partial shape | Ground Truth | Completed shape

Figure 5.17: Shape completion of a table in Completion3D

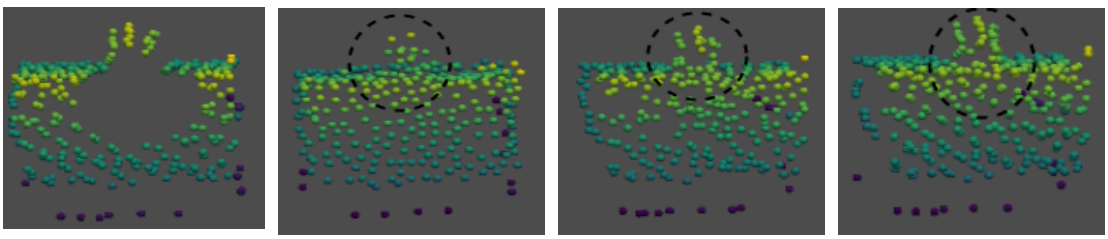
5.4.3 Results of Shape Completion in RoofN3D

The results of shape completion in the ALS roof point cloud dataset RoofN3D are shown in Table 5.3. We can see a large margin of difference between PCN and PCT, which suggests the ability of our model to generalize to real datasets containing various imperfections. As mentioned above, most methods fail to capture planes, corners which are major components in a roof shape. This could be the reason for the more significant difference in performance between PCN and PCT in the RoofN3D dataset.

Table 5.3: Results of Shape completion in RoofN3D

Method	**Chamfer Distance**(10^{-4})
PCN	36.9
PCT (Ours)	**24**

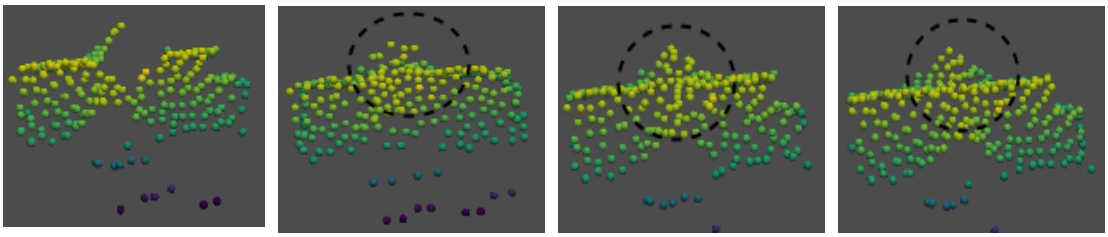
The visual results of shape completion in ALS roof point clouds are illustrated in Figures 5.18 - 5.25. Figure 5.18 shows the reconstruction of a roof point cloud with partial input shape containing superstructures (marked inside black color circle), and PCT clearly shows superior reconstruction quality.



Partial Shape PCN PCT Ground Truth

Figure 5.18: Shape completion of ALS roof point cloud with missing points near projections

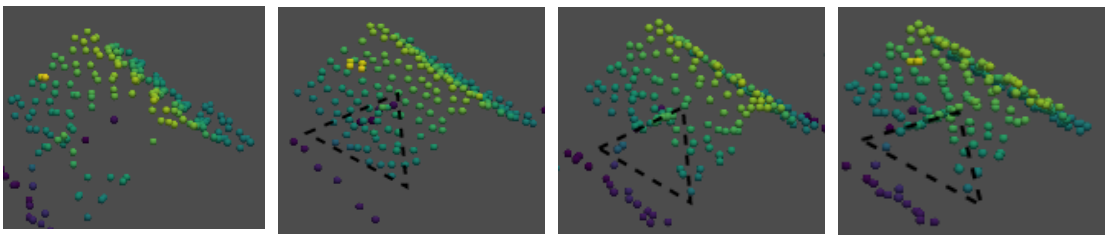
In Figure 5.19, we can see that PCT can retain existing superstructures (marked inside black color circle) and also complete partial ones meaningfully, unlike PCN, where the existing superstructure is completely missing in the predicted shape and the partial superstructure is predicted with lots of noise.



Partial Shape PCN PCT Ground Truth

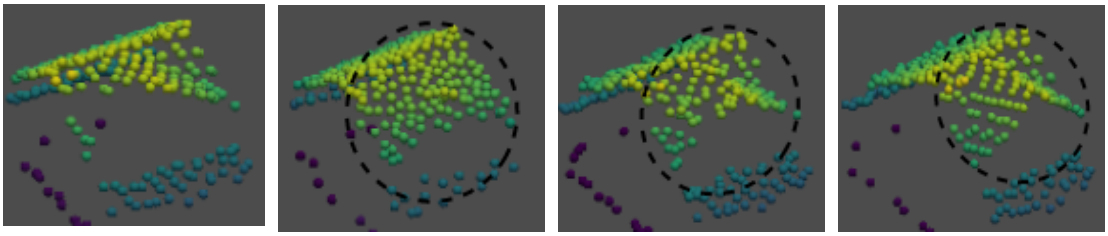
Figure 5.19: Shape completion of ALS Roof Point cloud with missing substructures

Figures 5.20 - 5.25 show few more examples where PCT can reconstruct complex roof structures which are more coherent to the ground truth when compared to PCN even with the presence of noise, outliers, and other imperfections in the partial input point cloud. All these results clearly suggest the superiority of the proposed method.



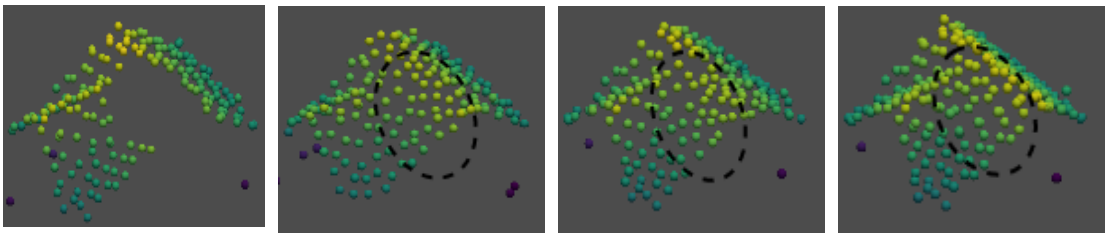
Partial Shape PCN PCT Ground Truth

Figure 5.20: Shape completion of ALS roof point cloud with missing points near sub structure



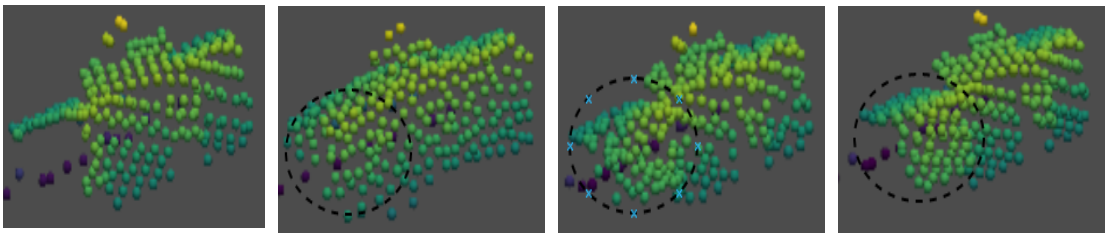
Partial Shape PCN PCT Ground Truth

Figure 5.21: Shape completion of ALS roof point cloud with missing points in sub structure



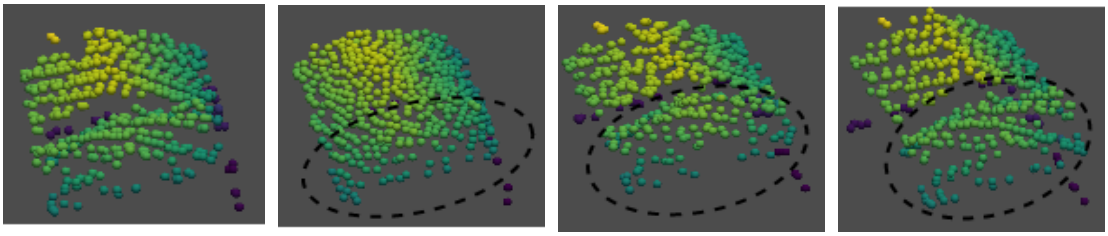
Partial Shape PCN PCT Ground Truth

Figure 5.22: Shape completion of ALS roof point cloud with missing points in joints of different sub structures



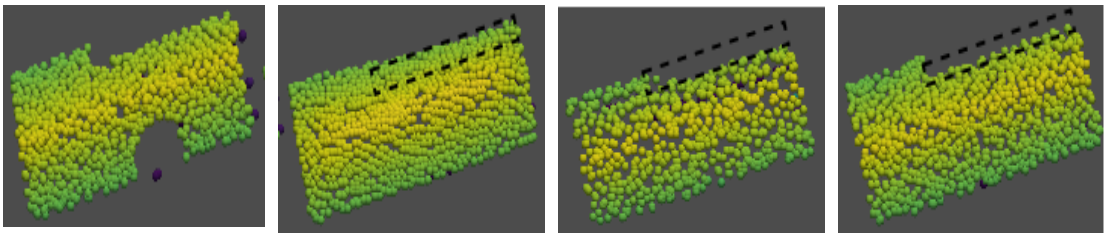
Partial Shape PCN PCT Ground Truth

Figure 5.23: Shape completion of ALS roof point cloud with missing points in sub structure at different level



Partial Shape PCN PCT Ground Truth

Figure 5.24: Shape completion of ALS roof point cloud with missing points in sub structure at different level



Partial Shape PCN PCT Ground Truth

Figure 5.25: Shape completion of ALS roof point cloud with different edge widths

5.4.4 Model Parameters, Size and Forward Pass time

In Table 5.4, the model parameters, size, and forward inference speeds of PCT are compared with other methods. These values prove that our model is very competitive in memory requirements and speed while still achieving high visual quality in shape completion.

Table 5.4: Model Parameters, Size and Forward Pass Time

Method	**Params (M)**	**Size (MB)**	**Time (ms)**
Folding	2.4	-	-
PCN	6.85	21	1.5
TopNet	9.96	79.8	-
PCT (Ours)	**8.66**	**34.7**	**38**

5.5 SUMMARY

We have proposed a full self-attention model for efficient point cloud shape completion. A novel local attention operation is introduced as an alternative for scalar dot product attention to capturing local geometric information. A multi-stage decoder is used to reconstruct complete point clouds at various resolutions hierarchically. The encoder representations further guide the decoder to generate a complete point cloud coherent to the partial input shape. While our model does not beat state-of-the-art methods in chamfer distance, we show strong evidence that the visual quality of reconstruction is much higher. This indicates that better evaluation metrics need to be introduced, which we leave for future work. Additionally, our model can work well on both real and synthetic datasets while being memory and time-efficient.

CHAPTER 6

CONTRIBUTIONS, FUTURE WORK AND CONCLUSION

6.1 CONTRIBUTIONS

This thesis was perhaps the foremost research for applying deep learning-based methods for deriving efficient representation of ALS roof point clouds. Our research contributed innovative or unconventional deep learning architectures for processing ALS roof point clouds and effectively showed high performance and generalization capacity across various tasks like classification, retrieval, and shape completion. The proposed methods were also evaluated on benchmark synthetic datasets and showcased competitive performance and improved robustness.

Specifically, our research contributions can be summarized as follows:

1. MVCNN-SA
    - First work to study the effectiveness of deep learning methods for ALS roof point cloud processing.
    - Introduced an adaptive view pooling method based on the relative importance of each view to efficiently fuse information.
2. Point Transformer
    - Novel approach to show self-attention was a natural fit for point cloud processing.
    - The proposed iterative transformer (with weight-sharing) can hierarchically learn complex information with a significantly lesser number of parameters.
    - A learnable grouping mechanism with a routing loss was introduced to group semantically similar points and derive region/group-wise feature vectors.
    - Focus on the model's robustness to unseen corruptions apart from classification performance on benchmark datasets.
    - Our specific design considerations ensure that the model is highly time and memory-efficient, making it ideal for real-time use cases.

3. Point Completion Transformer
    - Fully attention-based deep learning method for shape completion of ALS roof point clouds.
    - Novel local attention operation as an alternative for scalar dot product attention to capturing local geometric information.
    - Multi-stage decoder used to hierarchically reconstruct complete point clouds at various resolutions.
    - Chain of encoder-decoder connections to guide the decoder completion network based on the input partial point cloud and enables retention of local geometric information.

The significant results of the research are highlighted below:

1. The proposed view-based method MVCNN-SA achieved 1.2% higher accuracy while the point-based method Point Transformer achieved 1.26% higher accuracy in the roof-classification task when compared to existing methods. In the roof-retrieval task, MVCNN-SA achieved 6.14 higher MAP scores while Point Transformer achieved 6.54 higher MAP scores when compared to existing architectures. In benchmark datasets like ModelNet-40, Point Transformer achieved 92.5% accuracy in the shape classification task, 88.4 MAP in the retrieval task, and performed better than SOTA graph-based and other attention-based methods.

2. Detailed robustness experiments showed that Point Transformer outperformed existing methods by 3.57% in average accuracy across various corruptions in roof classification while maintaining a 90.77 MAP score in partial shape roof retrieval. In RobustPointSet, Point Transformer stood as the best performing method achieving 62.55% average accuracy on six unseen point corruptions. Additionally, Point Transformer was highly memory, time-efficient requiring only $1/3^{rd}$ of the total number of parameters as PointNet and $1/100^{th}$ of view-based methods like MVCNN helping achieve a faster forward inference time of 10ms per sample.

3. In the roof-shape completion task, Point Completion Transformer showcased a better performance by achieving 24×10^{-4} chamfer distance and maintained high visual quality. In benchmark dataset Completion3D, Point Completion Transformer achieved 12×10^{-4} chamfer distance and maintained detailed local geometric information.

6.2 FUTURE WORK

- In this thesis, we explored attention - as a complimentary sub-layer or as a core operation in a fully-attention network in three tasks - classification, retrieval, and shape completion. However, its complete effectiveness was not yet discovered for various other tasks - like segmentation, noise removal, etc.
- Additionally, this method can also be extended for larger point sets - aerial city scene, driving lidar scans, etc., for tasks applicable for autonomous vehicles.
- Our proposed methods were seen to be inherently robust to various unseen transformations, and this can be further improved, especially in the case of rotation, by introducing an orientation invariant input representation (like distance and polar angles).
- One of the major bottlenecks in attention is the scalar dot product operation which has an $O(N^2)$ time complexity, and reducing the same to lower orders like $O(N)$ is still an open research problem.

6.3 CONCLUSION

We have successfully shown that deep learning methods can be devised for effective ALS roof point cloud representation. Specifically, we choose three tasks - classification, retrieval, and shape completion and the proposed methods achieve high performance while showing increased robustness to unseen transformations. Attention is a core idea of the proposed methods, and to the best of our knowledge, its suitability for point cloud representation is first explored in this thesis. We hope to inspire future works in this direction and motivate other researchers to utilize these methods for other tasks like segmentation, scene understanding, noise removal, etc.

REFERENCES

1. **Ahmed, E.**, **A. Saint**, **A. E. R. Shabayek**, **K. Cherenkova**, **R. Das**, **G. Gusev**, **D. Aouada**, and **B. E. Ottersten** (2018). Deep learning advances on different 3-D data representations: A survey. *CoRR*, **abs/1808.01462**. URL http://arxiv.org/abs/1808.01462.

2. **Akgul, C. B.**, **B. Sankur**, **Y. Yemez**, and **F. Schmitt** (2009). 3-D model retrieval using probability density-based shape descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **31**(6), 1117–1133.

3. **Alexa, M.**, **J. Behr**, **D. Cohen-Or**, **S. Fleishman**, **D. Levin**, and **C. Silva** (2003). Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, **9**(1), 3–15.

4. **Alidoost, F.** and **H. Arefi** (2016). Knowledge based 3-D building model recognition using convolutional neural networks from lidar and aerial imageries. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, **XLI-B3**, 833–840.

5. **Anguelov, D.**, **P. Srinivasan**, **D. Koller**, **S. Thrun**, **J. Rodgers**, and **J. Davis** (2005). Scape: Shape completion and animation of people. *ACM Trans. Graph.*, **24**(3), 408–416. ISSN 0730-0301. URL https://doi.org/10.1145/1073204.1073207.

6. **Awrangjeb, M.**, **S. A. N. Gilani**, and **F. U. Siddiqui** (2018). An effective data-driven method for 3-d building roof reconstruction and robust change detection. *Remote Sensing*, **10**(10). ISSN 2072-4292. URL https://www.mdpi.com/2072-4292/10/10/1512.

7. **Axelsson, M.**, **U. Soderman**, **A. Berg**, and **T. Lithen**, Roof type classification using deep convolutional neural networks on low resolution photogrammetric point clouds from aerial imagery. *In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, 1293–1297.

8. **Bello, S. A.**, **S. Yu**, **C. Wang**, **J. M. Adam**, and **J. Li** (2020). Review: Deep learning on 3-D point clouds. *Remote Sensing*, **12**(11). ISSN 2072-4292. URL https://www.mdpi.com/2072-4292/12/11/1729.

9. **Blender Online Community** (). *Blender - a 3-D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam. URL http://www.blender.org.

10. **Boscaini, D.**, **J. Masci**, **E. Rodolà**, and **M. Bronstein**, Learning shape correspondence with anisotropic convolutional neural networks. *In* **D. Lee**, **M. Sugiyama**, **U. Luxburg**, **I. Guyon**, and **R. Garnett** (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016, 3197–3205. URL https://proceedings.neurips.cc/paper/2016/file/228499b55310264a8ea0e27b6e7c6ab6-Paper.pdf.

11. **Castagno, J.** and **E. Atkins** (2018). Roof shape classification from lidar and satellite image data fusion using supervised learning. *Sensors*, **18**, 3960.

12. **Chang, A. X.**, **T. Funkhouser**, **L. Guibas**, **P. Hanrahan**, **Q. Huang**, **Z. Li**, **S. Savarese**, **M. Savva**, **S. Song**, **H. Su**, **J. Xiao**, **L. Yi**, and **F. Yu** (2015). ShapeNet: An Information-Rich 3-D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University - Princeton University - Toyota Technological Institute at Chicago.

13. **Charles, R. Q.**, **H. Su**, **M. Kaichun**, and **L. J. Guibas**, Pointnet: Deep learning on point sets for 3-D classification and segmentation. *In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, 77–85.

14. **Chen, D.**, **R. Wang**, and **J. Peethambaran** (2017). Topologically aware building rooftop reconstruction from airborne laser scanning point clouds. *IEEE Transactions on Geoscience and Remote Sensing*, **55**(12), 7032–7052.

15. **Chen, D.**, **L. Zhang**, **P. T. Mathiopoulos**, and **X. Huang** (2014*a*). A methodology for automated segmentation and reconstruction of urban 3-d buildings from als point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **7**(10), 4199–4217.

16. **Chen, J.-Y.**, **C.-H. Lin**, **P.-C. Hsu**, and **C.-H. Chen** (2014*b*). Point cloud encoding for 3-D building model retrieval. *IEEE Transactions on Multimedia*, **16**(2), 337–345.

17. **Chen, Y.-C.**, **B.-Y. Lin**, and **C.-H. Lin** (2017). Consistent roof geometry encoding for 3d building model retrieval using airborne lidar point clouds. *ISPRS International Journal of Geo-Information*, **6**(9), 269.

18. **Chen, Y.-C.** and **C.-H. Lin** (2016). Image-based airborne lidar point cloud encoding for 3-D building model retrieval. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, **XLI-B8**, 1237–1242. URL `https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLI-B8/1237/2016/`.

19. **Cheng, G.**, **P. Zhou**, and **J. Han** (2016). Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images. *IEEE Trans. Geoscience and Remote Sensing*, **54**(12), 7405–7415. URL `https://doi.org/10.1109/TGRS.2016.2601622`.

20. **Dai, A.**, **C. R. Qi**, and **M. Nießner**, Shape completion using 3-D-encoder-predictor cnns and shape synthesis. *In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, 6545–6554. URL `https://doi.org/10.1109/CVPR.2017.693`.

21. **Dai, A.**, **D. Ritchie**, **M. Bokeloh**, **S. Reed**, **J. Sturm**, and **M. Nießner**, Scancomplete: Large-scale scene completion and semantic segmentation for 3-D scans. *In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018, 4578–4587. URL `http://openaccess.thecvf.com/content_cvpr_2018/html/Dai_ScanComplete_Large-Scale_Scene_CVPR_2018_paper.html`.

22. **Dehbi, Y.**, **S. Koppers**, and **L. Plümer** (2021). Looking for a needle in a haystack: Probability density based classification and reconstruction of dormers from 3-D point clouds. *Transactions in GIS*, **25**(1), 44–70.

23. **Deng, J.**, **W. Dong**, **R. Socher**, **L.-J. Li**, **K. Li**, and **L. Fei-Fei**, Imagenet: A large-scale hierarchical image database. *In 2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, 248–255.

24. **Dorninger, P.** and **N. Pfeifer** (2008). A comprehensive automated 3-D approach for building extraction, reconstruction, and regularization from airborne laser scanning point clouds. *Sensors (Basel, Switzerland)*, **8**, 7323 – 7343.

25. **Engel, J.**, **T. Schöps**, and **D. Cremers**, Lsd-slam: Large-scale direct monocular slam. *In* **D. Fleet**, **T. Pajdla**, **B. Schiele**, and **T. Tuytelaars** (eds.), *Computer Vision – ECCV 2014*. Springer International Publishing, Cham, 2014, 834–849.

26. **Feng, Y.**, **Z. Zhang**, **X. Zhao**, **R. Ji**, and **Y. Gao**, Gvcnn: Group-view convolutional neural networks for 3-D shape recognition. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, 264–272.

27. **Fischler, M. A.** and **R. C. Bolles** (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, **24**(6), 381–395. ISSN 0001-0782. URL https://doi.org/10.1145/358669.358692.

28. **Gkeli, M.** and **C. Ioannidis** (2018). Automatic 3-D reconstruction of buildings roof tops in densely urbanized areas. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, **XLII-4/W10**, 47–54. URL https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-4-W10/47/2018/.

29. **Guo, Y.**, **H. Wang**, **Q. Hu**, **H. Liu**, **L. Liu**, and **M. Bennamoun** (2020). Deep learning for 3-D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1.

30. **Guptha, S.** and **R. Bohare** (2019). Roof classification, segmentation, and damage completion using 3-D point clouds. https://github.com/sarthakTUM/roofn3d.

31. **Haala, N.** and **M. Kada** (2010). An update on automatic 3-D building reconstruction. *ISPRS Journal of Photogrammetry and Remote Sensing*, **65**(6), 570–580. ISSN 0924-2716. URL https://www.sciencedirect.com/science/article/pii/S0924271610000894. ISPRS Centenary Celebration Issue.

32. **Han, J.**, **D. Zhang**, **X. Hu**, **L. Guo**, **J. Ren**, and **F. Wu** (2015). Background prior-based salient object detection via deep reconstruction residual. *IEEE Trans. Circuits Syst. Video Techn.*, **25**(8), 1309–1321. URL https://doi.org/10.1109/TCSVT.2014.2381471.

33. **Han, X.**, **Z. Li**, **H. Huang**, **E. Kalogerakis**, and **Y. Yu**, High-resolution shape completion using deep neural networks for global structure and local geometry inference. *In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, 85–93. URL https://doi.org/10.1109/ICCV.2017.19.

34. **He, K.**, **X. Zhang**, **S. Ren**, and **J. Sun**, Deep residual learning for image recognition. *In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, 770–778.

35. **He, X.**, **Y. Zhou**, **Z. Zhou**, **S. Bai**, and **X. Bai**, Triplet-center loss for multi-view 3-D object retrieval. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, 1945–1954.

36. **Henn, A.**, **G. Gröger**, **V. Stroh**, and **L. Plümer** (2013). Model driven reconstruction of roofs from sparse LIDAR point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, **76**, 17–29.

37. **Hou, J.**, **A. Dai**, and **M. Nießner**, 3-D-sis: 3-D semantic instance segmentation of RGB-D scans. *In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, 4421–4430. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Hou_{3D}-SIS_3D_Semantic_Instance_Segmentation_of_RGB-D_Scans_CVPR_2019_paper.html.

38. **Huang, Z.**, **Y. Yu**, **J. Xu**, **F. Ni**, and **X. Le**, Pf-net: Point fractal network for 3-D point cloud completion. *In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, 2020, 7659–7667. URL https://doi.org/10.1109/CVPR42600.2020.00768.

39. **Jiang, J.**, **D. Bao**, **Z. Chen**, **X. Zhao**, and **Y. Gao**, MLVCNN: multi-loop-view convolutional neural network for 3-D shape retrieval. *In The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019, 8513–8520. URL https://doi.org/10.1609/aaai.v33i01.33018513.

40. **Jung, J.**, **Y. Jwa**, and **G. Sohn** (2017). Implicit regularization for reconstructing 3-D building rooftop models using airborne lidar data. *Sensors*, **17**(3). ISSN 1424-8220. URL https://www.mdpi.com/1424-8220/17/3/621.

41. **Kada, M.** and **A. Wichmann** (2012). Sub-surface growing and boundary generalization for 3-D building reconstruction. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, **I-3**, 233–238. URL https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/I-3/233/2012/.

42. **Kanezaki, A.** (2016). Rotationnet: Learning object classification using unsupervised viewpoint estimation. *CoRR*, **abs/1603.06208**. URL http://arxiv.org/abs/1603.06208.

43. **Lafarge, F.** and **C. Mallet**, Building large urban environments from unstructured point data. *In 2011 International Conference on Computer Vision*. 2011, 1068–1075.

44. **Lafarge, F.** and **C. Mallet** (2012). Creating large-scale city models from 3-D-point clouds: A robust approach with hybrid representation. *International Journal of Computer Vision*, **99**, 69–85.

45. **Lee, J.**, **Y. Lee**, **J. Kim**, **A. R. Kosiorek**, **S. Choi**, and **Y. W. Teh** (2019). Pytorch implementation of set transformers. https://github.com/juho-lee/set_transformer/issues/3.

46. **Li, J.**, **B. M. Chen**, and **G. H. Lee**, So-net: Self-organizing network for point cloud analysis. *In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018*a*, 9397–9406. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Li_SO-Net_Self-Organizing_Network_CVPR_2018_paper.html.

47. **Li, Y.**, **R. Bu**, **M. Sun**, **W. Wu**, **X. Di**, and **B. Chen**, Pointcnn: Convolution on transformed points. *In Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18. Curran Associates Inc., Red Hook, NY, USA, 2018*b*, 828–838.

48. **Liu, T.**, **J. Gao**, and **Y. Zhao** (2018). An approach to 3-D building model retrieval based on topology structure and view feature. *IEEE Access*, **6**, 31685–31694.

49. **Liu, X.**, **Z. Han**, **Y. Liu**, and **M. Zwicker**, Point2sequence: Learning the shape representation of 3-D point clouds with an attention-based sequence to sequence network. *In The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019*a*, 8778–8785. URL https://doi.org/10.1609/aaai.v33i01.33018778.

50. **Liu, X.**, **Z. Han**, **X. Wen**, **Y. Liu**, and **M. Zwicker**, L2G auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention. *In* **L. Amsaleg**, **B. Huet**, **M. A. Larson**, **G. Gravier**, **H. Hung**, **C. Ngo**, and **W. T. Ooi** (eds.), *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*. ACM, 2019*b*, 989–997. URL https://doi.org/10.1145/3343031.3350960.

51. **Liu, Y.**, **B. Fan**, **G. Meng**, **J. Lu**, **S. Xiang**, and **C. Pan**, Densepoint: Learning densely contextual representation for efficient point cloud processing. *In 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South),*

October 27 - November 2, 2019. IEEE, 2019*c*, 5238–5247. URL https://doi.org/10.1109/ICCV.2019.00534.

52. **Lu, H.** and **H. Shi** (2021). Deep learning for 3-D point cloud understanding: A survey.

53. **Makantasis, K.**, **K. Karantzalos**, **A. Doulamis**, and **K. Loupos**, Deep learning-based man-made object detection from hyperspectral data. volume 9474. 2015. ISBN 978-3-319-27856-8, 717–727.

54. **Maturana, D.** and **S. Scherer**, Voxnet: A 3-D convolutional neural network for real-time object recognition. *In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, 922–928.

55. **Mohajeri, N.**, **D. Assouline**, **B. Guiboud**, **A. Bill**, **A. Gudmundsson**, and **J.-L. Scartezzini** (2018). A city-scale roof shape classification using machine learning for solar energy applications. *Renewable Energy*, **121**, 81–93. ISSN 0960-1481. URL https://www.sciencedirect.com/science/article/pii/S0960148117313009.

56. **Musialski, P.**, **P. Wonka**, **D. G. Aliaga**, **M. Wimmer**, **L. Gool**, and **W. Purgathofer** (2013). A survey of urban reconstruction. *Comput. Graph. Forum*, **32**(6), 146–177. ISSN 0167-7055. URL https://doi.org/10.1111/cgf.12077.

57. **Omidalizarandi, M.** and **M. Saadatseresht** (2013). Segmentation and classification of point clouds from dense aerial image matching. *The International Journal of Multimedia & Its Applications*, **5**, 33–51.

58. **Park, J. J.**, **P. Florence**, **J. Straub**, **R. A. Newcombe**, and **S. Lovegrove**, Deepsdf: Learning continuous signed distance functions for shape representation. *In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, 165–174. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Park_DeepSDF_Learning_Continuous_Signed_Distance_Functions_for_Shape_Representation_CVPR_2019_paper.html.

59. **Phong, B. T.** (1975). Illumination for computer generated pictures. *Commun. ACM*, **18**(6), 311–317. ISSN 0001-0782. URL http://doi.acm.org/10.1145/360825.360839.

60. **Poullis, C.** and **S. You**, Automatic reconstruction of cities from remote sensor data. *In 2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, 2775–2782.

61. **Qi, C. R.**, **H. Su**, **M. Nießner**, **A. Dai**, **M. Yan**, and **L. J. Guibas**, Volumetric and multi-view cnns for object classification on 3-D data. *In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, 5648–5656. URL https://doi.org/10.1109/CVPR.2016.609.

62. **Qi, C. R.**, **L. Yi**, **H. Su**, and **L. J. Guibas**, Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *In Proceedings of the 31st International*

Conference on Neural Information Processing Systems, NIPS'17. Curran Associates Inc., Red Hook, NY, USA, 2017. ISBN 9781510860964, 5105–5114.

63. **Remondino, F.** (2011). Heritage recording and 3-D modeling with photogrammetry and 3-D scanning. *Remote Sensing*, **3**(6), 1104–1138. ISSN 2072-4292. URL https://www.mdpi.com/2072-4292/3/6/1104.

64. **Ronneberger, O.**, **P. Fischer**, and **T. Brox**, U-net: Convolutional networks for biomedical image segmentation. *In* **N. Navab**, **J. Hornegger**, **W. M. Wells**, and **A. F. Frangi** (eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Springer International Publishing, Cham, 2015. ISBN 978-3-319-24574-4, 234–241.

65. **Rottensteiner, F.**, **J. T. B**, **S. C. C**, and **K. K. C** (). Automated delineation of roof planes from lidar data.

66. **Sampath, A.** and **J. Shan** (2010). Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds. *IEEE Transactions on Geoscience and Remote Sensing*, **48**(3), 1554–1567.

67. **Shajahan, D. A.**, **V. Nayel**, and **R. Muthuganapathy** (2020). Roof classification from 3-D lidar point clouds using multiview cnn with self-attention. *IEEE Geoscience and Remote Sensing Letters*, **17**(8), 1465–1469.

68. **Shen, X.** (2019). A survey of object classification and detection based on 2-D/3-D data. *CoRR*, **abs/1905.12683**. URL http://arxiv.org/abs/1905.12683.

69. **Simonyan, K.** and **A. Zisserman**, Very deep convolutional networks for large-scale image recognition. *In* **Y. Bengio** and **Y. LeCun** (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015. URL http://arxiv.org/abs/1409.1556.

70. **Sohn, G.**, **X. Huang**, and **V. Tao** (2008). Using a binary space partitioning tree for reconstructing polyhedral building models from airborne lidar data. *Photogrammetric Engineering and Remote Sensing*, **74**, 1425–1438.

71. **Stutz, D.** and **A. Geiger** (2020). Learning 3-D shape completion under weak supervision. *International Journal of Computer Vision*, **128**(5), 1162–1181.

72. **Su, H.**, **S. Maji**, **E. Kalogerakis**, and **E. G. Learned-Miller**, Multi-view convolutional neural networks for 3-D shape recognition. *In 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*. IEEE Computer Society, 2015, 945–953. URL https://doi.org/10.1109/ICCV.2015.114.

73. **Su, J.**, **M. Gadelha**, **R. Wang**, and **S. Maji**, A deeper look at 3-D shape classifiers. *In* **L. Leal-Taixé** and **S. Roth** (eds.), *Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11131 of *Lecture Notes in Computer Science*. Springer, 2018, 645–661. URL https://doi.org/10.1007/978-3-030-11015-4_49.

74. **Sung, M.**, **V. G. Kim**, **R. Angst**, and **L. J. Guibas** (2015). Data-driven structural priors for shape completion. *ACM Trans. Graph.*, **34**(6), 175:1–175:11. URL https://doi.org/10.1145/2816795.2818094.

75. **Taghanaki, S. A.**, **J. Luo**, **R. Zhang**, **Y. Wang**, **P. K. Jayaraman**, and **K. M. Jatavallabhula** (2020). Robustpointset: A dataset for benchmarking robustness of point cloud classifiers. URL http://arxiv.org/abs/2011.11572. Cite arxiv:2011.11572Comment: Published at the Robust and Reliable Machine Learning in the Real World Workshop, ICLR 2021.

76. **Tarsha-Kurdi, F.**, **T. Landes**, and **P. Grussenmeyer** (2008). Extended ransac algorithm for automatic detection of building roof planes from lidar data. *The Photogrammetric Journal of Finland*, **21**, 97–109.

77. **Tchapmi, L. P.**, **V. Kosaraju**, **H. Rezatofighi**, **I. Reid**, and **S. Savarese**, Topnet: Structural point cloud decoder. *In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, 383–392.

78. **Vanegas, C. A.**, **D. G. Aliaga**, and **B. Benes** (2012). Automatic extraction of manhattan-world building masses from 3-D laser range scans. *IEEE Transactions on Visualization and Computer Graphics*, **18**(10), 1627–1637.

79. **Varney, N.**, **V. K. Asari**, and **Q. Graehling**, Dales: A large-scale aerial lidar data set for semantic segmentation. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, 186–187.

80. **Vaswani, A.**, **N. Shazeer**, **N. Parmar**, **J. Uszkoreit**, **L. Jones**, **A. N. Gomez**, **L. u. Kaiser**, and **I. Polosukhin**, Attention is all you need. *In* **I. Guyon**, **U. V. Luxburg**, **S. Bengio**, **H. Wallach**, **R. Fergus**, **S. Vishwanathan**, and **R. Garnett** (eds.), *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, 5998–6008. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

81. **Verma, V.**, **R. Kumar**, and **S. Hsu**, 3-D building detection and modeling from aerial lidar data. *In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2. 2006, 2213–2220.

82. **Vosselman, G.** and **H.-G. Maas**, *Airborne and Terrestrial Laser Scanning.*. Whittles Publishing, 2010. ISBN 978-1-904445-87-6.

83. **Wang, R.**, **J. Peethambaran**, and **D. Chen** (2018). Lidar point clouds to 3-d urban models: a review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **11**(2), 606–627.

84. **Wang, Y.**, **Y. Sun**, **Z. Liu**, **S. E. Sarma**, **M. M. Bronstein**, and **J. M. Solomon** (2019). Dynamic graph CNN for learning on point clouds. *ACM Trans. Graph.*, **38**(5), 146:1–146:12. URL https://doi.org/10.1145/3326362.

85. **Wichmann, A.** (2018). *Grammar-guided reconstruction of semantic 3-D building models from airborne LiDAR data using half-space modeling*. Doctoral thesis,

Technische Universität Berlin, Berlin. URL http://dx.doi.org/10.14279/depositonce-6803.

86. **Wichmann, A.**, **A. Agoub**, and **M. Kada** (2018). Roofn3d: Deep learning training data for 3-D building reconstruction. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, **XLII-2**, 1191–1198. URL https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-2/1191/2018/.

87. **Wu, B.**, **B. Yu**, **Q. Wu**, **S. Yao**, **F. Zhao**, **W. Mao**, and **J. Wu** (2017). A graph-based approach for 3-D building model reconstruction from airborne lidar point clouds. *Remote Sensing*, **9**(1). ISSN 2072-4292. URL https://www.mdpi.com/2072-4292/9/1/92.

88. **Wu, Z.**, **S. Song**, **A. Khosla**, **F. Yu**, **L. Zhang**, **X. Tang**, and **J. Xiao**, 3-D shapenets: A deep representation for volumetric shapes. *In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, 1912–1920.

89. **Xie, H.**, **H. Yao**, **S. Zhou**, **J. Mao**, **S. Zhang**, and **W. Sun**, Grnet: Gridding residual network for dense point cloud completion. *In* **A. Vedaldi**, **H. Bischof**, **T. Brox**, and **J. Frahm** (eds.), *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part IX*, volume 12354 of *Lecture Notes in Computer Science*. Springer, 2020, 365–381. URL https://doi.org/10.1007/978-3-030-58545-7_21.

90. **Xie, J.**, **G. Dai**, **F. Zhu**, **E. K. Wong**, and **Y. Fang** (2017). Deepshape: Deep-learned shape descriptor for 3-D shape retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**(7), 1335–1345.

91. **Xie, S.**, **S. Liu**, **Z. Chen**, and **Z. Tu**, Attentional shapecontextnet for point cloud recognition. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, 4606–4615.

92. **Yan, J.**, **J. Shan**, and **W. Jiang** (2014). A global optimization approach to roof segmentation from airborne lidar point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, **94**, 183–193. ISSN 0924-2716. URL https://www.sciencedirect.com/science/article/pii/S0924271614001178.

93. **Yang, J.**, **Q. Zhang**, **B. Ni**, **L. Li**, **J. Liu**, **M. Zhou**, and **Q. Tian**, Modeling point clouds with self-attention and gumbel subset sampling. *In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, 3323–3332. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Yang_Modeling_Point_Clouds_With_Self-Attention_and_Gumbel_Subset_Sampling_CVPR_2019_paper.html.

94. **Yang, Y.**, **C. Feng**, **Y. Shen**, and **D. Tian**, Foldingnet: Point cloud auto-encoder via deep grid deformation. *In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT,*

USA, June 18-22, 2018. IEEE Computer Society, 2018, 206–215. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Yang_FoldingNet_Point_Cloud_CVPR_2018_paper.html.

95. **Yuan, W.**, **T. Khot**, **D. Held**, **C. Mertz**, and **M. Hebert**, PCN: point completion network. *In 2018 International Conference on 3-D Vision, 3-DV 2018, Verona, Italy, September 5-8, 2018*. IEEE Computer Society, 2018, 728–737. URL https://doi.org/10.1109/3DV.2018.00088.

96. **Zhang, D.**, **J. Han**, **C. Li**, **J. Wang**, and **X. Li** (2016). Detection of co-salient objects by looking deep and wide. *Int. J. Comput. Vision*, **120**(2), 215–232. ISSN 0920-5691. URL http://dx.doi.org/10.1007/s11263-016-0907-4.

97. **Zhang, D.**, **D. Meng**, and **J. Han** (2017). Co-saliency detection via a self-paced multiple-instance learning framework. *IEEE Trans. Pattern Anal. Mach. Intell.*, **39**(5), 865–878. ISSN 0162-8828. URL https://doi.org/10.1109/TPAMI.2016.2567393.

98. **Zhang, L.** and **L. Zhang** (2018). Deep learning-based classification and reconstruction of residential scenes from large-scale point clouds. *IEEE Trans. Geoscience and Remote Sensing*, **56**(4), 1887–1897. URL https://doi.org/10.1109/TGRS.2017.2769120.

99. **Zhang, W.**, **C. Long**, **Q. Yan**, **A. L. Chow**, and **C. Xiao** (2020). Multi-stage point completion network with critical set supervision. *Computer Aided Geometric Design*, **82**, 101925. ISSN 0167-8396. URL https://www.sciencedirect.com/science/article/pii/S0167839620301126.

100. **Zhang, X.**, **A. Zang**, **G. Agam**, and **X. Chen**, Learning from synthetic models for roof style classification in point clouds. *In Proceedings of the 22Nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '14. ACM, New York, NY, USA, 2014. ISBN 978-1-4503-3131-9, 263–270. URL http://doi.acm.org/10.1145/2666310.2666407.

101. **Zhou, Q.-Y.** and **U. Neumann**, Fast and extensible building modeling from airborne lidar data. *In Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '08. Association for Computing Machinery, New York, NY, USA, 2008. ISBN 9781605583235, 1–8. URL https://doi.org/10.1145/1463434.1463444.

102. **Zhou, Q.-Y.** and **U. Neumann**, 2.5-D dual contouring: A robust approach to creating building models from aerial lidar point clouds. ECCV'10. Springer-Verlag, Berlin, Heidelberg, 2010. ISBN 364215557X, 115–128.